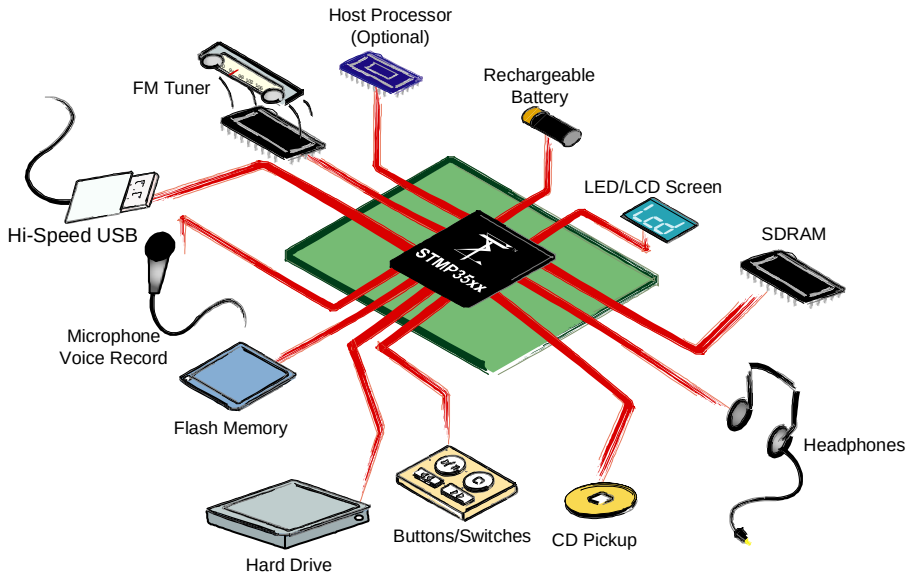
Integrated Mixed-Signal Solutions

PRODUCT DATA SHEET

STMP35xx

D-Major™ Audio System on Chip with USB 2.0, LCD, Voice Record and Battery Charger

Third Generation Audio Decoder
Version 1.03 September 17, 2003



PRELIMINARY INFORMATION 9/17/03

5-35xx-D1-1.03-091703

1. TABLE OF CONTENTS

ADDITIONAL SUPPORT

Additional product and company information can be obtained by going to the Sigma-Tel website at:***www.sigmatel.com***. Additional product and design information is available for authorized customers at: ***extranet.sigmatel.com***

2. PRODUCT OVERVIEW

2.1. Features

- Decodes MP3 and WMA and is upgradeable to other digital music formats
- Supports WMA Digital Rights Management (DRM) and other security schemes
- Includes on-chip read only unique ID for digital rights management algorithms
- USB High Speed Device Interface (up to 480Mb/s transfers)
  - Enables file transfer and firmware upgrade using USB Mass Storage Class
  - Both Windows and Macintosh drivers available
  - Integrated USB High Speed PHY
  - Direct connection to USB 5V power for operation and battery charging
- 96K Words (288K Bytes) of on-chip RAM
- Hardware support for flexible external storage options
  - NAND Flash, MMC, Secure Digital, SmartMedia, CompactFlash
  - Five byte address support for new 1Gb/die (128KB block) NAND Flash
  - MLC NAND Flash support
  - 1.8V NAND Interface Support
  - 16 bit wide NAND support
  - Hardware accelerated ECC offloads DSP bit error correction
  - SDRAM
  - ATA/IDE Hard Disk digital devices.
- Optimized for very long battery life
  - 50 hours of operation on a single AA battery
  - Flexible, efficient on-chip DC-DC converter
  - Flexible battery configurations, including 1xAA, 1xAAA, 2xAA, 2xAAA, LiIon
  - Pulse frequency modulation mode for low standby power
  - Energy saving dynamic power management
  - Typical off current is 250µA (crystal oscillator & real time clock only)
  - More than 1 year battery life in *"off"* mode on one AA Alkaline battery
  - Integrated battery charger for LiIon and NiMH
  - Battery temperature sensor support for safest charging protocols
  - Real time clock with alarm function wakes up from powerdown/standby modes
- High quality integrated audio mixed signal sub-system
  - <0.05% THD direct drive headphone amplifier
    - Eliminates DC blocking capacitors
    - Including anti-pop and short-circuit protection
  - High performance 18-bit ΣΔ technology stereo D/A and A/D converters
  - Full analog mixer configuration
  - Line-in to Headphone/Line-out SNR >90 dB
  - Two analog line-level inputs: Line1 In (stereo), Line2 In (stereo, 144-pin package)
  - Mic(mono) input with integrated pre-amp and microphone biasing circuit
  - Volume control
- GPIO, button I/O controls, and LCD/LED Display Compatible Interface
- Pulse Width Modulators for EL backlights
- Integrated 75MHz DSP with Filter Coprocessor for power optimization
  - Optimized for audio applications
  - Field upgradeable firmware

- Integrated Development Environment, SDK, and debugger
- Application and support libraries
- Bass and Treble control; configurable multiple band EQ control
- Voice record in ADPCM format (upgradable to other formats)

- FM tuner input and control support
- Optional interface to a host chip/processor for cell phone & PDA applications, etc.
- Application notes, reference schematics, sample PCB layouts are available.
- Offered in 100-pin TQFP, and 144-pin fpBGA packages
- Backward pin and firmware compatible with STMP3410

2.2. STMP35xx Block Diagram

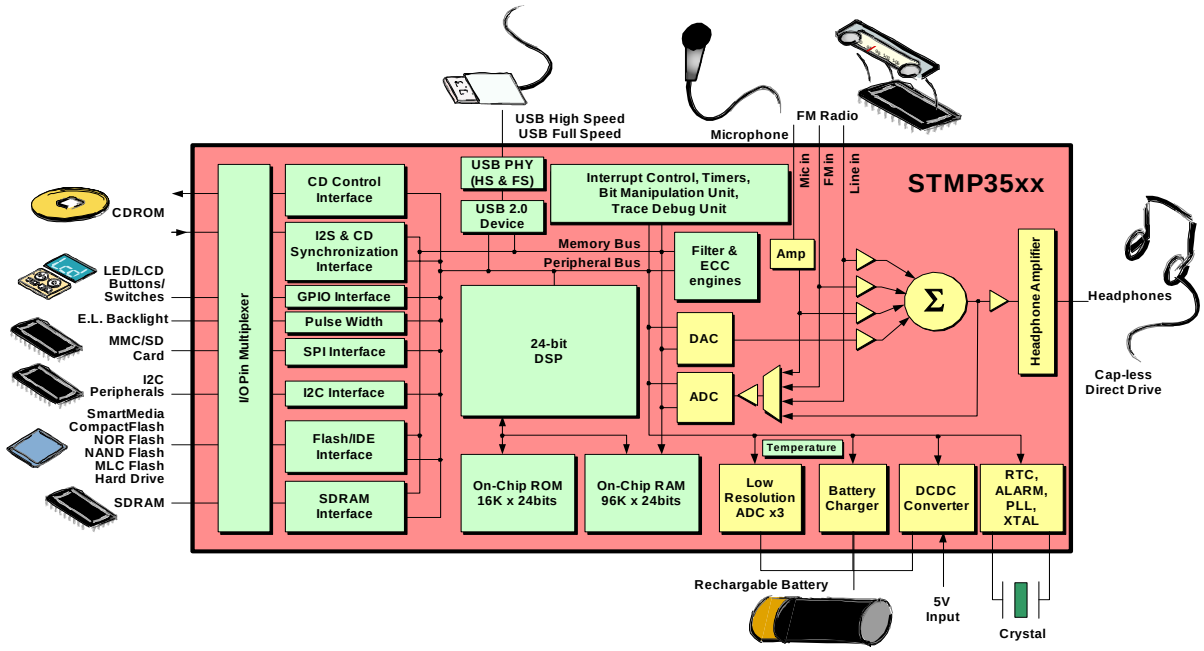
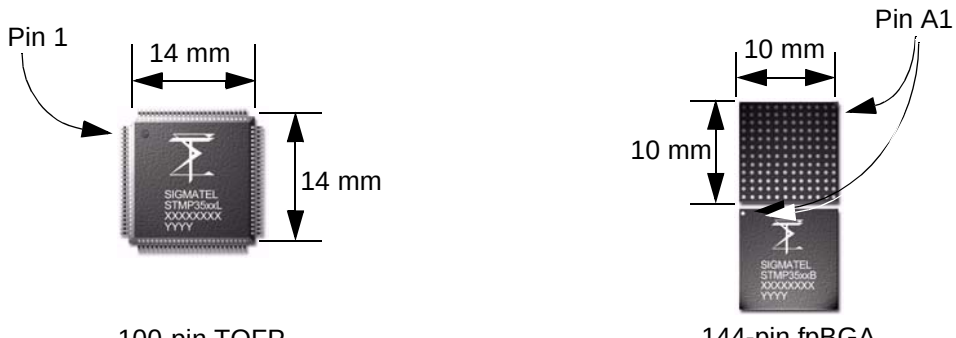


Figure 1. Chip Block Diagram



100-pin TQFP 144-pin fpBGA

For additional package measurements, please see 32. “PACKAGE DRAWINGS” on page 380.

Figure 2. Chip Package Photos

2.3. Description

SigmaTel's STMP35xx is a third generation single-chip highly-integrated digital music system solution for devices such as digital audio players, PDAs, voice recorders, MP3-encode recorders, and cell phones. It includes a high performance DSP, 288KBytes of on-chip SRAM, and a USB 2.0 interface (including High speed 480Mb/second transfers) for downloading music and uploading voice and MP3 recordings. The chip also includes a mixer, DAC, ADC and provides interfaces to IDE Hard Drives, CD-DSPs, Flash memory, LCD/LEDs, button & switch inputs, headphone driver, FM tuner input & controls and a microphone. The chip's highly programmable architecture supports MP3, WMA, and other digital audio standards. WMA digital rights management and other security schemes are also supported. For devices like PDAs and cell phones, the STMP35xx can act as a slave chip to a host chip/processor.

The DAC includes a headphone driver to directly drive low impedance headphones. The ADC includes inputs for both microphone and analog audio in to support voice recording & FM radio integration and MP3 encode features. SigmaTel's proprietary Sigma-Delta (ΣΔ) technology achieves a DAC SNR in excess of 90 dB for high-quality audio playback.

The STMP35xx has low power consumption to allow long battery life and includes an efficient flexible on-chip DC-DC converter that allows many different battery configurations, including 1xAA, 1xAAA, 2xAA, 2xAAA and LiIon. The chip includes an integrated intelligent charger for NiMH and LiIon batteries. In addition, the single-chip design and low pin count enables very small digital audio devices to be designed.

2.3.1. DSP Core

The on-chip DSP core is modeled exactly after the Motorola DSP56004. It supports the identical instruction set, registers, addressing modes, etc., as the DSP56000 family of digital signal processors. Figure 3 shows a high level view of the DSP core. This architecture is highly optimized for battery operated audio applications. Its 24-bit intrinsic data size provides sufficient precision for high quality audio algorithms while minimizing the number of register and data path signals that must be toggled for any operation. The term "*WORD*", as used in this data sheet, refers to a 24-bit unit of storage unless otherwise noted.

The functionality that defines the on-chip DSP, is the memory map, interrupt processing, and peripherals it offers.

The integrated DSP comprises three execution units, an interrupt controller and a debug interface. It connects to the rest of the STMP35xx chip via three memory busses, a set of interrupt input signals and various reset and clock inputs. It implements a 3 memory space Harvard architecture, simultaneously referencing an X data element, a Y data element and a program element. These references are conveyed over the program or "P" bus, the X bus and the Y bus. Each bus comprises a 24 bit wide data path and a 16 bit address bus. Program accessible I/O registers reside in the top 4K word addresses on the X-bus. The DSP architecture has special programmed I/O support for the top 64 words of this space but SigmaTel has extended this space to the top 4K words, i.e. addresses $F000 through $FFFF, inclusive.

The DSP Core also implements the OnCE debugger that is the norm for this DSP architecture. The OnCE interface connects to an external debugger over four I/O signal pins on the STMP35xx.

Using an industry standard instruction set architecture and debugger interface for the integrated DSP means that development tools and debuggers are in the highly evolved and stable portion of their life cycle. In addition, it means that system devel-

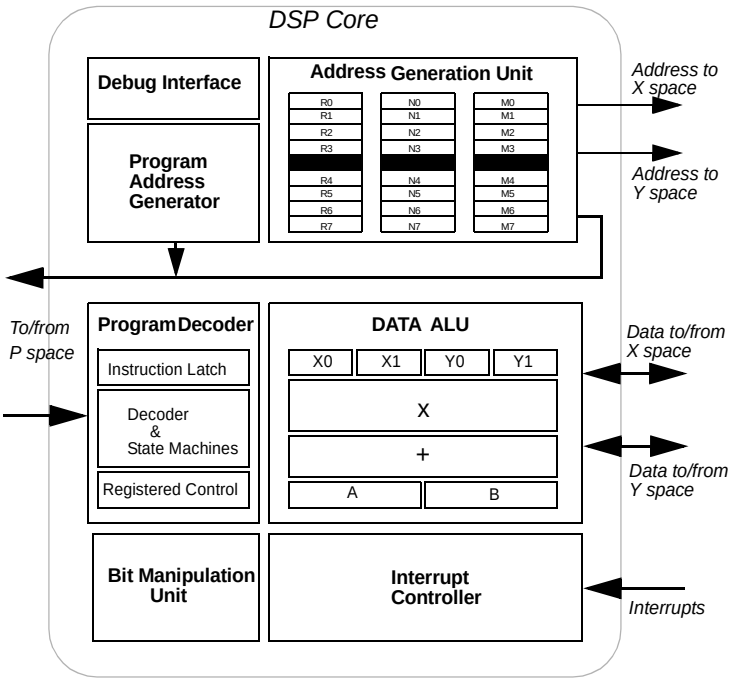


Figure 3. DSP Core at a Glance

opers with experience developing on this DSP can be found. The SigmaTel software developers kit (SDK) provides an excellent integrated development environment with an assembler, C compiler, debugger and other requisite tools.

2.3.2. On-chip RAM and ROM

The STMP35xx includes 96K words of on-chip RAM. This amounts to 2.25Mbits of on-chip SRAM in six 16K Word blocks. The RAM is split into two 48K word banks with one bank attached to the X bus and one attached to the Y bus. The P bus is

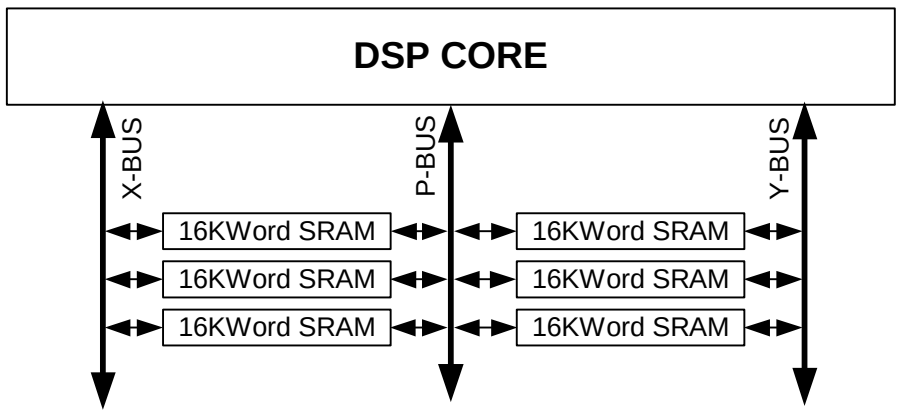


Figure 4. 6 x 16K Word On-chip SRAM Blocks

connected to both RAM banks so that program space can be allocated from the same two banks that hold X and Y data values. An adjustable switching mechanism

is provided so that a portion of the X bus RAM or Y bus RAM can be allocated to the P bus in units of 8K words from 0K to the full 64K words.

A typical application will allocate a portion of the X bus RAM and another portion of the Y bus RAM to the P bus. In normal operation, this switching mechanism will present a contiguous block of RAM beginning at location zero in the P bus address space, or P:$0000 as it is written in assembler syntax

Suppose we allocate 24K Words from the X bus RAM to the P bus and another 24K Words from the Y bus RAM to the P Bus. This gives us 24K Words of X RAM, 24K Words of Y RAM and 48K Words of P RAM for our application. All 96K Words of on-chip SRAM are allocated, as shown in Figure 5.

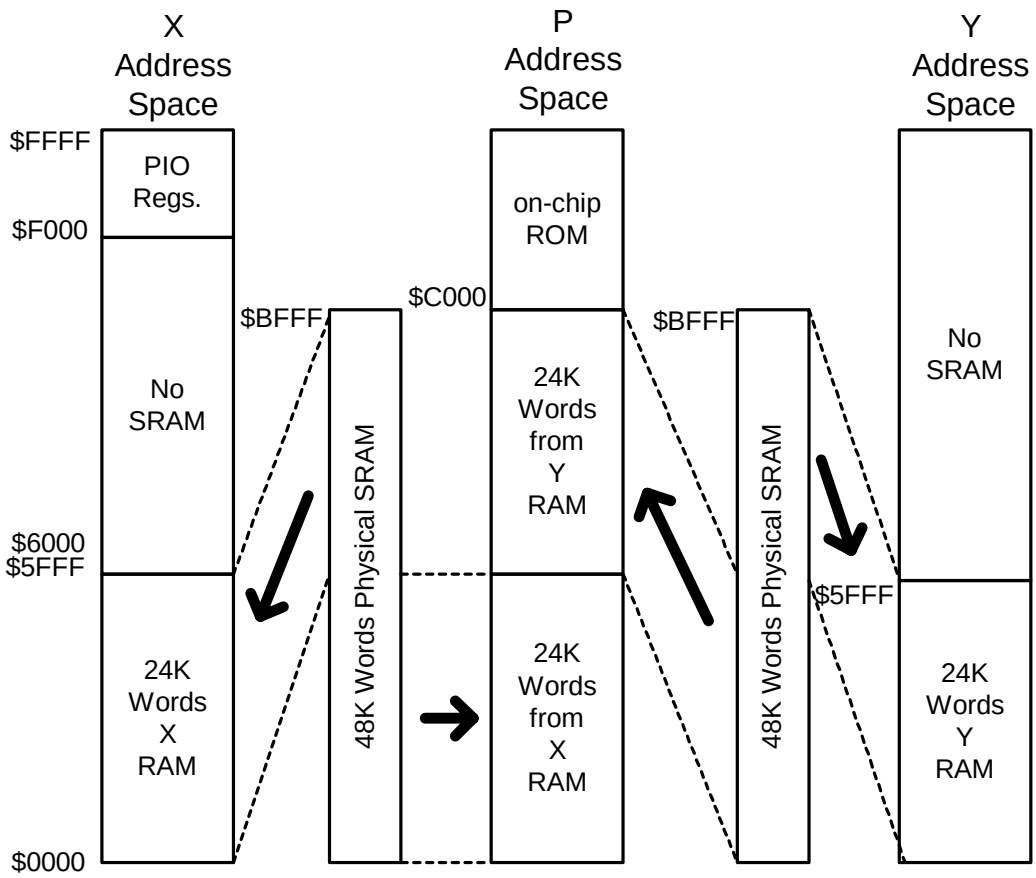


Figure 5. On-Chip RAM Allocation Example

The STMP35xx contains an on-chip 16K Word ROM which holds the Bootstrap code. At power-on time, the first instruction executed by the DSP comes from this ROM. Power-on reset causes the on-chip ROM to be placed at P:$0000. The reset interrupt vector is located at P:$0000, thus the first instructions executed come from this ROM. Software in this ROM offers a large number of BOOT configuration options, including manufacturing boot modes for *"burn-in"* and *"tester"* operation.

Other boot modes are responsible for loading application code from off-chip into the on-chip RAM. Off chip sources for application bootstrapping include:

- External NAND FLASH
- Host (PC) controlled bootstrapping using USB,

- Host (PC) controlled bootstrapping using I^2C slave.
- I^2C Master transfers from serial EEPROM

Once the on-chip boot code has loaded the application code into on-chip RAM, it can relocate the 16K Word on-chip ROM to the very top of the P address space, see Figure 5. “On-Chip RAM Allocation Example” on page 7. The on-chip ROM can be disabled entirely so that all 64K words of P space is available for on-chip RAM.

The on-chip boot code includes a firmware recovery mode. If the device fails to boot from NAND flash, for example, the device will boot from a PC host connected to its USB port. This firmware recovery mode can be invoked at anytime by holding the PSWITCH or “play” button for at least five seconds during power up.

The on-chip RAM serves as one end of all DMA transfers, e.g either the source or destination. Every SRAM block has three potential accessors: P-BUS, DMA-BUS, and its respective X-BUS or Y-BUS. A number of the integrated peripheral controllers use a distributed DMA implementation to transfer data to or from on-chip SRAM.

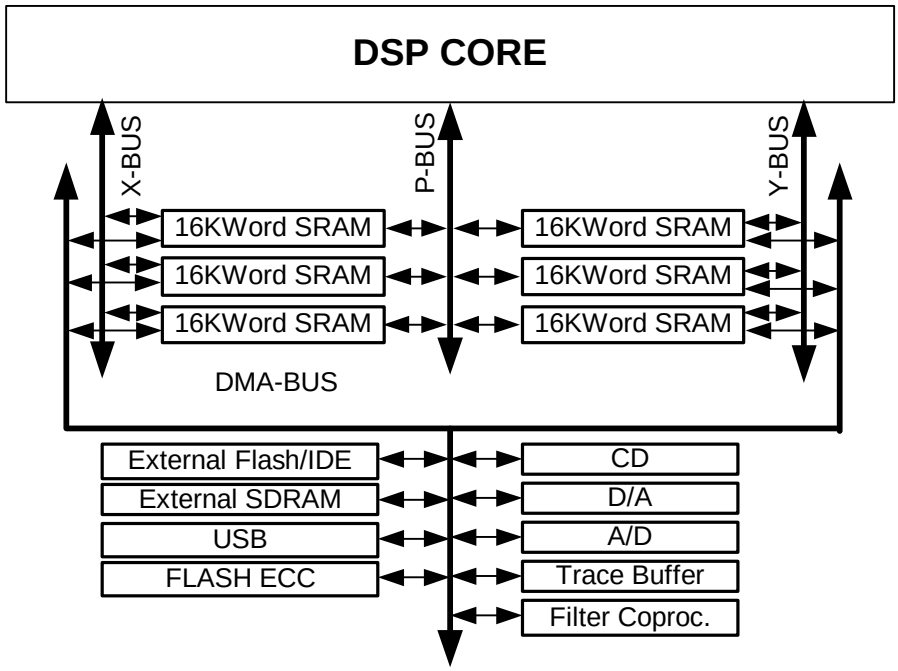


Figure 6. On-Chip RAM & Distributed DMA

In this distributed DMA architecture, all of the peripheral controllers that use DMA share a common DMA address and data bus path to and from on-chip RAM. Each peripheral controller implements its own address generator. Address generation can be highly sequential as in the case of the D/A converter or fairly random as in the case of the USB controller. Thus each device that uses the distributed DMA will have at least one base address register (HW_xxxBAR) and various address modifying registers. Most of the distributed DMA devices implement some form of circular buffering in their addressing modes. There is a centralized arbiter that selects which of the distributed DMA peripheral controllers has access to the DMA bus on any given clock cycle.

With a DSP core clock of 65MHz, a single block of on-chip RAM can provide 65MHz times 3 bytes or 195 MByte/second of bandwidth. There are four 24 bit data busses connected to the on-chip SRAM blocks. Furthermore, each SRAM block is single ported and has its own independent address and data busses. Thanks to the arbitration logic in this memory subsystem, all four busses (P, X, Y, DMA) can be made to cycle on every clock. Thus the peak bandwidth available from the on-chip RAM is four times 195 MByte/second or 780 MByte/second.

Of course, there are times when more than one data bus needs to transfer into or out of the same SRAM block. When conflicts occur, the arbiter will *"stall"* the DSP for one (or more) clock(s) to resolve the conflict.

The reader should not be surprised to see devices like A/D or D/A converters using DMA transfers. Some readers may be surprised to learn that external FLASH and external SDRAM are only accessible via the DMA. The external memories are *not* mapped into the *"load/store"* space of the DSP's instruction set.

2.3.3. Power Subsystem

The STMP35xx contains a sophisticated power subsystem including two integrated DC to DC converters to produce a very cost effective product with flexible battery configurations. In addition, it contains power monitoring circuits for battery brownout detection as well as system overload brownout detection. The chip also contains detection circuits for battery installation and removal. It manages power state changes caused by battery changes or from monitoring the on/off power switch circuit.

The chip has two programmable integrated DC-DC converters that can be used to provide power for the device *as well as the entire application*. The converters can be configured to operate from standard battery chemistries in the range of 0.9-4.2 volts including alkaline cells, NiMH, LiIon etc. These converters use off chip reactive components (L/C) in a pulse width or frequency modulated DC to DC converter.

The DC to DC converter circuit consists of the off-chip reactive components, an integrated controller and integrated low resistance FET switches. The DC-DC converter #1, as shown in Figure 7, has one n-channel FET and three independently controlled p-channel FETs generating three independent channels of separately controlled voltages. For the case shown, the battery is a single AA alkaline battery in the range 0.9 to 1.5 volts. DC-DC converter #1 is used to *"boost"* this input voltage to 3.3 volts for use in driving the I/O VDD rail and two separate 1.8 volt sources for driving the analog VDD rail and the digital VDD rail. This case is shown in the first row of Table 1, "Flexible Battery Configurations," on page 11. Other rows show different configurations supported by the DC to DC converters. For example, when the battery chemistry provides an input voltage that is higher than that desired for the I/O rail, digital rail or analog rail, then the DC to DC converters can operate in *"buck"* mode which provides a regulated output that is lower than its input.

One obvious use for the DCDC converter is in boosting the output of a nearly depleted alkaline battery delivering 0.9 volts up to the regulated 3.3 volt I/O rail voltage and the regulated 1.8 volt digital and core rails. The DCDC converter can also be used to lower the voltage of a 4.2 volt LiIon battery down to the 1.8V digital core and analog rails. Table 1, "Flexible Battery Configurations," on page 11 shows various battery configurations that can be supported.[1]

1.Note VddA3 == VddHP, VddA4== VddPLL. The analog power pair formerly known, in the STMP3410, as VddA2 and VssA2 have been redefined for the STMP35xx as Vdd5V and LRADC2.

In addition, the DCDC converter can regulate these voltages to lower the standard core and I/O values to extend the battery life. Recall that energy consumed in a CMOS AC circuit is proportional to V^2 so this reduction can be quite significant. These program controlled reductions in operating voltage are used in various SigmaTel software applications to provide very long battery life products. The STMP35xx also contains a silicon speed sensor so that each device can tailor its operating voltage to the minimum required for safe operation as constrained by its individual silicon process parameters and junction temperature.

The DC to DC converters control the power up sequence of the device and hold the rest of the chip in reset until the power supplies have stabilized at the correct voltages. The power up sequence begins when the battery is connected to the BATT pin. As shown in Figure 7, the crystal oscillator will begin running as soon as the battery is connected ***and*** the pswitch is asserted. The crystal oscillator and the real time clock (RTC) can be programmed to continue to operate even when the player is in the off state. The crystal oscillator and RTC are the only drains on the battery in the off state and designed for very small energy consumption. The RTC module includes an alarm function that can be used to "wake-up" the DC to DC converters which will then wake up the rest of the system.

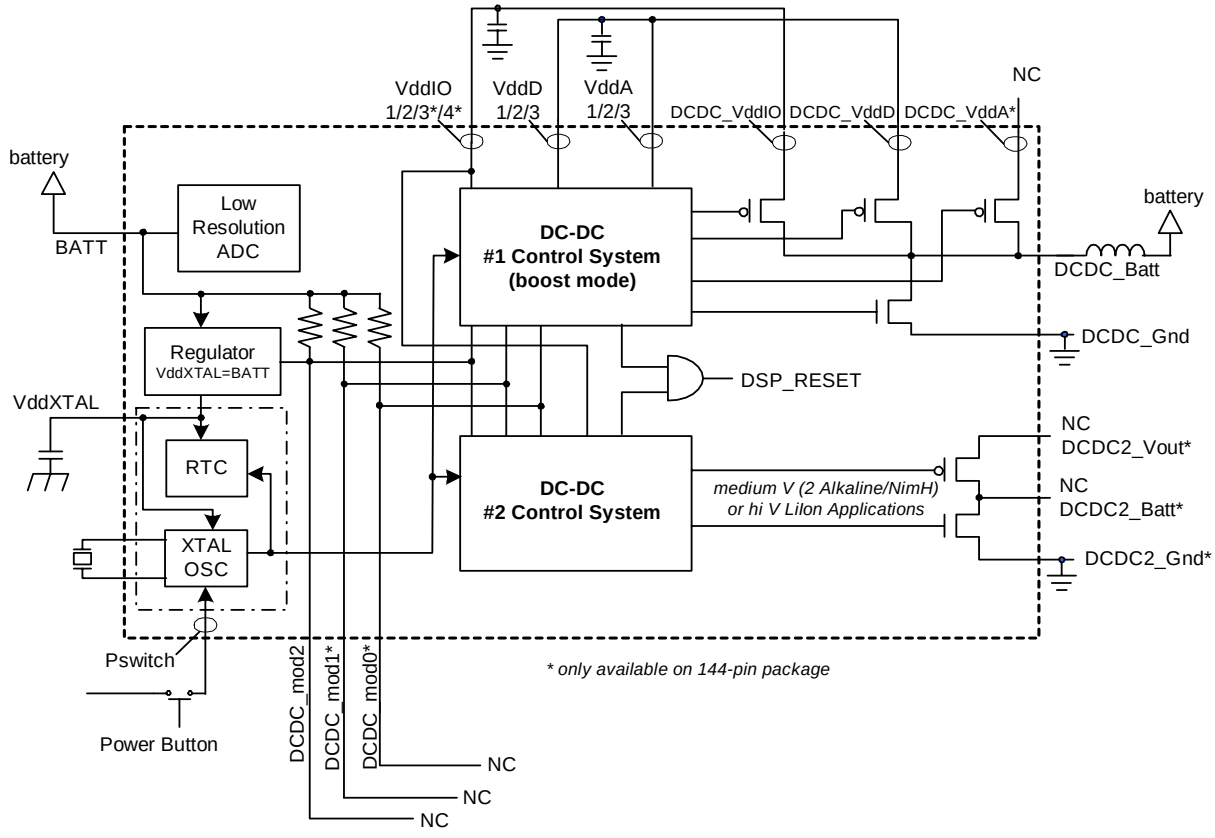


Figure 7. Lowest Cost 1xAA 100-pin Configuration[1]

The power down sequence is also controlled by the DC to DC converters. When a power down event is detected, they return the player to the power off state. In the power off state with non-LiIon chemistries, the I/O Vdd rail is connected to the Battery and the internal VddD and VddA rails are pulled down to ground to minimize

leakage currents. For LiIon mode, the I/O Vdd rail is connected to ground instead of connecting to BATT.

In addition to the various voltage detectors, a power up or power down event can be signaled by the special power switch circuitry in the DC to DC converters. A simple resistor network and momentary contact push button switch is sufficient for player on/off control.

There is a special three channel low resolution A/D converter on-chip to help with battery based applications. One channel is dedicated to measuring the voltage on the BATT pin and is used to monitor the battery condition to estimate its remaining life. All low resolution channel also have digital trip point comparator functions that can be used to generate interrupts to the DSP. The trip point can be programmatically set at one of 512 levels for battery brown out detection on the Battery LRADC or for threshold detection on the other two LRADCs. NOTE: *ONLY* the battery can be connected directly to the BATT pin for correct operation of the device, thus the battery channel of the low resolution A/D converter is not available for any other purpose. The second and third low resolution A/D converters are uncommitted and available for application use. An optional current source can be enabled to either the second or third LRADC pin to support external temperature sensors with minimal external components.

In addition, the DC to DC converters have comparators to monitor their output voltages. They can report “brownout” conditions resulting from over drawing their power capabilities. These conditions are reported either on a normal interrupt level or as a non-maskable interrupt (NMI).

The device contains an integrated PLL which is referenced to the 24.0MHz crystal oscillator. It can generate clock sources from 39.6MHz to 120.0MHz in steps of 1.2MHz. It includes a post divide stage for the digital clock from a divide by one to a divide by 2048. With the PLL turned off and the post divider set to 2048, one can achieve a low power 11.7KHz operating point.

POWER SOURCE	VDD I/O	VDD D	VDD A
1 Alkaline or 1 NiMH (0.9V-1.5V)	DCDC1 DCDC_VddIO Boost 3.3V	DCDC1 DCDC_VddD Boost 1.8V	DCDC1 DCDC_VddA Boost 1.8V better noise floor (144-pin)
1 Alkaline or 1 NiMH (0.9V-1.5V)	DCDC1 DCDC_VddIO Boost 3.3V	DCDC1 DCDC_VddD Boost 1.8V lowest cost (shared passives & 100-pin)	DCDC1 DCDC_VddD Boost 1.8V
LiIon, (3.0-3.6V)	LiIon Battery	DCDC1 Buck 1.8V	DCDC1 Buck 1.8V
2 Alkaline or 2 NiMH (1.8V-3.0V)	DCDC2 Boost 3.3V	DCDC1 Buck 1.8v	DCDC1 Buck 1.8V
LiIon (3.3V-4.2V)	DCDC2 Buck 3.3V	DCDC1 Buck 1.8V	DCDC1 Buck 1.8V

Table 1. Flexible Battery Configurations

There is an integrated watchdog reset timer available for automatic recovery from catastrophic software errors. If programmed by software, this circuit will generate a reset sequence if its timer is ever allowed to reach zero. Normally functioning software will reload the watchdog count before expiration of the count. The maximum delay until a watchdog reset is greater than four hours.

2.3.4. Battery Chargers

Figure 8 shows additional detail to the DCDC converter circuit of Figure 7 which supports NiMH battery charging in a USB or AC line attached environment. In this case, 5 V is brought into the STMP35xx on the charger pin (VDD5V). A pair of on-chip linear regulators drop the incoming voltage to 3.3V for the I/O circuits and 1.8V for the digital and analog core. Additional mux (pass) transistors have been added to allow internal switching and routing of the line supplied voltage so that the I/O, digital and analog supply needs of the chip are met from the line voltage. Additional mux transistors isolate the battery pin so that it can receive charge current from the DCDC_BAT pin. The charging current path is shown as a very bold line overlaying the block diagram. The linear charger controlled current source and the DCDC converter #1 precisely controls the charging current to the program specified amount. The battery channel of the low resolution ADC is used to monitor the charging progress of the NiMH battery.

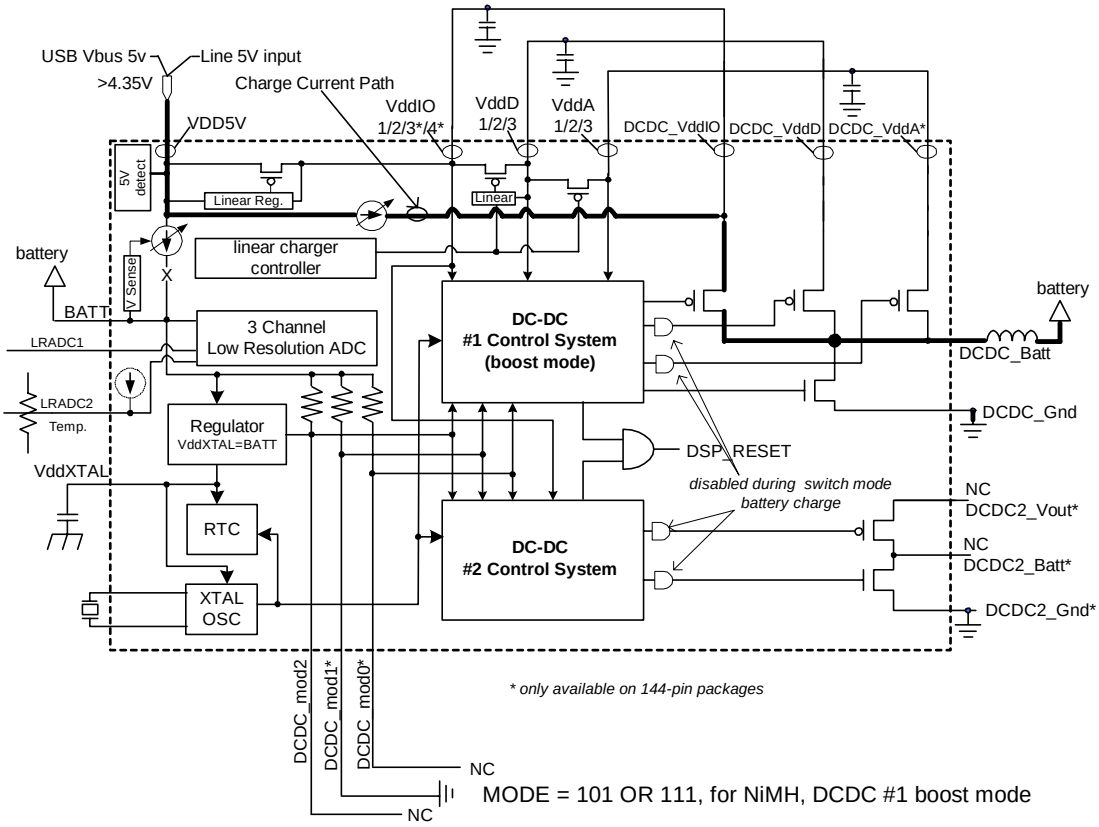


Figure 8. Integrated NiMH Battery Charger

A thermo-sensor can be attached to the second or third channel of the LRADC so that battery temperature can be monitored by software for an optimal charging cycle. Finally, software can make use of the RTC alarm function to periodically wake

up and check the charge progress. Most of the better NiMH charging protocols can be implemented in this fashion.

The STMP35xx also integrates support for LiIon charging protocols in USB or AC line attached environments. When the 5 V source is detected on the VDD5V pin the power management system automatically reconfigures to use the integrated linear regulators to supply the core and I/O rails. Software can then enable the integrated current source to provide battery charge current, as shown by the bold path in Figure 9. The variable current sources tappers of the charge current as it approaches the maximum LiIon battery voltage. Software can then take over to control the final "topping-off" algorithm, as desired.

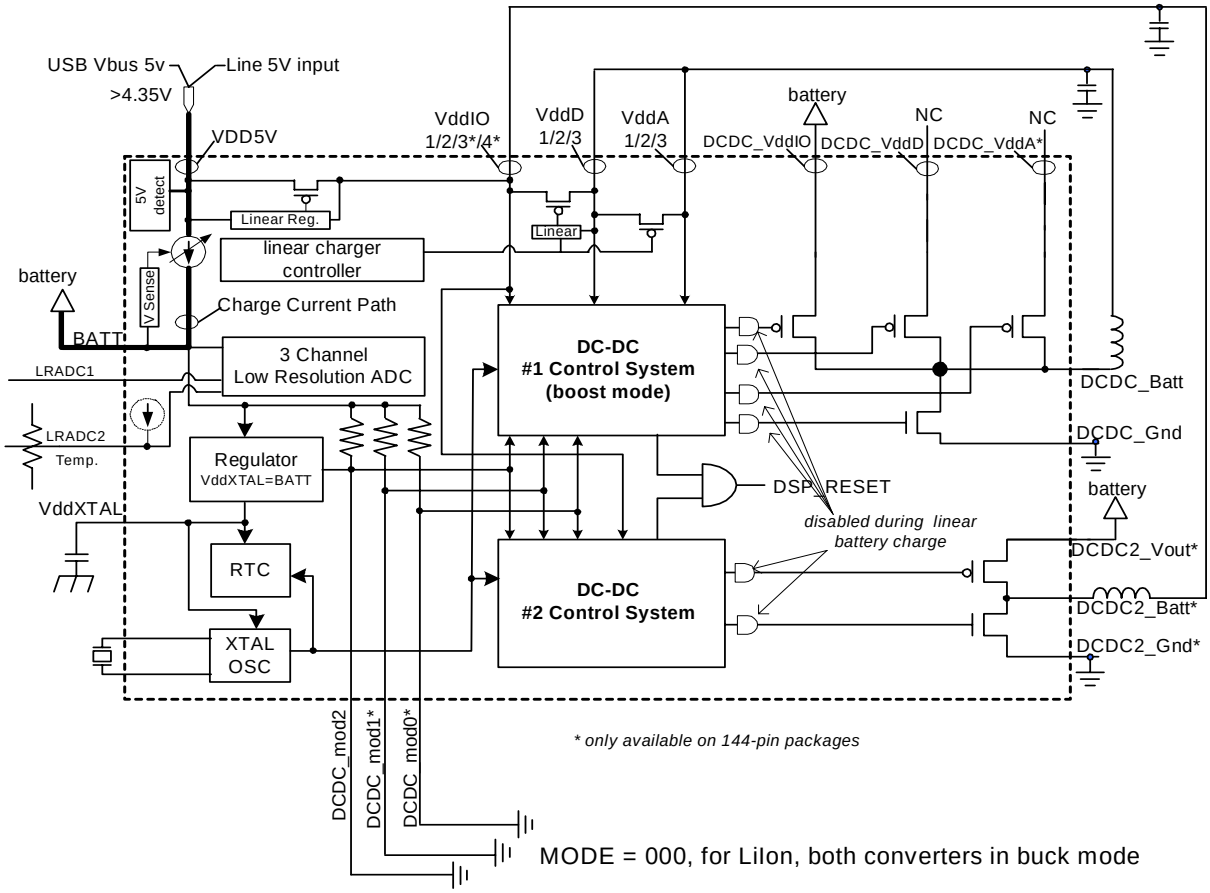


Figure 9. Integrated LiIon Battery Charger

2.3.5. *USB Interface*

The chip includes a Universal Serial Bus (USB) version 2.0 controller and integrated UTMI PHY. The STMP35xx device interface can be attached to USB 2.0 hosts and hubs running in the USB 2.0 High Speed mode at 480Mbit/second. It can be attached to USB 2.0 Full Speed interfaces at 12Mbit/second. Of course, the USB 2.0 Full Speed mode allows the STMP35xx to attach to USB 1.1 compliant hosts and hubs.

The USB interface is used to download digital music data or program code into external memory and to upload voice recordings or MP3 encoded recordings from external memory to the PC. Program updates can also be loaded into the flash memory area using the USB interface.

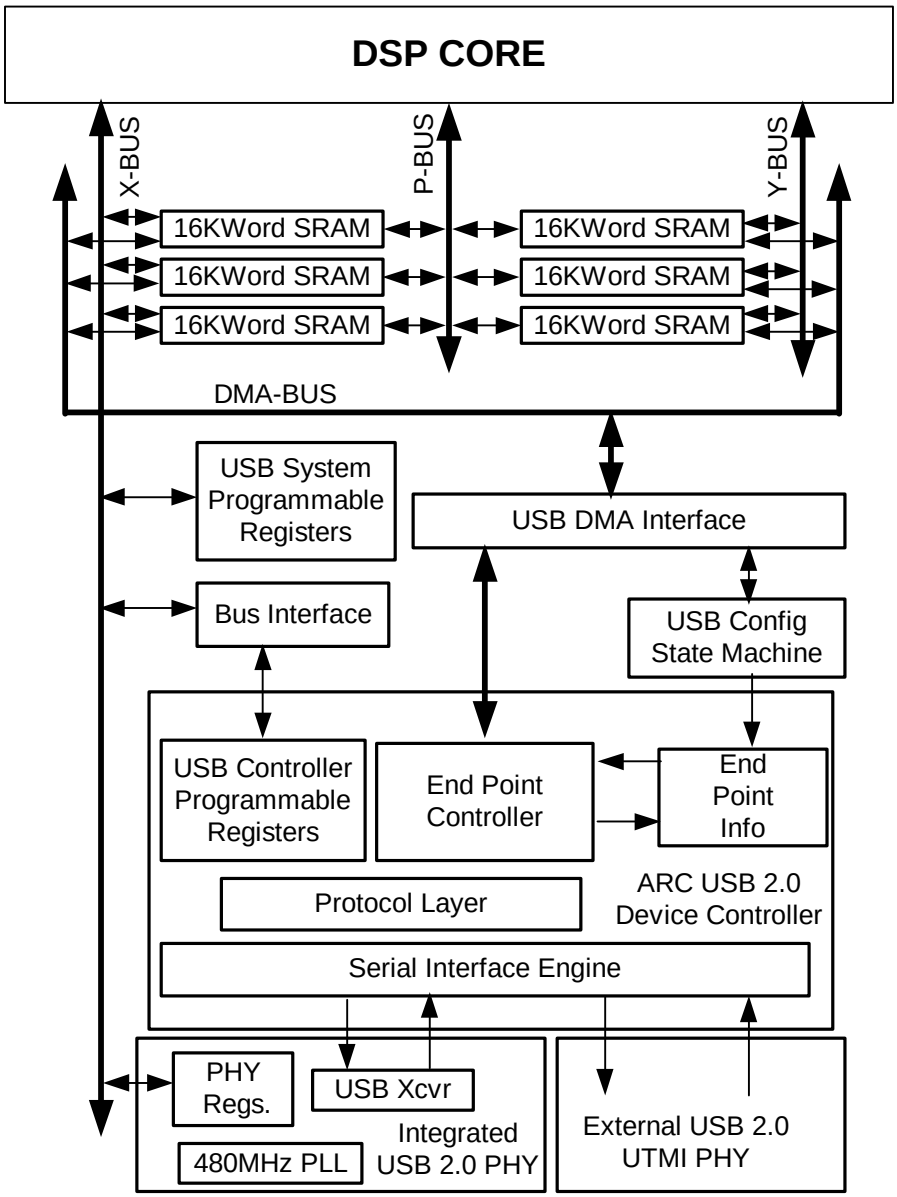


Figure 10. USB Interface Block Diagram

The Universal Serial Bus (USB) is a cable bus that supports data exchange between a host computer and a wide range of simultaneously accessible peripherals. The attached peripherals share USB bandwidth through a host-scheduled, token based protocol. The bus allows perpherals to be attached, configured, used and detached while the host and other peripherals are in operation.

The USB subsystem is designed to make efficient use of system resources within the SMTP35xx. It contains a random access DMA engine that reduces the interrupt

load on the DSP and reduces the total bus bandwidth that must be dedicated to servicing the eight on-chip physical endpoints

It is a dynamically configured port which can support up to 6 general use physical endpoints and 8 logical endpoints, each of which may be configured for bulk, interrupt or isochronous transfers. The USB configuration information is read from on-chip memory via the USB controller's DMA.

Figure 10 shows a block diagram of the USB controller. This device makes extensive use of the DMA to read and write the multiple buffers associated with all of the endpoints that it can have open at one time

2.3.6. External Memory Interfaces.

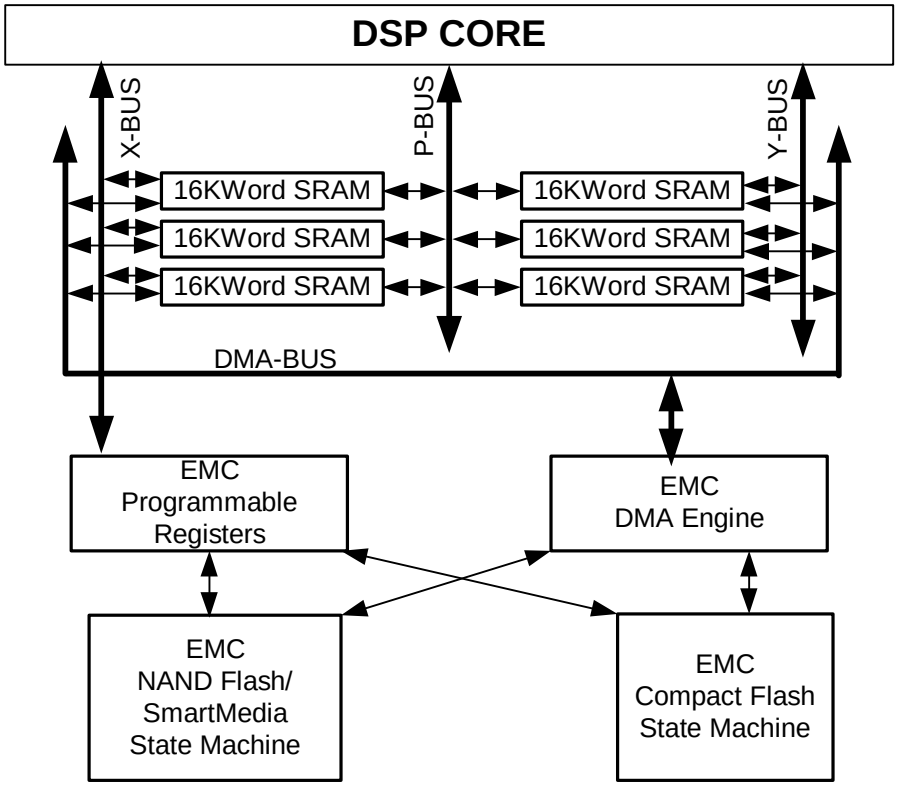


Figure 11. External Memory Controller

The chip includes an external memory controller that has two major functional modes: SmartMedia/NAND and CompactFlash. The SmartMedia/NAND flash interface provides a state machine that provides all of the logic necessary to perform DMA functions between on-chip RAM and the flash. The CompactFlash interface supports all three major CompactFlash modes: Memory, I/O and IDE. These modes can be used to communicate with standard CompactFlash (CF) devices such as CF Flash and the IBM MicroDrive. The CF Memory mode can be used to communicate with standard ATA/ATAPI devices like CD-ROM and Hard drives

The external memory controller can be described as three fairly independent devices in one: a SmartMedia/NAND flash interface based on the STMP3410 flash controller, a CompactFlash/NOR flash/IDE interface, and a new general purpose flash state machine that can support the new 1Gb/die NAND flash devices with

128KByte block erase modes. These interfaces share the same device pins, some registers and the DMA engine.

The interfaces use memory mapped registers to setup and control the transactions. Data is always sent through DMA – there are no data registers that correspond to the interface data bus. Transactions are always started with a kick bit. The interface sets up the control lines and transfers data to/from the internal RAM. Once the transaction is complete the interface signals the DSP with either a polled flag or an interrupt.

2.3.7. Hardware Acceleration for ECC for Robust External Storage

The forward error correction module is used to provide STMP35xx applications with a reliable interface to various storage media, especially storage media that would otherwise have unacceptable bit error rates. The ECC module comprises two different error correcting code processors:

- 1-bit correcting Samsung SSFDC (Hamming-code) encoder/decoder.
- 4-symbol correcting (9-bits/symbol) Reed-Solomon encoder/decoder.

The 1-bit hamming code is defined by Samsung for use with all SSSFDC compliant NAND flash memories. This code is capable of correcting a single incorrect bit over the block for which the ECC is valid (256 bytes per page).

The purpose of the Reed-Solomon decoder is to process a coded block (data block followed by “parity” check data) to determine if there is an error and, if there are errors, where they are located and how to correct them. The purpose of the Reed-Solomon encoder is to read a block of 503-symbols from RAM, calculate and append 8-parity symbols to form a 512-symbol RS-codeword.

The Hamming code error corrector is strong enough to detect two bits in error in 256 bytes and to correct 1-bit/256 byte errors. Both of these error correction encoder/decoders use DMA transfers to move data to and from on-chip RAM completely in parallel with the DSP performing other useful work.

2.3.8. Mixed Signal Audio Subsystem

The STMP35xx contains an integrated high quality mixed signal audio subsystem, including high quality sigma delta D/A and A/D converters. The D/A is of course the mainstay of the Audio Decoder/Player product application while the A/D is used for Voice Record and MP3 Encode applications.

The chip includes a low noise headphone driver that allows it to directly drive low impedance (8Ω or 16Ω) headphones. The direct drive, or “cap-less” mode removes the need for large expensive DC blocking capacitors in the headphone circuit. The headphone power amplifier can detect headphone shorts and report them via the DSP interrupt system. A digitally programmable master volume control allows user control of the headphone volume. Annoying clicks and pops are eliminated by zero crossing updates in the volume/mute circuits and by headphone driver startup and shutdown circuits.

There is an integrated analog mixer that drives the master volume control programmable gain amplifier. The chip provides for two stereo line level inputs and a mono microphone input. The microphone circuit has a mono to stereo programmable gain pre-amp and an optional microphone bias generator. The line inputs have programmable gain/attenuation and balance capability. The integrated sigma delta DAC has a programmable gain/attenuation analog amplifier. The programmable gain/attenuation stage outputs from all three stereo inputs and from the DAC are mixed together

to drive the master volume control. There is an analog mux in front of the ADC that can select any of the three input sources or the mixer output. The selected source is then sent to the ADC Gain stage and from there to the ADC. The mixer can be independently powered down. In this configuration, the mixer is bypassed so that the DAC can still play audio through the headphone driver saving power consumption and improving the SNR and THD performance.

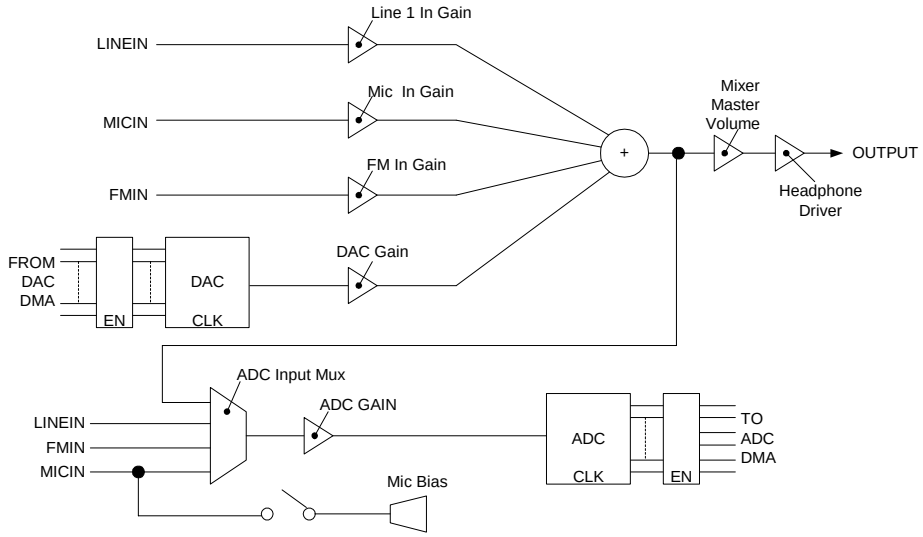


Figure 12. Mixed Signal Audio Elements

2.3.9. Filter Coprocessor

A filter coprocessor has been added to the STMP35xx to reduce the DSP load associated with filter calculations. Additional enhancements in the ADC and DAC buffer management have greatly reduced the DSP work load as compared with STMP3410 based applications. The concomitant reduction in DSP overhead yields more available MIPS for more intensive software applications or allows the reduction in clock frequency/voltage and thereby dramatically extends the battery life. The filter coprocessor is a DMA based engine that overlaps execution with the DSP.

2.3.10. IDE/ATA Hard Drive Interface

The external memory controller interface supports the attachment of an ATA/IDE hard drive device. This is particularly useful for one inch 2GByte hard drive and 2.5 inch 10GByte hard drive MP3 players. Hard drive and external SDRAM configurations are supported in the same application, i.e. large blocks of hard drive data can be copied to SDRAM leaving the hard drive unused for most of the MP3 play time.

2.3.11. SDRAM Interface

The STMP35xx contains an SDRAM controller that can be used to connect external SDRAM memory chips. The controller is designed to work with 8 bit wide memory systems. It supports SDRAM products from the 64Mbit, 128Mbit and 256MBit JEDEC families. SDRAM memory systems as small as 8MBytes can be configured. SDRAM memory subsystem are useful for applications that include CD-ROM or IDE hard drives.

2.3.12. Serial Peripheral Control Interface and I^2C

The chip contains a four wire SPI bus. It can act as a master for this bus to control other chips in the system, such as EEPROMs. It can also act as a slave on this bus to allow a host processor to communicate with the STMP35xx. The STMP35xx includes an enhanced SPI interface that provides DMA transfer support. In addition, the chip contains a two wire SMB/I^2C bus interface. It can act as either a slave or master on the SMB interface.

2.3.13. LCD/LED and GPIO

The STMP35xx contains 85 GPIO pins in the 144-pin package. Most digital pins that are available for specific functions, e.g. SDRAM interface are also available as GPIO pins if they are not otherwise used in a particular application.

Most LCD and LED displays can be directly controlled from the GPIO interface.

2.3.14. PULSE WIDTH MODULATOR (PWM) CONTROLLER

The STMP35xx contains four PWM output controllers that can be used in place of GPIO pins. Applications include LED brightness control and high voltage generators for electroluminescent lamp (E.L.) display back lights. Independent output control of each phase allows zero, one or hi-Z to be independently selected for the active and inactive phases. Individual outputs can be run in lock step with guaranteed non-overlapping portions for differential drive applications.

The controller does not use the DMA. Initial values of Period, Active, and Inactive widths are set for each desired channel. The outputs are selected by phase and then the desired PWM channels are simultaneously enabled. This effectively launches the PWM outputs to autonomously drive their loads without further DSP intervention.

Each PWM channel has a dedicated internal 12 bit counter which increments once for each divided clock period presented from the clock divider. The internal counter resets when it reaches the value stored in the channel control registers. The Active flip flop is set to one when the internal counter reaches the value stored in a register. It remains high until the internal counter exceeds the value stored in another register. These two value define the starting and ending points for the logically *“active”* portion of the waveform. The actual state on the output for each phase, e.g. active or inactive, is completely controlled by the active and inactive state values in the channel control registers and can be: HIGH, LOW, or TRI-STATE.

2.4. Signal Pin Sharing Among Various Application Configurations

A large number of the chips I/.O pins are shared between various functions. The exact conflicts can be found in Table 489, “Pin Definition Table,” on page 368.

	EMC NAND & SmartMedia	EMC CompactFlash Memory & ATA/IDE	SDRAM	CapLess Mode Headphone	Line In 1)
EMC NAND & SmartMedia		#2	#1		
EMC CompactFlash memory & ATA/IDE	#2		#1		
SDRAM	#1	#1			
CapLess Mode Headphone					#3
Line In 1				#3	

Table 2. Pin Sharing Constraints by Subsystem

NOTES:

#1: The EMC and SDRAM interfaces share a number of pins, including address and data busses. While precluding exactly simultaneous accesses, careful attention to chip selects and controller programing allows them to be used within the same application, e.g. reading CompactFlash in IDE mode and writing SDRAM for HDD MP3 player applications. Recommendation: use driver level mutual exclusion semaphores.

#2: Within the EMC devices, conflicts can occur between shared pins in the NAND/SmartMedia interface and the CompactFlash interface. Use driver level mutual exclusion semaphores.

#3: Capless headphone mode common amplifier output shares a pin with the analog line 1 Right input. In addition, the headphone common mode sense input shares a pin with analog line 1 Left input. Only one of these uses can be designed into a specific application.

2.5. Additional Documentation

Additional documentation and information is available from SigmaTel, including an extensive software development kit (SDK), application notes, reference schematics, sample PCB board layouts, sample bill of materials, etc.

It is specifically recommended that the reader refer to the peripheral device include files from the SDK. These files provide constant declarations for address offsets to the registers defined in the following sections. Note that the name of each programmable register defined in this data sheet corresponds to a C language #define or assembly language equate of the exact same name. In addition, these files contain

declarations that allow symbolic access to the individual bit fields within these registers. User programs can include all of these peripheral include files by simply including the file hw_equ.inc into their assembly files and hw_equ.h into their C files.

3. CHARACTERISTICS/SPECIFICATIONS

3.1. Absolute Maximum Ratings

PARAMETER	MIN	MAX	UNITS
Ambient operating temperature (Note 1)	-10	70	°C
Storage temperature	-40	125	°C
Battery Pin (BATT) DCDCMODE 000, 001, 010,011	-0.3	4.2	V
Battery Pin (BATT) DCDCMODE 101,111	-0.3	1.98	V
5 Volt Source Pin (VDD5V)	-0.3	5.25	V
PSWITCH DCDCMODE 101,111	-0.3	BATT	V
PSWITCH DCDCMODE 000,001,010,011(Note 2).	-0.3	VddIO+0.6	V
Analog supply voltage (VddA1, 2, VddHP, VddPLL)	Digital Supply	1.98	V
Digital supply voltage (VddD1, 2, 3)	-0.3	Analog Supply	V
I/O Supply (VddIO1, 2, 3, 4)	-0.3	3.63	V
DCDC converter #1 (DCDC_VddD)	-0.3	VddD Rail	V
DCDC converter #1 (DCDC_VddA)	-0.3	VddA Rail	V
DCDC converter #1 (DCDC_VddIO) DCDC Mode 000	-0.3	4.2V	V
DCDC converter #1 (DCDC_VddIO) all other DCDC Modes	-0.3	3.6V	V
DCDC converter #1 (DCDC_Batt)	-0.3	max (VddIO, BATT)	V
DCDC converter #2 (DCDC2_Vout) DCDC Mode 000	-0.3	4.2	V
DCDC converter #2 (DCDC2_Vout) all other DCDC Modes	-0.3	3.6	V
Input voltage on any DCDC MODE input pin relative to ground (DCDCMOD) (Note 3)	-0.3	BATT	V
Input voltage on any digital I/0 pin relative to ground (DIO3) (Note 3)	-0.3	VDDIO+0.3	V
Input voltage on any digital I/O pin in 1.8V mode relative to ground (DIO18) (Note 3)	-0.3	VDDD+0.3	V
Input voltage on USB D+, D- pins relative to ground (USBIO) (Note 3)	-0.3	3.6	V
Input voltage on any analog pin relative to ground (AIO) (Note 3)	-0.3	VDDA+0.3	V

Table 3. Absolute Maximum Ratings

Note:
1. Contact SigmaTel for extended temperature range options. In most systems designs, battery and display specifications will limit the operating range to well within these specifications.
2. The maximum voltage limit on the PSWITCH can be achieved in DCDCMODE 000, 001, 010, and 011 by tying the switch to VddIO through a 10KΩ resistor. The ESD protection diode limits the input voltage to an acceptable level. In mode 111 and mode 101, the switch is tied to the battery.
3. Pin sets for DCDCMODE, DIO3, DIO18, and AIO are defined in the pin list, see Table 489. “Pin Definition Table” on page 368.

3.2. Recommended Operating Conditions

PARAMETER	MIN	TYP	MAX	UNITS
Digital core supply voltage – VddD1, VddD2, VddD3 (Note 4) *Specification dependent on DSP frequency*	1.25	-	Analog Supply	V
Digital I/O supply voltage – VddIO1, VddIO2	2.9	3.0	3.6	V
Analog supply voltage – VddA(VddA1, VddA2, VddA3) *Specification dependent on maximum output power*	Digital Supply	-	1.98	V
Battery startup input voltage in 1xAA or 1xAAA mode	0.9	-	-	V
Full Scale Input Voltage: Line Inputs (Note 5) Mic Input With 20 dB boost Without 20 dB boost	 - -	 0.6 0.06 0.6	 - -	 Vrms Vrms
Full Scale Output Voltage with 16Ω load: Headphone/Line Outputs (VddA = 1.8 V)	-	0.54	-	Vrms
Headphone/Line Outputs (VddA = 1.35 V)	-	0.42	-	Vrms
Crosstalk between input channels (16Ω loads at 1Khz)	-	-75	-	dB
THD+N (16Ω headphone at 1 Khz)	-	-70	-66	dB
THD+N (10KΩ load at 1 Khz)		-87		
Analog line input resistance (Note 6)	-	25	-	kΩ
Microphone input resistance	-	100	-	kΩ
Analog output resistance	-	-	<1	Ω
DAC SNR Idle Channel (Note 7)	-	96	-	dB
DAC -60dB dynamic range (Note 7)	92	94	-	dB
ADC SNR Idle Channel (Note 7)		90		dB
ADC -60dB dynamic range (Note 7)		90		dB
Line SNR (Note 7)	92	94	-	dB
Standby Current (Note 8)		TBD	TBD	uA

Table 4. Recommended Operating Conditions

Note:
4. Recommended operating voltages for DCLK can be found in Table 5.
5. At 1.35VddA max input is 0.45Vrms
6. Input resistance changes with volume setting: 10KΩ at +12dB, 25KΩ at 0dB, 50KΩ at -34.5dB
7. Measured “A weighted” over a 20 Hz to a 20 kHz bandwidth, relative to full scale output voltage (1.8V)
8. The chip consumes current when in the *“OFF”* mode to keep the crystal oscillator and the real time clock running. With a typical 2850mAh AA battery, the standby current would take more than 1 year to drain the battery fully. It also is possible to design a system that disables the crystal oscillator and real time clock to achieve a much lower OFF current.

The following table can be used to select a proper setting for VddD and VddD brownout voltages based on standard analysis of worst case design and characterization data.

MAX DCLK TARGET	Min. VddD	HW_DCDC_VDDD_ VOLTAGE_LEVEL	Corresponding VddD Brownout Voltage	HW_DCDC_VDDD_ BROWNOUT_LEVEL
75MHz	TBD V	TBD	TBD V	TBD
70MHz	TBD V	TBD	TBD V	TBD
65MHz	TBD V	TBD	TBD V	TBD
60MHz	TBD V	TBD	TBD V	TBD
55MHz	TBD V	TBD	TBD V	TBD
50MHz	TBD V	TBD	TBD V	TBD
45MHz	TBD V	TBD	TBD V	TBD
40MHz	TBD V	TBD	TBD V	TBD
35MHz	TBD V	TBD	TBD V	TBD
30MHz	TBD V	TBD	TBD V	TBD
25MHz	TBD V	TBD	TBD V	TBD
20MHz	TBD V	TBD	TBD V	TBD
15MHz	TBD V	TBD	TBD V	TBD
10MHz	TBD V	TBD	TBD V	TBD
5MHz	TBD V	TBD	TBD V	TBD
200KHz	TBD V	TBD	TBD V	TBD

Table 5. Recommended Operating Conditions for specific dclk targets

After split lot characterization of the part performance versus speed sensor values, a closed loop method for setting VddD voltage and brownout levels will be provided which allows VddD settings to be tuned to the actual process corner of a part, at the then current ambient temperature and voltage.

Preliminary

3.3. DC Characteristics

PARAMETER	MIN	TYP	MAX	UNITS
Power Dissipation, VddD = TBD V, VddA = TBD V, VddIO = TBD V DCDC mode = TBD, DCLK = TBD MHz on PLL, USB off, Application = MP3 Play, minimum power configuration selected.		TBD		mW
V_{iH} (DIO3) - Input high voltage for DIO3 digital I/O pin set, in 3.3 Volt mode.	2.0			V
V_{IL} (DIO3) - Input low voltage for DIO3 digital I/O pin set in 3.3 Volt mode.			0.8	V
V_{IH} (DIO18) - Input high voltage for DIO18 digital I/O pin set in 1.8 Volt mode.	0.7*VddD			V
V_{IL} (DIO18) - Input low voltage for DIO18 digital I/O pin set in 1.8 Volt mode.			0.3*VddD	V
V_{OH} (DIO3) - Output high voltage for DIO3 digital I/O pin set in 3.3 Volt mode, 4mA mode.	0.7VddIO			V
V_{OH} (DIO3) - Output high voltage for DIO3 digital I/O pin set in 3.3 Volt mode, 8mA mode.	TBD			V
V_{OL} (DIO3) - Output low voltage for DIO3 digital I/O pin set in 3.3 Volt mode.			0.4	V
V_{OH} (DIO18) - Output high voltage for DIO18 digital I/O pin set in 1.8 Volt mode.	VddD - 0.4			V
V_{OL} (DIO18) - Output low voltage for DIO18 digital I/O pin set in 1.8 Volt mode.			0.4	V

Table 6. DC Characteristics

Preliminary

4. DSP CORE

The on-chip DSP core is modeled exactly after the Motorola DSP56004. It supports the identical instruction set, registers, addressing modes, etc., as the DSP56000 family of digital signal processors. Figure 13 shows the DSP architecture.The DSP core is a general-purpose 24-bit DSP especially suited to high fidelity digital audio applications for very low power/energy environments.

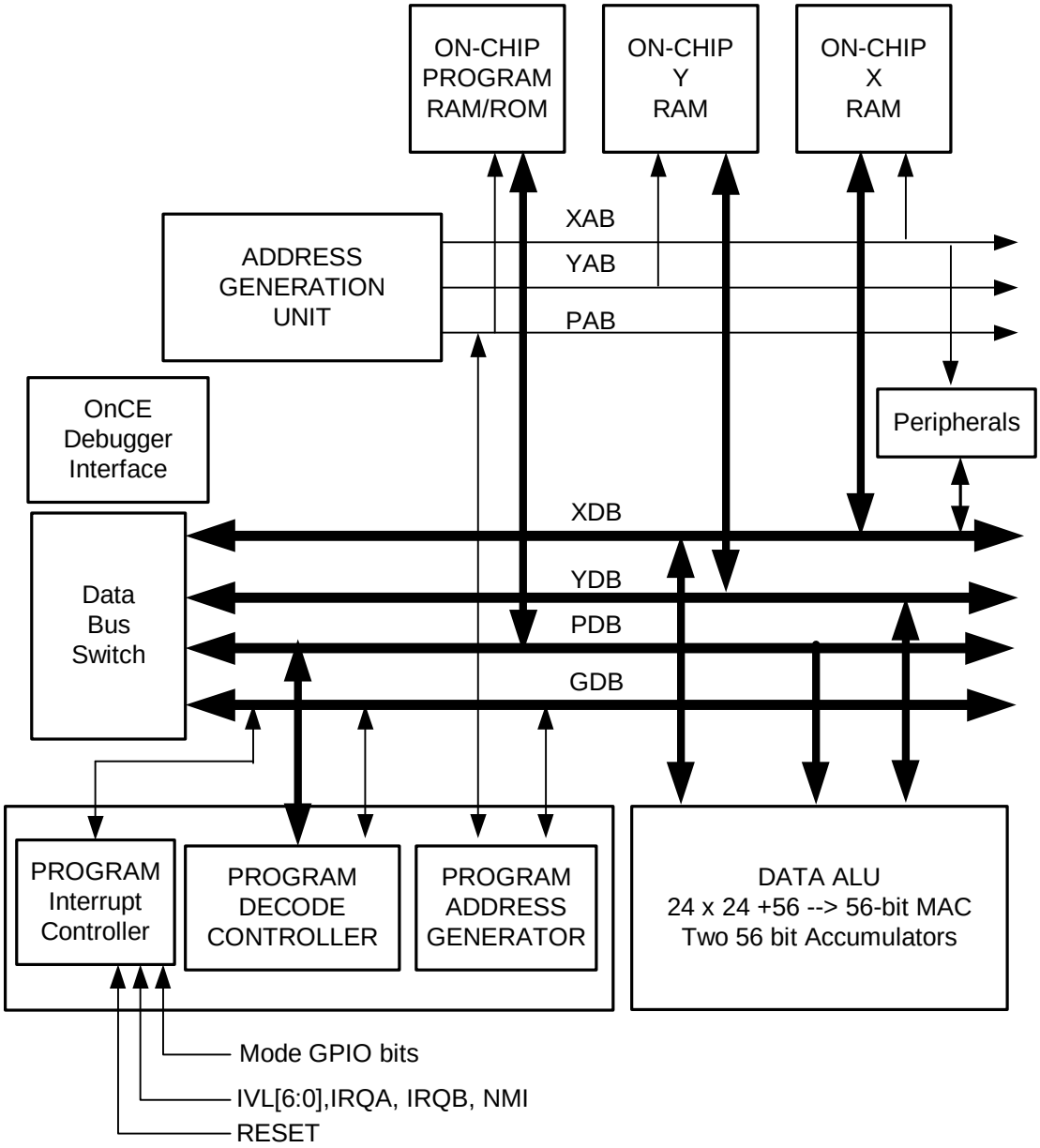


Figure 13. DSP Architecture

The central components are:

- Data Busses (XD,YD,PD)
- Address Busses (XA,YA,PA)
- Data Arithmetic Logic Unit (Data ALU)
- Address Generation Unit (AGU)
- Program Control Unit (PCU)
- On-Chip Program ROM
- On-Chip X,Y,P RAM
- On-Chip Emulation circuitry

The DSP is organized around the registers of three independent execution units: the PCU, the AGU and the data ALU. Data movement between the execution units occurs over four bidirectional 24-bit busses: the X data bus (XDB), the Y data bus (YDB), the program data bus (PDB) and the global data bus (GDB). Certain instructions treat the X and Y data buses as one 48-bit data by concatenating them. Data transfers between the data ALU and the X data memory or Y data memory occur over the XDB and YDB respectively.

The bus structure supports general register-to-register, register-to-memory, and memory-to-register data movement. It can transfer up to two 24-bit words or one 56-bit word in the same instruction cycle.

Transfers between busses occur in the internal bus switch. The internal bus switch, which is similar to a switch matrix, can connect any two internal busses without adding pipeline delays. Thus greatly simplifying the programming model.

The bit manipulation unit is located in the bus switch so that it can access each memory space. The bit manipulation unit performs bit operations on memory locations, address registers, control registers and data registers over the XDB, YDB and GDB.

The data ALU performs all of the arithmetic and logical operations on data operands. It consists of four 24 bit input registers, two 48-bit accumulators and two 8-bit accumulator extension registers, an accumulator shifter, two data bus shifter/limiter circuits, and a parallel single cycle, non-pipelined multiply-accumulator (MAC) unit.

The address generation unit (AGU) performs all of the address storage and address generation computations necessary to indirectly address data operands in memory. It operates in parallel with other DSP resources to minimize address generation overhead and keep the supply of data operands fed to the data ALU. The AGU has two identical address arithmetic units that can generate two 16-bit addresses every instruction cycle. Each of the arithmetic units can perform one of three types of address arithmetic: linear, modulo and reverse-carry.

The program control unit performs instruction prefetch, instruction decode, hardware DO loop control, and interrupt/exception processing. It consists of three components: the program address generator, the program decode controller, and the program interrupt controller. It contains a 15-level by 32 bit system stack memory and the following directly addressable registers: the program counter (PC), loop address (LA), loop counter (LC), status register (SR), and the operating mode register (OMR).

The DSP core responds to 7 interrupt vector level inputs (IVL[6:0], two peripheral interrupts (IRQA, IRQB) and a non-maskable interrupt (NMI).

4.1. Operating Mode Register

The organization of the operating mode register is shown below. The operating mode register determines chip configuration including boot modes, and memory configuration. The HW_OMR is a core register that is accessible by special DSP instructions. It therefore has no address.

HW_OMR SPECIAL

BITS	LABEL	RW	RESET	DEFINITION
23:8	**RSRVD**	R	0	**Reserved –** Must be written with 0.
7	**MDLP**	RW	0	**Low Power Mode –** Writing a 0 to this bit enables some clock gating in the DSP core that reduces its power consumption. This register must be written with a 1 for proper operation of the debug functionality in the DSP.
6	**SD**	RW	0	**Stop Delay –** This bit is exported from the core as an output. It can be used when waking up from the STOP low power standby mode. If this bit is set, then when an IRQA interrupt occurs to wake up the core from the STOP state, the clock control circuitry will wait a time period (e.g. 65536 clock cycles) before allowing the clocks back in to the DSP core. This can be used, for example, to restabilize a PLL clock oscillator. If this bit has a zero value, then the clocks will be allowed back into the core immediately after the occurrence of the IRQA interrupt, thus implementing a “warm boot” from the STOP low power standby state.
5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4	**MC**	RW	GP0	**Operating Mode C –** This bit is used to configure the boot mode for the STMP35xx. When the hardware reset is active, this bit samples the state of GP0 pin. Once the boot code executes, it can check the state of this bit in order to make decisions about what type of boot mode to perform.
3	**YD**	RW	0	**Internal Y-Memory Disable –** This bit has no effect.
2	**DE**	RW	1	**Data ROM Enable –** This bit has no effect.
1	**MB**	RW	GP1	**Operating Mode B –** This bit is used to configure the boot mode for theSTMP35xx. When the hardware reset is active, this bit samples the state of the GP1 pin. Once the boot code executes, it can check the state of this bit in order to make decisions about what type of boot mode to perform.
0	**MA**	RW	1	**Operating Mode A –** This bit is used to choose between Boot ROM and Program Memory for instruction fetches and read accesses. When this bit is set, as it is after hardware reset, the Boot ROM space is activated and any fetches or read accesses to the P: space will refer to the on-chip ROM. When this bit is a zero, the Program RAM Memory space is enabled instead of the Boot ROM memory space and any fetch or read access to P: space will refer to the on-chip RAM. Writes to P: space always access the program RAM regardless of the state of the MA bit. It is not possible to write to the program ROM. This bit affects ROM access in region P:$0000 through P:$3FFF. Accesses to the high mapped region of ROM, P:$C000 through P:$FFFF is controlled by **HW_RAM_ROM_CFG_PROMIE**.

Table 7. Operating Mode Register Description

4.2. General Debug Register

The HW_GDBR Register is also mapped into the X Peripheral I/O space. This register is used as a gateway between the DSP and the Debug port. For instance, when displaying the states of the internal registers and memory of the DSP core, the DSP moves the data to this register and the data is then shifted out the OnCE_DSO pin. The HW_GDBR register operation is controlled automatically by the emulator and the debug circuitry within the core. An added feature of the Debug Unit is that the

emulator cannot access the debug unit unless a write to the HW_GDBR Register is executed by the DSP (normally in the boot code).

HW_GDBR X:$FFFC

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
VALUE																							

Table 8. HW_R

BITS	LABEL	RW	RESET	DEFINITION
23:0	**VALUE**	RW	0	Value to be read by the debugger or debugger value to be read by a DSP instruction from the debugger.

Table 9. General Debug Register (GDBR)

4.3. OnCE (On-Chip Emulator) Debug Interface

The DSP on-chip emulation (OnCE) circuitry provides a sophisticated debugging tool that allows simple, inexpensive, and speed independent access to the processor's internal registers, memories and peripherals. OnCE provides software engineers with access to the internal state including the addresses of the last five instructions and provides the ability to modify that state, and single step the processor. OnCE capabilities are accessed through a four pin interface

1. Debug Serial input (OnCE_DSI)
2. Debug Serial Clock (OnCE_DSK)
3. Debug Serial Output (OnCE_DSO)
4. Debug Request Input (OnCE_DRN)

The OnCE controller and serial interface consists of the following blocks: OnCE command register, bit counter, OnCE decoder and the status/control register. For a block diagram, see Figure 14. “OnCE Interface Block Diagram” on page 29.

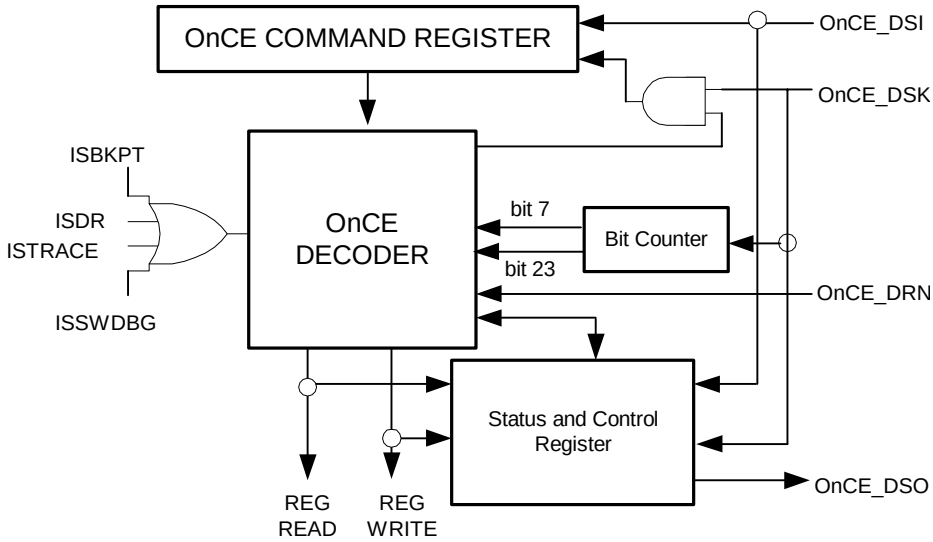


Figure 14. OnCE Interface Block Diagram

The OnCE Command Register is an 8 bit shift register that receives the serial data from the OnCE_DSI pin. It holds the 8 bit commands to be used as input for the OnCE controller.

OnCE Command Register

BITS	LABEL	RW	RESET	DEFINITION
7	**RW**	RW		The read/write bit specifies the direction of the data transfer. For zero, write the data associated with the command into the register specified in the **RS** field. For one, read the data contained in the register specified in the **RS** field.
6	**GO**	RW		If the GO bit is set, the chip will execute the instruction which resides in the PIL register. To execute the instruction, the processor leaves the debug mode, and the status is reflected on the OS0,OS1 pins. The processor will return to the debug mode immediately after executing the instruction.
5	**EX**	RW		If the Exit Command bit is set, the processor will leave the debug mode and resume normal operation. The Exit command is executed only if the Go command was issued and the operation is a write to OPDBR or a read/write to" No Register Selected".
4:0	**RS**	RW		Register Select field

Table 10. OnCE Command Register

The Register Select field (RS[4:0]) selects one of 32 OnCE debug registers to be read or written.

RS[4:0]	REGISTER SELECTED
00000	OnCE Status and Control Register
00001	Memory Breakpoint register
00010	Reserved
00011	Trace Counter
0010X	Reserved
00110	Memory Upper Limit
00111	Memory Lower Limit

Table 11. OnCE Register Selects

RS[4:0]	REGISTER SELECTED
01000	GDB Register NOTE: this register can be read or written by the DSP instructions
01001	PDB Register
01010	PAB Register for Fetch
01011	PIL register, next instruction from debugger comes from here
01100	Clear Memory Breakpoint Counter
01101	Reserved
01110	Clear Trace Counter
01111	Reserved
10000	Reserved
10001	Program Address Bus FIFO and Increment Counter
10010	Reserved
10011	PAB Register for Decode
101XX	Reserved
11XX0	Reserved
11111	No Register Selected

Table 11. OnCE Register Selects

5. ON-CHIP MEMORY SUBSYSTEM

The chip includes 96 kwords of on-chip RAM (96k x 24 bits = 2.25 Mbits) that is used for program and data storage, and 16K words of on-chip ROM (16K x 24 bits = 384kbits) that is used for the code that boots the device (see Section 20 for more details on boot modes and the contents of the on-chip ROM). The on-chip ROM is mapped at the address range P:$0000-$3FFF at reset, it can also be configured to be mapped at address range P:$CFFF-$FFFF, or it can be disabled.

The on-chip RAM is organized into two banks of 48K words each, called PXRAM and PYRAM. PXRAM can be mapped into the DSP P memory space, starting at P:$0000, or into the DSP X memory space, starting at X:$0000. PYRAM can be mapped into the DSP P memory space, starting immediately after the end of the PXRAM memory, or into the DSP Y memory space, starting at Y:$0000. Both PXRAM and PYRAM memory can be allocated to the DSP P, X or Y memory spaces in 8K word increments, from a minimum of 0K words to all available memory. The memory configuration is controlled by the PX & PY Memory Configuration registers documented below. There are no hardware safeguards against improper programming of these registers. It is possible to allocate less than all of the on-chip RAM, unallocated memory will then be invisible to the DSP.

5.1. PXRAM Configuration Register

HW_PXCFG X:$FFE8

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
										PXXSIZE								PXPSIZE					

Table 12. HW_PXCFG

BITS	LABEL	RW	RESET	DEFINITION
23:14	**RSRVD**	R	0	**Reserved –** Must be written with 0.
13:8	**PXXSIZE**	RW	011000	Number of kwords of PXRAM that is mapped in the DSP X memory space. Initialize 24KW to X RAM. This six bit field represents 1K increments of RAM. It must be allocated in 8K word chunks.
7:6	**RSRVD**	R	0	**Reserved –** Must be written with 0.
5:0	**PXPSIZE**	RW	011000	Number of kwords of PXRAM that is mapped in the DSP P memory space. Initialize 24KW to Y RAM. This six bit field represents 1K increments of RAM. It must be allocated in 8K word chunks.

Table 13. PXRAM Configuration Register Description

5.2. PYRAM Configuration Register

HW_PYCFG X:$FFE9

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
										PYYSIZE								PYPSIZE					

Table 14. HW_PYCFG

BITS	LABEL	RW	RESET	DEFINITION
23:14	**RSRVD**	R	0	**Reserved –** Must be written with 0.
13:8	**PYYSIZE**	RW	011000	Number of kwords of PYRAM that is mapped in the DSP Y memory space. Initialized to 24KW Y RAM.This six bit field represents 1K increments of RAM. It must be allocated in 8K word chunks.
7:6	**RSRVD**	R	0	**Reserved –** Must be written with 0.
5:0	**PYPSIZE**	RW	011000	Number of kwords of PYRAM that is mapped in the DSP P memory space. Initialized to 24KW to P RAM. This six bit field represents 1K increments of RAM. It must be allocated in 8K word chunks.

Table 15. PYRAM Configuration Register Description

The PXRAM bank is accessible to DSP P space accesses, DSP X space accesses, and DMA accesses. The PYRAM bank is accessible to DSP P space accesses, DSP Y space accesses, and DMA accesses. PXRAM & PYRAM are made up of 3 physical blocks of 16K words each for a total of 48K words each. Since the allocation of on-chip RAM to DSP P space, X space, & Y space is in 8K Word increments, it is possible for a single physical memory block in the PXRAM bank to be accessed by the DSP P, DSP X, and DMA busses at the same time, similarly for the PYRAM bank. When this happens, the memory interface control logic steals one or more clock cycles from the DSP to allow all of the accesses to complete in separate clock cycles. In case of conflict, DMA accesses take priority over DSP accesses, and DSP program accesses take priority over DSP data accesses.

It is possible to eliminate cycle steals because of DSP P space and DSP X or Y conflicts by allocating PXRAM & PYRAM in increments of 16K words, this means that each physical memory block is allocated to DSP P, X or Y space. It is not possible to eliminate the cycle steal that happens because a DMA access conflicts with a DSP access to memory. In order to allow the programmer to monitor total number of cycle steals are happening, the contents of the Cycle Steal Count Register (HW_CYCSTLCNT) will increment whenever a cycle is stolen for a memory access conflict.

Figure 15 below shows an example of how the PXRAM and PYRAM memory banks can be allocated. In this example, PXRAM is allocated as 24K words P, &

24K words X, and PYRAM is allocated as 24K words P, & 24K words Y, for a total of 48K words of P, 24K words of X, and 24K words of Y.

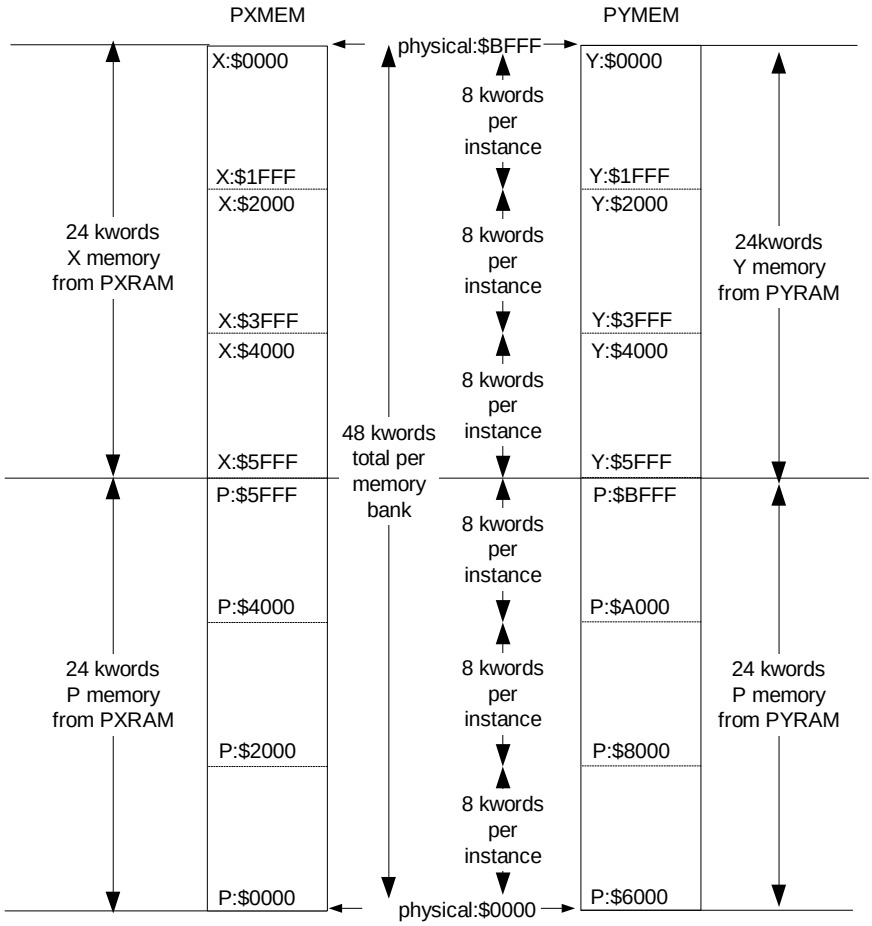


Figure 15. On-Chip RAM Organization

When the memory is configured as shown in Figure 15, the DSP's view of the memory map will be as shown below in Figure 16. The DSP P address space has 48K words of RAM as shown in the range P:$0000 to P:$BFFF. If the ROM is enabled then it appears in the DSP P address space at P:$C000 to P:$FFFF. The DSP X space has 24K words of RAM at X:$0000 to X:$5FFF, likewise the DSP Y space has 24K words of RAM at Y:$0000 to Y:$5FFF. The address ranges X:$6000 to X:$EFFF, and Y:$6000 to Y:$FFFF are not populated with any memory. The address range P:$C000 to P:$FFFF is unpopulated if the ROM is not enabled. The address range X:$F000 to X:$FFFF is reserved for on-chip peripherals.

In the above example memory configuration, it is possible for cycle steals to happen because of either DSP access conflicts, or because of DMA access conflicts. The areas in the memory map where DSP access conflicts can occur are cross hatched in Figure 16. If the DSP attempts to access an address in the range P:$4000 to P:$5FFF at the same time as an address in the range X:$4000 to X:$5FFF, a stall cycle will occur. If the DSP attempts to access an address in the range P:$A000 to

P:$BFFF at the same time as an address in the range Y:$4000 to Y:$5FFF, a stall cycle will also occur.

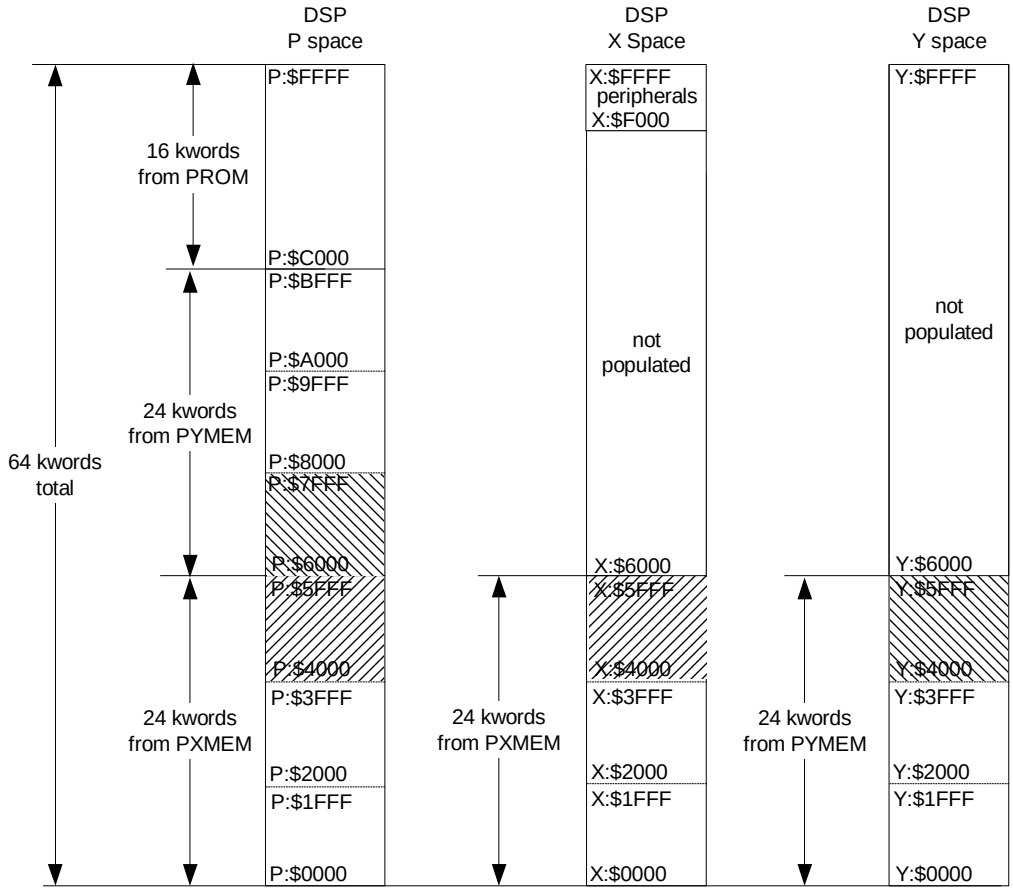


Figure 16. DSP Memory MAP

It is possible to reallocate on-chip RAM at any time, however doing so is dangerous and should be done carefully. In particular, the DSP P space memory that is allocated from the PYMEM memory bank ***will move within the P address space*** if the allocation of PXRAM memory is changed.

5.3. On-Chip ROM

When the chip comes out of reset, the OMR MODE A bit is set to one, see 4.1. “Operating Mode Register” on page 27. This bit, when set to one, enables the On-Chip ROM to be read instead of whatever PXRAM or PYRAM is assigned to the first 16KW of the P address space, i.e. P:$0000 through P:$3FFF. For the default PXRAM/PYRAM allocation described above, the cross hatched region of Figure 17 shows the 16KW of PXRAM space that are overlaid by ROM. Whenever a P space access is made in the range P:$0000 through P:$3FFF, ***AND*** OMR_MODE_A =1 ***AND*** a read cycle is performed, then ROM data is presented to the DSP P-bus. If mode A is not set then reads in this range access the allocated PXRAM data. Whenever a P-bus write cycle is performed within this address range it is always directed

to PXRAM. Thus at power on time, Instructions and P-space read data are read from the On-Chip ROM while P-space write data is written to the PXRAM.

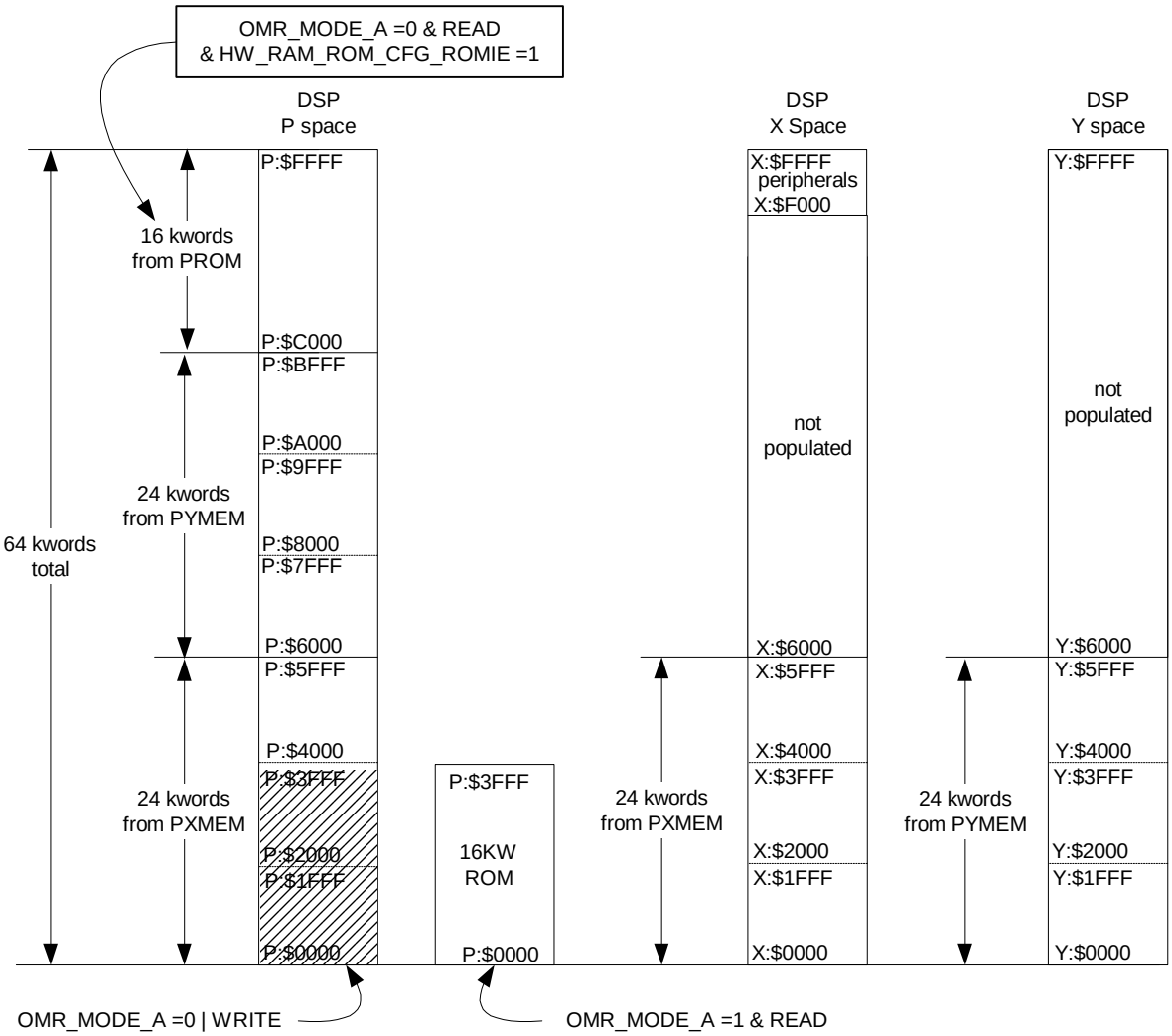


Figure 17. DSP Memory MAP with On-Chip ROM

The boot strap code, which begins execution in the on-chip ROM takes advantage of these mode settings as follows: The boot code reads the new program code in from the NAND Flash, or other source. It copies the data, as described in the boot mode section below, to the appropriate memory locations. If a block is copied to P-space in the range P:$0000 through P:$3FFF then it is written to PXRAM as if the ROM were not there. When the boot load is complete, the DSP software must turn off the ROM and begin execution of the loaded code. All code loaded by the ROM Boot begins execution at location P:$0000 after the ROM is turned off. The ROM boot loader calls a routine to copy the program and branch to P:$0000, as follow:

```
jsr (R5)        ; call the boot copy program
bclr  #0,HW_OMR ; turn off the ROM
jmp $0          ; begin execution at the first location in RAM
```

Recall that the DSP is pipelined and that the reset of the OMR bit happens when the bclr instruction reaches the execute stage. By that time, the jmp $0 has already been fetched into the pipeline and will fetch a RAM location instead of a ROM location for its next instruction.

5.4. On-Chip Memory Configuration Register

HW_RAM_ROM_CFG X:$FFED

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		PYRAM_CLK_EN	PXRAM_CLK_EN	ROM_CLK_EN	PROMIE		RAMAWT	RAMRM				PROMCT				PYRAMCT				PXRAMCT			

Table 16. HW_RAM_ROM_CFG

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**PYRAM_CLK_EN**	RW	1	**PYRAM Clock Enable –** This bit enables or disables the clock to the PYRAM array. This clock must be turned on for normal operation. 0 Disable clock for the PYRAM array 1 Enable clock for the PYRAM array
20	**PXRAM_CLK_EN**	RW	1	**PXRAM Clock Enable –** This bit enables or disables the clock to the PXRAM array. This clock must be turned on for normal operation. 0 Disable clock for the PXRAM array 1 Enable clock for the PXRAM array
19	**ROM_CLK_EN**	RW	1	**PROM Clock Enable –** This bit enables or disables the clock to the on-chip ROM. This clock should be turned off to save power if the ROM is not being used. It must be turned on for the correct operation of the on-chip ROM. 0 Disable clock for the on-chip ROM 1 Enable clock for the on-chip ROM
18	**PROMIE**	RW	1	**PROM Image Enable –** After reset, the on-chip ROM is located at the address range P:$0000-$3FFF. Once the bootloader in the ROM clears the MODEA bit in the Operating Mode Register (HW_OMR), this address range reverts to on-chip RAM. This mode bit enables the contents of the ROM to be viewed at the address range P:$C000-$FFFF, irrespective of the state of the MODEA bit of the Operating Mode Register. Since this address range can also be used by on-chip RAM, it is necessary to be able to disable the ROM in this address range. 0 Disable P:$C000-$FFFF image for on-chip ROM 1 Enable P:$C000-$FFFF image for on-chip ROM NOTE: This bit does not affect ROM accesses in the lower region of P: space between P:$0000 and P:3FFF which is controlled solely by **HW_OMR_MA**.
17	**RSRVD**	R	0	**Reserved –** Must be written with 0.
16	**RAMAWT**	RW	0	**RAM AWT Mode –** This register is used to select the asynchronous write through (AWT) mode for the on-chip RAM. This is a test mode that should not be used in normal operation. 0 Disable AWT mode 1 Enable AWT mode

Table 17. On-Chip Memory Configuration Register Description

BITS	LABEL	RW	RESET	DEFINITION
15:12	**RAMRM**	RW	0000	**RAM Read Margin –** This register is used to optimize the read performance of all of the on-chip RAM. The optimum value of this register will be determined by SigmaTel, and should not be modified.
11:8	**PROMCT**	RW	1000	**PROM clock tune –** This register is used to optimize the placement of the clock to the on-chip ROM. The optimum value of this register will be determined by SigmaTel, and should not be modified.
7:4	**PYRAMCT**	RW	1000	**PYRAM Clock Tune –** This register is used to optimize the placement of the clock to the PYRAM block. The optimum value of this register will be determined by SigmaTel, and should not be modified.
3:0	**PXRAMCT**	RW	1000	**PXRAM Clock Tune –** This register is used to optimize the placement of the clock to the PXRAM block. The optimum value of this register will be determined by SigmaTel, and should not be modified.

Table 17. On-Chip Memory Configuration Register Description (Continued)

5.5. PXRAM0 Repair Register

HW_PXRAM0_RPR X:$F5A3

BITS	LABEL	RW	RESET	DEFINITION
23:0	**REPAIR**	RW	$000000	Repair register, each bit controls a 2:1 mux on the input and output of the PXRAM0 array. If the bit is set to one, then the corresponding bit in the array is substituted by the next higher significant bit in the array. When set to zero, there is a 1:1 correspondence between the control bit position and the array bit position. This register controls the data bus for the 16K Word SRAM bank covering X:$8000 through X:$BFFF.

Table 18. PXRAM0 Repair Register Description

5.6. PXRAM1 Repair Register

HW_PXRAM1_RPR X:$F5A4

BITS	LABEL	RW	RESET	DEFINITION
23:0	**REPAIR**	RW	$000000	Repair register, each bit controls a 2:1 mux on the input and output of the PXRAM1 array. If the bit is set to one, then the corresponding bit in the array is substituted by the next higher significant bit in the array. When set to zero, there is a 1:1 correspondence between the control bit position and the array bit position. This register controls the data bus for the 16K Word SRAM bank covering X:$4000 through X:$7FFF.

Table 19. PXRAM1 Repair Register Description

5.7. PXRAM2 Repair Register

HW_PXRAM2_RPR X:$F5A5

BITS	LABEL	RW	RESET	DEFINITION
23:0	**REPAIR**	RW	$000000	Repair register, each bit controls a 2:1 mux on the input and output of the PXRAM2 array. If the bit is set to one, then the corresponding bit in the array is substituted by the next higher significant bit in the array. When set to zero, there is a 1:1 correspondence between the control bit position and the array bit position. This register controls the data bus for the 16K Word SRAM bank covering X:$0000 through X:$3FFF.

Table 20. PXRAM2 Repair Register Description

5.8. PYRAM0 Repair Register

HW_PYRAM0_RPR X:$F5A6

BITS	LABEL	RW	RESET	DEFINITION
23:0	**REPAIR**	RW	$000000	Repair register, each bit controls a 2:1 mux on the input and output of the PYRAM0 array. If the bit is set to one, then the corresponding bit in the array is substituted by the next higher significant bit in the array. When set to zero, there is a 1:1 correspondence between the control bit position and the array bit position. This register controls the data bus for the 16K Word SRAM bank covering Y:$8000 through Y:$BFFF.

Table 21. PYRAM0 Repair Register Description

5.9. PYRAM1 Repair Register

HW_PYRAM1_RPR X:$F5A7

BITS	LABEL	RW	RESET	DEFINITION
23:0	**REPAIR**	RW	$000000	Repair register, each bit controls a 2:1 mux on the input and output of the PYRAM1 array. If the bit is set to one, then the corresponding bit in the array is substituted by the next higher significant bit in the array. When set to zero, there is a 1:1 correspondence between the control bit position and the array bit position. This register controls the data bus for the 16K Word SRAM bank covering Y:$4000 through Y:$7FFF.

Table 22. PYRAM1 Repair Register Description

5.10. PYRAM2 Repair Register

HW_PYRAM2_RPR X:$F5A8

BITS	LABEL	RW	RESET	DEFINITION
23:0	**REPAIR**	RW	$000000	Repair register, each bit controls a 2:1 mux on the input and output of the PYRAM0 array. If the bit is set to one, then the corresponding bit in the array is substituted by the next higher significant bit in the array. When set to zero, there is a 1:1 correspondence between the control bit position and the array bit position. This register controls the data bus for the 16K Word SRAM bank covering Y:$0000 through Y:$3FFF.

Table 23. PYRAM2 Repair Register Description

6. CHIP WIDE PROGRAMMABLE CONTROL REGISTERS

6.1. Revision Register

The Revision Register reports the Device ID and revision to the software. In addition, it shows the strap state of the DCDC_MODE[2:0] bits. This register is read only. The organization of the Revision Register is shown below.

HW_REVR X:$FA02

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RMJ															RMP	DCDCMODE				RMN			

Table 24. HW_REVR

BITS	LABEL	RW	RESET	DEFINITION
23:8	**RMJ**	R	$3500	**Revision Major ID –** This is the device part number in binary coded decimal: STMP35xx $3500
8	**RMP**	R	depends on DCDC mode pin strapping	**DCDC mode/package type** 0 DCDC is configured in one of the modes that are only supported in a 144-pin package, therefore this device must be in a 144-pin package. 1 DCDC is configured one of the modes that are supported in both 100-pin and 144-pin packages, therefore no inference can be made about whether the current package is a 144-pin or 100-pin package.
7:5	**DCDCMODE**	R	depends on DCDC mode pin strapping	DCDC MODE[2:0] pin state, see 30. “DC-DC CONVERTER” on page 332.
4:0	**RMN**	R	depends on silicon revision	**Revision Minor ID –** Device revision number $00 TA1 revision

Table 25. Revision Register Description

6.2. Reset Control Register

The organization of the Reset Control Register is shown below.

HW_RCR X:$FA01

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SOVFL	SUNFL	IRQB2NMI	SOVFLEN	SOVFLLVL				SUNFLEN	SUNFLLVL				NMI	IRQB	IRQA	SRST				STKLVL			

Table 26. HW_RCR

BITS	LABEL	RW	RESET	DEFINITION
23	**SOVFL**	R	0	**Stack Overflow Status Bit –** This bit indicates that the current stack depth is equal to or greater than the value of the SOVFLLVL Stack Overflow Interrupt Level field. This is set or cleared independently of the SOVLLEN Stack Overflow Interrupt Enable field. In other words, once set this bit can only be cleared by fixing the stack underflow event by adding words to the stack.
22	**SUNFL**	R	0	**Stack Underflow Status Bit –** This bit indicates that the current stack depth is equal to or greater than the value of the SUNFLLVL Stack Under Interrupt Level field. This is set or cleared independently of the SUNFLEN Stack Underflow Interrupt Enable field. In other words, once set this bit can only be cleared by fixing the stack underflow event by adding words to the stack.
21	**IRQB2NMI**	RW	0	**Redirect battery+VddD+VddIO Brownout Interrupt to NMI** 0 battery+VddD+VddIO brownout interrupt on IRQB only 1 battery+VddD+VddIO brownout interrupt on IRQB & NMI (see brownout Figure 132 for details)
20	**SOVFLEN**	RW	0	**Stack Overflow Interrupt Enable** 0 Stack overflow interrupt disabled 1 Stack overflow interrupt enabled
19:16	**SOVFLLVL**	RW	1111	**Stack Overflow Interrupt Level**
15	**SUNFLEN**	RW	0	**Stack Underflow Interrupt enable** 0 Stack underflow interrupt disabled 1 Stack underflow interrupt enabled
14:11	**SUNFLLVL**	RW	0000	**Stack Underflow Interrupt Level**
10	**NMI**	R	1	**NMI Interrupt –** An NMI interrupt will be generated by a stack over-/underflow event. If the IRQB2NMI control bit above is set, an NMI interrupt will also be generated when a brownout event is detected. This bit will be cleared by hardware when a DSP hardware stack over- or underflow is detected. A falling edge on this bit causes an NMI interrupt in the DSP. 0 Stack over-/underflow (or brownout if the IRQB2NMI bit is set) detected 1 No Stack over-/underflow (or brownout if the IRQB2NMI bit is set) detected
9	**IRQB**	R	1	**IRQB Interrupt –** An IRQB interrupt will be generated when a brownout event is detected. This bit will be cleared by hardware to indicate that a brownout event has been detected. 0 Brownout detected 1 No brownout detected

Table 27. Reset Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
8	**IRQA**	R	1	**IRQA Interrupt –** An IRQA interrupt will be generated when a headphone short is detected. This bit will be cleared by hardware to indicate that a headphone short event has been detected. 0 Headphone short detected 1 No headphone short detected
7:4	**SRST**	RW	0000	**Software Reset –** Writing 1101 will cause a full chip hardware reset, writing any other value will have no effect.
3:0	**STKLVL**	R	0000	**Stack Level –** The current position of the DSP hardware stack.

Table 27. Reset Control Register Description (Continued)

Note that when the IRQB2NMI bit is set, the ISR that services the NMI will need to do a few special things. When this bit is set, if both interrupts happen at the same time, only one NMI will be received. Also, if one interrupt occurs while the other is being serviced, a new NMI interrupt will not be triggered. The NMI routine will need to check each possible NMI event and take appropriate action to clear the interrupt event, and then recheck that all NMI interrupt sources are clear before returning. If the NMI ISR returns with a NMI interrupt event still pending, that event (and all subsequent NMI interrupts) will be lost. Finally, the NMI ISR needs to be able to deal with the case where no interrupt event is pending when the ISR checks it, since the latency of the NMI interrupt is several cycles long, and the stack over-/underflow could have been cleared by the time that the NMI runs. In fact, this is guaranteed to happen in the case of the stack underflow interrupt, as the very fact of calling the NMI will push items onto the stack.

6.3. Hardware Profiling Support

Two clock counters are implemented to allow accurate code profiling and cycle counting. The HW_DCLKCNTL & HW_DCLKCNTU registers together form a 48 bit counter that will increment once per clock cycle. There is no hardware support for reading and writing these registers atomically. It is possible to read the lower register before it wraps around, and then read the upper register after it has been wrapped around, and get an erroneous value. Various software techniques can be used to avoid this problem. The HW_DCLKCNTL & HW_DCLKCNTU registers will increment every clock cycle, including cycles that the DSP doesn't see because they are stolen for DMA accesses or because of an access conflict in the on-chip RAMs. The HW_CYCSTLCNT register counts these cycles, and if you want to calculate the number of cycles actually seen by the DSP during a period of time, you must subtract the number of stolen clock cycles by referring to this register.

6.3.1. DCLK Count Lower Register

HW_DCLKCNTL X:$FFEA

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
HW_DCLKCNTL																							

Table 28. HW_DCLKCNTL

BITS	LABEL	RW	RESET	DEFINITION
23:0	**HW_DCLKCNTL**	RW	0	**DCLK Counter Lower –** This counter will increment once per DSP clock cycle.

Table 29. DCLK Count Lower Register Description

6.3.2. DCLK Count Upper Register

HW_DCLKCNTU X:$FFEB

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
HW_DCLKCNTU																							

Table 30. HW_DCLKCNTU

BITS	LABEL	RW	RESET	DEFINITION
23:0	**HW_DCLKCNTU**	RW	0	**DCLK Counter Upper –** This counter will increment every time that the HW_DCLKCNTL counter overflows.

Table 31. DCLK Count Upper Register Description

6.3.3. *Cycle Steal Count Register*

HW_CYCSTLCNT X:$FFEC

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
HW_CYCSTLCNT																							

Table 32. HW_CYCSTLCNT

BITS	LABEL	RW	RESET	DEFINITION
23:0	**HW_CYCSTLCNT**	RW	0	**Cycle Steal Counter –** This counter will increment every time a cycle is stolen from the DSP for DMA or because of an access conflict in the RAMs

Table 33. Cycle Steal Count Register Description

6.4. Clock Control Register

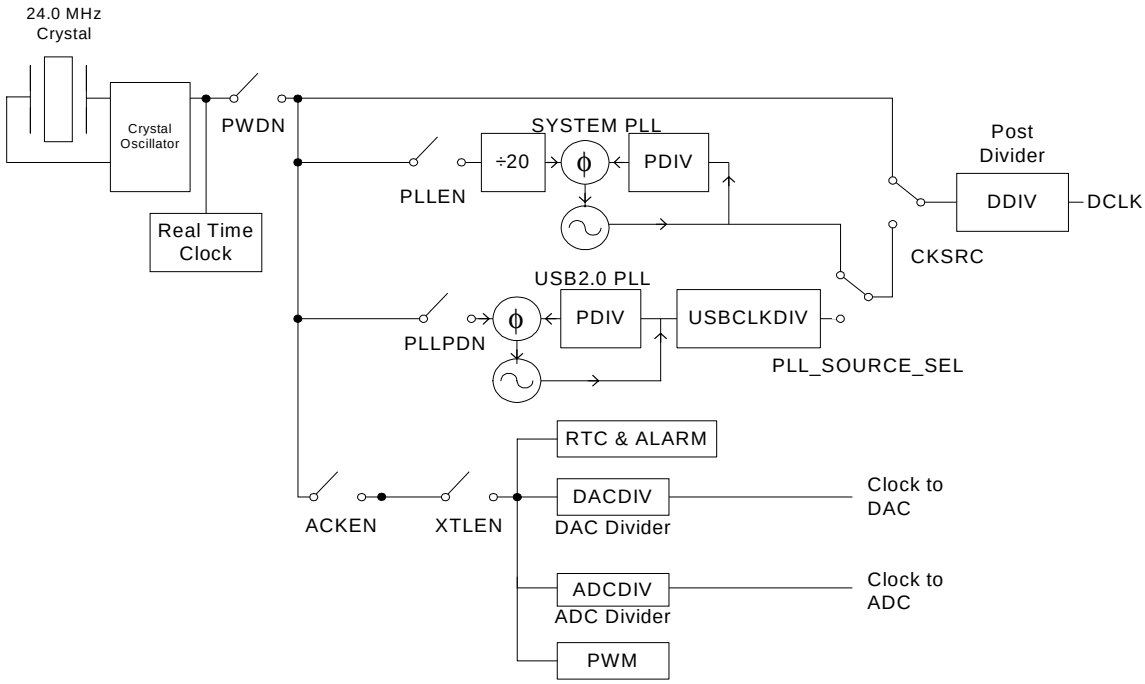


Figure 18. Clock Control Register (HW_CCR)

The Clock Control Register configures the system clock sources, including the Analog clock and PLL. It is also used to shut down system power by turning off the DC-DC converter. Note that none of the bits in this register have any effect unless the CLKRST bit (bit [0]) is set to one.

The digital clock can be set to nearly any value between 12 kHz and 77 MHz using the PLL and post divider.

The various clock multiplexors shown in Figure 18 are fully synchronous “glitch-free” switches. In order to successfully change a clock multiplexor “switch setting”, the clock inputs to each leg must be operating. In particular to throw the switch controlled by **HW_CCR_PLL_SOURCE_SEL**, both the system PLL and the USB 2.0 PLL must be operating. In addition, dividers like **HW_CCR_DDIV** are glitch free upon change. This implies that it takes several clocks to transition to new divider settings. This requires certain rules for changing various controls in the HW_CCR register.

1. **HW_CCR_DDIV** must not be changed in the same clock change window as either **HW_CCR_CKSRC** or **HW_CCR_PLL_SOURCE_SEL**. The clock change window is satisfied with four DSP NOPS.
2. If **DDIV** is going to a larger value, then change it first followed by **CKSRC**. If **DDIV** is going to a smaller value, then change it after **CKSRC**, again observing clock change window.
3. Changing **HW_CCR_PLL_SOURCE_SEL** and **DDIV** have similar constraints.

HW_CCR X:$FA00

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DDIV_MSB	DACDIV			LOCK	ACKEN	PWDN	PDIV					DDIV			CKSRC	ADCDIV			PLL_SOURCE_SEL	XTLEN	PLLEN	LTC	CKRST

Table 34. HW_CCR

BITS	LABEL	RW	RESET	DEFINITION
23	**DDIV_MSB**	RW	0	Extension to digital clock divider, used in conjunction with **DDIV** field below.
22:20	**DACDIV**	RW	000	**Analog clock divider for DAC –** Changes the clock rate used by the DAC. This is used to scale down the frequencies used in the DAC when using audio sample rates less than the maximum. These are calculated for a 24.0MHz crystal (not 24.576MHz). 000 xtal/4 = 6.0 MHz = 128*Fssc = 128*46.875 kHz 001 xtal/6 = 4.0 MHz = 128*Fssc = 128*31.25 kHz 010 xtal/8 = 3.0 MHz = 128*Fssc = 128*23.4375 kHz 011 xtal/12 = 2.0 MHz = 128*Fssc = 128*15.625 kHz 100 xtal/8 = 3.0 MHz = 128*Fssc = 128*23.4375 kHz 101 xtal/12 = 2.0 MHz = 128*Fssc = 128*15.625 kHz 110 xtal/16 = 1.5 MHz = 128*Fssc = 128*11.71875 kHz 111 xtal/24 = 1.0 MHz = 128*Fssc = 128*7.8125 kHz

Table 35. Clock Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
19	**LOCK**	R		**PLL lock status** 0 PLL not locked 1 PLL locked
18	**ACKEN**	RW	0	**Analog clock enable –** The crystal clock runs digital circuitry for the ADC, DAC, PWM and for the TIMERs when in crystal clock mode.This bit enables clocks to the analog circuitry. This bit must be set before using the RTC, Alarm, PWM, Timers, DAC or ADC. In addition to this mode bit, the ADC or DAC power down bits in the Mixer Power Down Control Status Register must not be asserted for the ADC or DAC to operate. 0 Analog clocks disabled 1 Analog clocks enabled
17	**PWDN**	RW	0	**System power-down** 0 No action 1 Power down system *Note: bit 9 of HW_MIXTBR must be set low to enable this functionality* *Note: When the once port is attached, the DCDC converter does not accept a power down signal from the **HW_CCR_PWDN** bit.*
16:12	**PDIV**	RW	00000	**PLL frequency divider –** Assuming a 24.0 MHz crystal, the PLL can be programmed in 1.2 MHz steps, from a minimum frequency of 39.6 MHz to a maximum frequency of 76.8 MHz. The reset value is 00000, which yields a PLL frequency of 39.6 MHz. When used in combination with DDIV post-divider, it is possible to reach frequencies below 39.6 MHz with smaller granularities. PLL output frequency = (33 + PDIV) * (Crystal frequency/20)

11:9	**DDIV**	RW	000	**Digital clock post-divider –** The post divider one cycle to update. It is recommended to change the clock source to be the crystal (via CKSRC) before altering DDIV to limit the maximum frequency the divider will output during the update time. To achieve the minimum power mode, select the crystal as the source for the digital clocks (CKSRC=0), turn off the PLL (PLLEN=0), and set this divider to the maximum divide rate. The digital clock will then be set to 24.0 MHz/2048 = 11.7 kHz.
				if **DDIV_MSB** = 0:
				000 divide by 1 011 divide by 8 110 divide by 64
				001 divide by 2 100 divide by 16 111 divide by 128
				010 divide by 4 101 divide by 32
				if **DDIV_MSB** = 1:
				000 divide by 256 001 divide by 512 010 divide by 1024
				011 divide by 2048 1XX undefined

8	**CKSRC**	RW	0	**Clock source –** This bit may only be set from 0 to 1 if the PLL is enabled and locked. The new clock source may not be available to the system until two periods of the crystal clock have elapsed. 0 Digital clock generated from crystal 1 Digital clock generated from PLL

Table 35. Clock Control Register Description (Continued)

BITS	LABEL	RW	RESET	DEFINITION
7:5	**ADCDIV**	RW	000	**Analog clock divider for ADC –** Changes the clock rate used by the ADC. This is used to scale down the frequencies used in the ADC when using audio sample rates less than the maximum.These are calculated for a 24.0MHz crystal (not 24.576MHz). 000 xtal/4 = 6.0 MHz = 128*Fssc = 128*46.875 kHz 001 xtal/6 = 4.0 MHz = 128*Fssc = 128*31.25 kHz 010 xtal/8 = 3.0 MHz = 128*Fssc = 128*23.4375 kHz 011 xtal/12 = 2.0 MHz = 128*Fssc = 128*15.625 kHz 100 xtal/8 = 3.0 MHz = 128*Fssc = 128*23.4375 kHz 101 xtal/12 = 2.0 MHz = 128*Fssc = 128*15.625 kHz 110 xtal/16 = 1.5 MHz = 128*Fssc = 128*11.71875 kHz 111 xtal/24 = 1.0 MHz = 128*Fssc = 128*7.8125 kHz
4	**PLL_SOURCE_SEL**	RW	0	**PLL_SOURCE_SEL –** This bit selects either the low speed system PLL or the USB 2 PHY high speed PLL as the source for DCLK, see Figure 33. “USB 2.0 PHY PLL Block Diagram” on page 88. 0 select system PLL 1 select USB 2.0 PHY high speed PLL
3	**XTLEN**	RW	0	**Crystal clock enable –** The crystal clock runs digital circuitry for the ADC, DAC, PWM and for the TIMERs when in crystal clock mode. This clock should be disabled when all of these blocks are not in use. The crystal clock must enabled for these blocks to function correctly. 0 Crystal clock disabled 1 Crystal clock enabled
2	**PLLEN**	RW	0	**PLL enable** 0 PLL disabled 1 PLL enabled
1	**LTC**	RW	0	**Lock timer reset –** Resets the PLL lock timer. Must be written as 0, then 1 to start the PLL lock timer.
0	**CKRST**	RW	0	**Clock reset –** Must be written with 1 for all writes to this register. If set to 0, the entire clock logic block stays in its reset state.

Table 35. Clock Control Register Description (Continued)

6.5. Misc./Spare Register

The Misc./Spare Register is used for system updates (TBD as they are available). It is used to detect the state of the analog PSWITCH pin and to configure the I^2S pins onto alternate pins. It also indicates when the ONCE port is plugged in and activated. The organization of the Misc./Spare Register is shown below.

HW_SPARER X:$FA16

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
													ONCE_MODE	PSWITCH		SPARE_BITS							I2S_SELECT

Table 36. HW_SPARER

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10	**ONCE_MODE**	R	0	When the once port debugger is attached to the chip several special operating modes are forced, e.g. power down is suppressed, etc. This bit is set to one to indicate when the once port debugger is attached and enabled. It is set to zero for normal operation.
9	**PSWITCH**	R		**PSWITCH status –** This bit indicates the state of the PSWITCH pin. Normally, this pin will be a 1 when the power switch is pressed. 0 PSWITCH pin is 0 1 PSWITCH pin is 1
8	**RSRVD**	R	0	**Reserved –** Must be written with 0.
7:2	**SPARE_BITS**	RW	0	**Spare bits –** These bits can be written or read normally but have no function in this revision of the chip.
1	**RSRVD**	R	0	**Reserved –** Must be written with 0.
0	**I2S_SELECT**	RW	0	**I2S select –** This bit is used to switch the I^2S functions onto alternate pins. One setting allows a subset of the I^2S functionality to be supported in a 100-pin package, the other setting allows the full I^2S functionality but is only supported in a 144-pin package. (see 22.1. “I^2S External Pins” on page 253.) 0 Subset of I^2S functionality for the 100-pin package 1 Full I^2S functionality for the 144-pin package

Table 37. Misc./Spare Register Description

6.6. PIN Control Register

Special controls related to the optional low voltage pin mode. Also see 31. “PIN DESCRIPTION” on page 368.

HW_PIN_CTRL X:$FA30

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											ESD_TEST				KEEPER_EN	PIN1_8V							

Table 38. HW_PIN_CTRL

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12	**ESD_TEST**	R	0	**ESD Test Mode –** Set to one to deactivate the ESD clamp devices. There is an internal delay of 400 to 600 nSeconds after this bit is set to one until the clamp is deactivated. NOTE: this bit is read only on the STMP35xx TA1.
11:9	**RSRVD**	R	000	**Reserved –** Must be written with 0.
8	**KEEPER_EN**	RW	0	**Keeper Enable –** Each of the 1.8V capable pads have an optional small keeper that can be enabled when this bit is set to one. WARNING: do NOT set **KEEPER_EN** to one until ***AFTER*** **PAD1_8V** bits have been set to one.
7:0	**PIN1_8V**	RW	$00	**1.8Volt Pin Control –** These bits select between 3.3V and 1.8V PAD I/O functions for the 8 pad groups, see Table 40. “Pin Control to PIn Mapping” on page 48. Set these pins to one ***BEFORE*** setting **KEEPER_EN** to one.

Table 39. Pin Control Register Description

100 TQFP PIN #	144 FPBGA PIN #	GPIO PIN #	PIN NAME	1.8 V ENABLE BIT
1	M2	14	SPI_MOSI	**HW_PIN_CTRL_PIN1_8V[7]**
2	L2	13	SPI_MISO	**HW_PIN_CTRL_PIN1_8V[7]**
3	K4	12	SPI_SCK	**HW_PIN_CTRL_PIN1_8V[7]**
16	K7	36	CF/RAM A4, SM CE3n	**HW_PIN_CTRL_PIN1_8V[0]**
17	J7	37	CF/RAM A5, SM CE2n	**HW_PIN_CTRL_PIN1_8V[1]**
18	K8	38	CF/RAM A6, SM CE0n	**HW_PIN_CTRL_PIN1_8V[2]**
20	L9	40	CF/RAM A8, SM CLE	**HW_PIN_CTRL_PIN1_8V[4]**
21	L10	41	CF/RAM A9, SM ALE	**HW_PIN_CTRL_PIN1_8V[4]**
23	M11	53	CF OEn, SM REn	**HW_PIN_CTRL_PIN1_8V[4]**
24	K10	45	CF CE0n, SM CE1n	**HW_PIN_CTRL_PIN1_8V[3]**
25	M12	56	CF WAITn, SM READY	**HW_PIN_CTRL_PIN1_8V[4]**
26	L11	54	CF WEn, SM WEn	**HW_PIN_CTRL_PIN1_8V[4]**

Table 40. Pin Control to PIn Mapping

100 TQFP PIN #	144 FPBGA PIN #	GPIO PIN #	PIN NAME	1.8 V ENABLE BIT
27	K12	55	SM WPn	**HW_PIN_CTRL_PIN1_8V[4]**
30	J10	24	CF/RAM/SM D0	**HW_PIN_CTRL_PIN1_8V[5]**
-	J12	79	CF/RAM/SM D15	**HW_PIN_CTRL_PIN1_8V[6]**
31	H9	25	CF/RAM/SM D1	**HW_PIN_CTRL_PIN1_8V[5]**
-	J11	78	CF/RAM/SM D14	**HW_PIN_CTRL_PIN1_8V[6]**
32	H12	26	CF/RAM/SM D2	**HW_PIN_CTRL_PIN1_8V[5]**
-	H11	77	CF/RAM/SM D13	**HW_PIN_CTRL_PIN1_8V[6]**
33	H10	27	CF/RAM/SM D3	**HW_PIN_CTRL_PIN1_8V[5]**
-	G11	76	CF/RAM/SM D12	**HW_PIN_CTRL_PIN1_8V[6]**
34	G10	28	CF/RAM/SM D4	**HW_PIN_CTRL_PIN1_8V[5]**
-	G12	75	CF/RAM/SM D11	**HW_PIN_CTRL_PIN1_8V[6]**
35	G9	29	CF/RAM/SM D5	**HW_PIN_CTRL_PIN1_8V[5]**
-	F9	74	CF/RAM/SM D10	**HW_PIN_CTRL_PIN1_8V[6]**
36	F11	30	CF/RAM/SM D6	**HW_PIN_CTRL_PIN1_8V[5]**
-	F12	73	CF/RAM/SM D9	**HW_PIN_CTRL_PIN1_8V[6]**
37	F10	31	CF/RAM/SM D7	**HW_PIN_CTRL_PIN1_8V[5]**
-	E9	72	CF/RAM/SM D8	**HW_PIN_CTRL_PIN1_8V[6]**

Table 40. Pin Control to PIn Mapping (Continued)

7. INTERRUPT SUBSYSTEM

7.1. Interrupt Priority Register

The DSP core has seven main interrupt lines, IVL[6:0]. This register is used to enable each line and set its priority. Some peripherals connect directly to this interrupt bus, but most interrupt sources go through the Interrupt collector, which multiplexes many interrupt sources onto 4 interrupts of these 7 interrupt lines.

If an interrupt with a higher priority level occurs while the DSP core is servicing another interrupt, the higher priority interrupt will preempt the lower priority interrupt. If the new interrupt is of the same or lower priority level, then it will not preempt the interrupt that is currently being serviced.

HW_IPR X:$FFFF

BITS	LABEL	RW	RESET	DEFINITION
23:22	**L6P**	RW	0	**Interrupt line 6 priority level** 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
21:20	**L5P**	RW	0	**Interrupt line 5 priority level –** SPI 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
19:18	**L4P**	RW	0	**Interrupt line 4 priority level –** I^2C 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
17:16	**L3P**	RW	0	**Interrupt line 3 priority level** 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
15:14	**L2P**	RW	0	**Interrupt line 2 priority level** 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
13:12	**L1P**	RW	0	**Interrupt line 1 priority level** 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
11:10	**L0P**	RW	0	**Interrupt line 0 priority level –** I^2S 00 Disabled 01 Priority Level 0 (lowest priority level) 10 Priority Level 1 11 Priority Level 2 (highest priority level)
9:6	**RSRVD**	R	0	**Reserved –** Must be written with 0.
5	**IRQBT**	RW	0	**IRQB Type** 0 Level 1 Negative Edge
4:3	**IRQBP**	RW	0	**IRQB Priority Level** 00 Disable 01 Enable 10 Enable 11 Enable
2	**IRQAT**	RW	0	**IRQA Type** 0 Level 1 Negative Edge
1:0	**IRQAP**	RW	0	**IRQA Priority Level** 00 Disable 01 Enable 10 Enable 11 Enable

Table 41. Interrupt Priority Register Description

7.2. Interrupt Collector

The DSP core provides seven interrupt lines, IVL[6:0], for individual interrupt requests. The peripheral interrupt count exceeds the seven interrupt request lines, so the Interrupt Collector (ICOLL) muxes all sources to the seven lines. Three of the seven interrupt lines are reserved for certain peripherals, so the ICOLL steers 36 interrupt sources to four of the interrupt request lines: IVL[6,3,2,1]. Within an individual interrupt request line, the ICOLL offers an 8-level priority for each of it is interrupt sources, although preemption of a lower priority interrupt by a higher priority is not supported. If preemption is required, the interrupt source that needs to preempt must be routed to a higher priority interrupt request line.

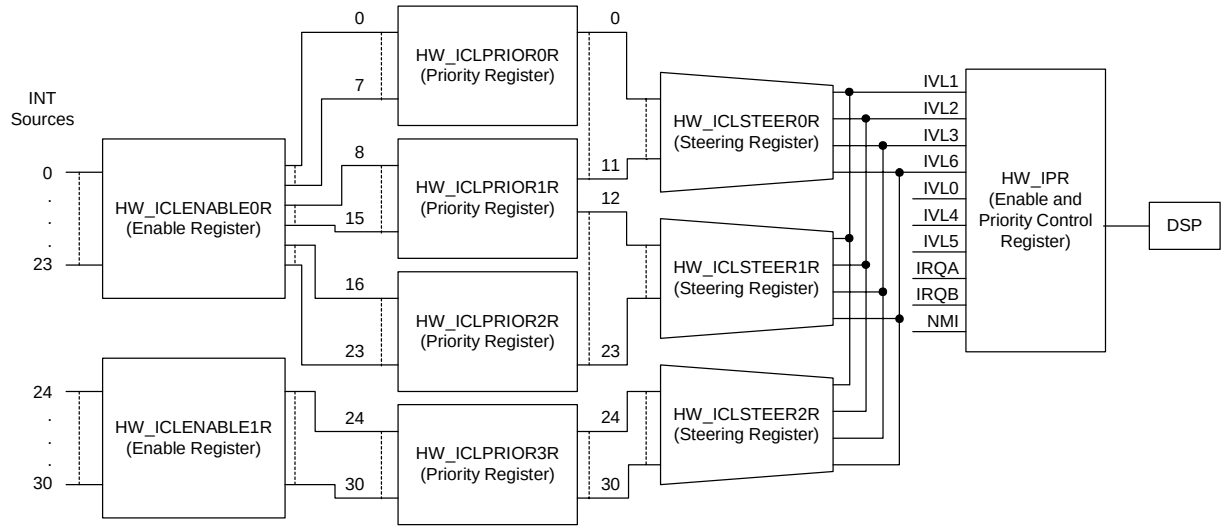


Figure 19. Interrupt Collector Diagram

Interrupt response time is dependent on the interrupt priority. The minimum interrupt time is as follows:

- 2 clock cycles for interrupt to appear at DSP.
- 2 clock cycles to respond and get vector to execute interrupt service routine.

7.2.1. Interrupt Sources

Table 42 shows all the interrupt sources.

INTERRUPT SOURCE	SRC	IML BIT #	VECTOR	DESCRIPTION
DAC Refill	0	6,3,2,1	$003C	DAC request to fill DAC FIFO buffer
DAC Underflow	1	6,3,2,1	$003E	DAC FIFO buffer underflow
ADC Refill	2	6,3,2,1	$0042	ADC request to empty ADC FIFO buffer
ADC Overflow	3	6,3,2,1	$0044	ADC FIFO buffer overflow
GP Flash Done	4	6,3,2,1	$006E	Flash transaction complete
EMC-CompactFlash Card IRQ	5	6,3,2,1	$0070	CompactFlash card interrupt
EMC-SmartMedia/NAND Timeout	6	6,3,2,1	$0072	SmartMedia/NAND card WAIT timeout
EMC-SmartMedia/NAND Interface Invalid Programming	7	6,3,2,1	$0074	Bad programming of SmartMedia/NAND interface
EMC-CompactFlash No Card	8	6,3,2,1	$0076	CompactFlash card remove/absence
EMC-CompactFlash Status Change	9	6,3,2,1	$0078	CompactFlash card status change
GPIO0	10	6,3,2,1	$0024	GPIO module 0 interrupt
GPIO1	11	6,3,2,1	$0020	GPIO module 1 interrupt
GPIO2	12	6,3,2,1	$0022	GPIO module 2 interrupt
Timer0	13	6,3,2,1	$0026	Timer module 0 interrupt
Timer1	14	6,3,2,1	$0028	Timer module 1 interrupt
Timer2	15	6,3,2,1	$002A	Timer module 2 interrupt
Timer3	16	6,3,2,1	$0048	Timer module 3 interrupt
GPIO3	17	6,3,2,1	$004A	GPIO module 3 interrupt
SDRAM	18	6,3,2,1	$004C	SDRAM interrupt
CDI	19	6,3,2,1	$007E	CDI interface interrupt
Vdd5V Connected	20	6,3,2,1	$0050	Vdd5V Connected(Vdd5V > VddIO+0.5V)
USB Controller	21	6,3,2,1	$0052	USB Controller Interrupt
USB Wakeup	22	6,3,2,1	$0054	USB Wakeup (Resume from Suspend)
Vdd5V Disconnected	23	6,3,2,1	$0056	Vdd5V pin disconnected from 5V
ESPI	24	6,3,2,1	$0058	ESPI (enhanced SPI)
FILCO	25	6,3,2,1	$005A	Filter Coprocessor
LRADC1	26	6,3,2,1	$005C	Low Resolution ADC
RTC Alarm	27	6,3,2,1	$005E	Real Time Clock Alarm
LRADC2	28	6,3,2,1	$0060	Low Resolution ADC #2
Flash ECC	29	6,3,2,1	$0062	Hardware ECC accelerator for ECC
-	30	6,3,2,1		reserved
CDSync Interrupt	31	6,3,2,1	$0066	CD synchronizer/formatter interrupt
CDSync Exception	32	6,3,2,1	$0068	CD synchronizer/formatter exception
CD-RS Interrupt	33	6,3,2,1	$006A	Reed-Solomon error corrector interrupt
I^2C Rx Ready	-	4	$0030	I^2C receiver data ready
I^2C Rx Overflow	-	4	$0032	I^2C receiver data overflow
I^2C Tx Empty	-	4	$0034	I^2C transmitter data empty
I^2C Tx Underflow	-	4	$0036	I^2C transmitter data underflow
SPI Complete	-	5	$000E	SPI transfer complete
I^2S Rx Overflow	-	0	$0016	I^2S receiver data overflow
I^2S Tx Underflow	-	0	$0012	I^2S transmitter data underflow
I^2S Rx Ready	-	0	$0014	I^2S receiver data ready
I^2S Tx Empty	-	0	$0010	I^2S transmitter data empty
IRQA - Headphone Short	-	IRQA	$0008	Headphone Short
IRQB - Battery LRADC	-	IRQB	$000A	Battery Brown Out
NMI	-	NMI	$001E	Non-Maskable Interrupt

Table 42. Interrupt Sources

7.2.2. Interrupt Vectors

Table 43 below shows the interrupt vectors used for each possible interrupt source.

ADDRESS	INTERRUPT SOURCE	ADDRESS	INTERRUPT SOURCE
P:$0000	Hardware Reset	P:$0040	-
P:$0002	Stack Error	P:$0042	ADC Full
P:$0004	Trace	P:$0044	ADC Overflow
P:$0006	SWI	P:$0046	-
P:$0008	IRQA/Headphone Short Detect	P:$0048	Timer 3
P:$000A	IRQB/Battery Brownout Detect	P:$004A	GPIO3
P:$000C	-	P:$004C	SDRAM
P:$000E	SPI Complete	P:$004E	-
P:$0010	I^2S Tx Data Empty	P:$0050	Vdd5V Pin Connected to 5V
P:$0012	I^2S Tx Underflow	P:$0052	USB Controller
P:$0014	I^2S Rx Data Full	P:$0054	USB Wakeup (Resume)
P:$0016	I^2S Rx Overflow	P:$0056	Vdd5V Pin Disconnected
P:$0018	-	P:$0058	ESPI (enhanced SPI)
P:$001A	-	P:$005A	FILCO (filter coprocessor)
P:$001C	-	P:$005C	Low Resolution ADC 1
P:$001E	NMI	P:$005E	RTC Alarm
P:$0020	GPIO 1	P:$0060	Low Resolution ADC 2
P:$0022	GPIO 2	P:$0062	Flash ECC Accelerator
P:$0024	GPIO 0	P:$0064	-
P:$0026	Timer 0	P:$0066	CDSync Interrupt
P:$0028	Timer 1	P:$0068	CDSync Exception
P:$002A	Timer 2	P:$006A	CD-RS Interrupt
P:$002C	-	P:$006C	-
P:$002E	-	P:$006E	GPFlash Done
P:$0030	I^2C Rx Data Ready	P:$0070	EMC-CompactFlash Card IRQ
P:$0032	I^2C Rx Overflow	P:$0072	EMC-SmartMedia/NAND Timeout
P:$0034	I^2C Tx Data Empty	P:$0074	EMC-SmartMedia/NAND Invalid Programming
P:$0036	I^2C Tx Underflow	P:$0076	EMC-CompactFlash No Card
P:$0038	Invalid DSP instruction	P:$0078	EMC-CompactFlash Status Change
P:$003A	-	P:$007A	-
P:$003C	DAC Empty	P:$007C	-
P:$003E	DAC Underflow	P:$007E	CDI Interrupt

Table 43. Interrupt Vector Map

7.2.3. Interrupt Collector Registers

7.2.3.1. ICOLL Enable 0 Register

The enable registers provide bits to enable/disable interrupts for each source. Each enable corresponds to a specific interrupt source listed in Table 42. A 1 enables the relevant interrupt, a 0 disables the relevant interrupt.

HW_ICLENABLE0R X:$F300

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SEN23:SEN0**	RW	0	0 Disables this interrupt 1 Enables this interrupt

Table 44. ICOLL Enable 0 Register Description

7.2.3.2. ICOLL Enable 1 Register

The enable registers provide bits to enable/disable interrupts for each source. Each enable corresponds to a specific interrupt source listed in Table 42. A 1 enables the relevant interrupt, a 0 disables the relevant interrupt.

HW_ICLENABLE1R X:$F301

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**SEN33:SEN24**	RW	0	0 Disables this interrupt 1 Enables this interrupt

Table 45. ICOLL Enable 1 Register Description

7.2.3.3. ICOLL Status 0 Register

The status registers reflect the interrupt state of each source. Each enable corresponds to a specific interrupt source listed in Table 42. A 1 indicates an active interrupt, a 0 indicates an inactive interrupt. This register is read only.

HW_ICLSTATUS0R X:$F302

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SST23:SST0**	R		0 Interrupt is not active 1 Interrupt is active

Table 46. ICOLL Status 0 Register Description

7.2.3.4. ICOLL Status 1 Register

The status registers reflect the interrupt state of each source. Each enable corresponds to a specific interrupt source listed in Table 42. A 1 indicates an active interrupt, a 0 indicates an inactive interrupt. This register is read only.

HW_ICLSTATUS1R X:$F303

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**SST33:SST24**	RW	0	0 Interrupt is not active 1 Interrupt is active

Table 47. ICOLL Status 1 Register Description

7.2.3.5. ICOLL Priority 0 Register

The priority registers set the priority for each source. Each enable corresponds to a specific interrupt source listed in Table 42. Lowest priority is 111 and highest is 000.

HW_ICLPRIOR0R X:$F304

BITS	LABEL	RW	RESET	DEFINITION
23:21	**S7P**	RW	Unknown	**Priority level for interrupt source 0** 000 Highest priority level . . . 111 Lowest priority level
20:18	**S6P**	RW	Unknown	
17:15	**S5P**	RW	Unknown	
14:12	**S4P**	RW	Unknown	
11:9	**S3P**	RW	Unknown	
8:6	**S2P**	RW	Unknown	
5:3	**S1P**	RW	Unknown	
2:0	**S0P**	RW	Unknown	

Table 48. ICOLL Priority 0 Register Description

7.2.3.6. ICOLL Priority 1 Register

The priority registers set the priority for each source. Each enable corresponds to a specific interrupt source listed in Table 42. Lowest priority is 111 and highest is 000.

HW_ICLPRIOR1R X:$F305

BITS	LABEL	RW	RESET	DEFINITION
23:21	**S15P**	RW	Unknown	**Priority level for interrupt source 1** 000 Highest priority level . . . 111 Lowest priority level
20:18	**S14P**	RW	Unknown	
17:15	**S13P**	RW	Unknown	
14:12	**S12P**	RW	Unknown	
11:9	**S11P**	RW	Unknown	
8:6	**S10P**	RW	Unknown	
5:3	**S9P**	RW	Unknown	
2:0	**S8P**	RW	Unknown	

Table 49. ICOLL Priority 1 Register Description

7.2.3.7. ICOLL Priority 2 Register

The priority registers set the priority for each source. Each enable corresponds to a specific interrupt source listed in Table 42. Lowest priority is 111 and highest is 000.

HW_ICLPRIOR2R X:$F306

BITS	LABEL	RW	RESET	DEFINITION
23:21	**S23P**	RW	Unknown	**Priority level for interrupt source 2** 000 Highest priority level . . . 111 Lowest priority level
20:18	**S22P**	RW	Unknown	
17:15	**S21P**	RW	Unknown	
14:12	**S20P**	RW	Unknown	
11:9	**S19P**	RW	Unknown	
8:6	**S18P**	RW	Unknown	
5:3	**S17P**	RW	Unknown	
2:0	**S16P**	RW	Unknown	

Table 50. ICOLL Priority 2 Register Description

7.2.3.8. ICOLL Priority 3 Register

The priority registers set the priority for each source. Each enable corresponds to a specific interrupt source listed in Table 42. Lowest priority is 111 and highest is 000.

HW_ICLPRIOR3R X:$F307

BITS	LABEL	RW	RESET	DEFINITION
23:21	**S31P**	RW	Unknown	**Priority level for interrupt source 3** 000 Highest priority level . . . 111 Lowest priority level
20:18	**S30P**	RW	Unknown	
17:15	**S29P**	RW	Unknown	
14:12	**S28P**	RW	Unknown	
11:9	**S27P**	RW	Unknown	
8:6	**S26P**	RW	Unknown	
5:3	**S25P**	RW	Unknown	
2:0	**S24P**	RW	Unknown	

Table 51. ICOLL Priority 3 Register Description

7.2.3.9. ICOLL Priority 4 Register

The priority registers set the priority for each source. Each enable corresponds to a specific interrupt source listed in Table 42. Lowest priority is 111 and highest is 000.

HW_ICLPRIOR4R X:$F311

BITS	LABEL	RW	RESET	DEFINITION
23:6	**RSRVD**	R	0	**Reserved –** Must be written with 0.
5:3	**S33P**	RW	Unknown	**Priority level for interrupt source 4** 000 Highest priority level . . . 111 Lowest priority level
2:0	**S32P**	RW	Unknown	

Table 52. ICOLL Priority 4 Register Description

7.2.3.10. ICOLL Steering 0 Register

The steering registers are used to steer a given source to a given IVL as follows:

SETTING	IVL
00	1
01	2
10	3
11	6

Each steering value corresponds to a specific interrupt source listed in Table 42.

HW_ICLSTEER0R X:$F308

BITS	LABEL	RW	RESET	DEFINITION
23:22	**S11S**	RW	Unknown	
21:20	**S10S**	RW	Unknown	
19:18	**S9S**	RW	Unknown	
17:16	**S8S**	RW	Unknown	
15:14	**S7S**	RW	Unknown	
13:12	**S6S**	RW	Unknown	
11:10	**S5S**	RW	Unknown	
9:8	**S4S**	RW	Unknown	
7:6	**S3S**	RW	Unknown	
5:4	**S2S**	RW	Unknown	

Table 53. ICOLL Steering 0 Register Description

BITS	LABEL	RW	RESET	DEFINITION
3:2	**S1S**	RW	Unknown	
1:0	**S0S**	RW	Unknown	

Table 53. ICOLL Steering 0 Register Description

7.2.3.11. ICOLL Steering 1 Register

The steering registers are used to steer a given source to a given IVL as follows:

SETTING	IVL
00	1
01	2
10	3
11	6

Each steering value corresponds to a specific interrupt source listed in Table 42.

HW_ICLSTEER1R X:$F309

BITS	LABEL	RW	RESET	DEFINITION
23:22	**S23S**	RW	Unknown	
21:20	**S22S**	RW	Unknown	
19:18	**S21S**	RW	Unknown	
17:16	**S20S**	RW	Unknown	
15:14	**S19S**	RW	Unknown	
13:12	**S18S**	RW	Unknown	
11:10	**S17S**	RW	Unknown	
9:8	**S16S**	RW	Unknown	
7:6	**S15S**	RW	Unknown	
5:4	**S14S**	RW	Unknown	
3:2	**S13S**	RW	Unknown	
1:0	**S12S**	RW	Unknown	

Table 54. ICOLL Steering 1 Register Description

7.2.3.12. ICOLL Steering 2 Register

The steering registers are used to steer a given source to a given IVL as follows:

SETTING	IVL
00	1
01	2
10	3
11	6

Each steering value corresponds to a specific interrupt source listed in Table 42.

HW_ICLSTEER2R X:$F30A

BITS	LABEL	RW	RESET	DEFINITION
23:20	**RSRVD**	R		**Reserved –** Must be written with 0.
19:18	**S33S**	RW	Unknown	
17:16	**S32S**	RW	Unknown	
15:14	**S31S**	RW	Unknown	
13:12	**S30S**	RW	Unknown	
11:10	**S29S**	RW	Unknown	
9:8	**S28S**	RW	Unknown	
7:6	**S27S**	RW	Unknown	
5:4	**S26S**	RW	Unknown	
3:2	**S25S**	RW	Unknown	
1:0	**S24S**	RW	Unknown	

Table 55. ICOLL Steering 2 Register Description

7.2.4. Interrupt Collector Debug Registers

7.2.4.1. ICOLL Debug Force 0 Register

The debug force value registers will force an interrupt for a given source. The enable registers enable the forcing mechanism. Each enable corresponds to a specific interrupt source listed in Table 42.

HW_ICLFORCE0R X:$F30B

BITS	LABEL	RW	RESET	DEFINITION
23:0	**S23FV:S0FV**	RW	0	

Table 56. ICOLL Force Value 0 Register Description

7.2.4.2. ICOLL Debug Force 1 Register

The debug force value registers will force an interrupt for a given source. The enable registers enable the forcing mechanism. Each enable corresponds to a specific interrupt source listed in Table 42.

HW_ICLFORCE1R X:$F30C

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**S33FV:S24FV**			

Table 57. ICOLL Force Value 1 Register Description

7.2.4.3. ICOLL Force Enable 0 Register

To generate a forced interrupt you have to write a 1 into the relevant position in both the force and force enable registers. Writing a 1 to the force enable register will block any interrupts from the normal interrupt source for the relevant bit. Each force bit corresponds to a specific interrupt source listed in Table 42.

HW_ICLFENABLE0R X:$F30D

BITS	LABEL	RW	RESET	DEFINITION
23:0	**S23FE:S0FE**	RW	0	

Table 58. ICOLL Force Enable 0 Register Description

7.2.4.4. ICOLL Force Enable 1 Register

To generate a forced interrupt you have to write a 1 into the relevant position in both the force and force enable registers. Writing a 1 to the force enable register will block any interrupts from the normal interrupt source for the relevant bit. Each force bit corresponds to a specific interrupt source listed in Table 42.

HW_ICLFENABLE1R X:$F30E

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**S33FE:S24FE**	RW	0	

Table 59. ICOLL Force Enable 1 Registers Description

7.2.5. ICOLL Observation Registers (HW_ICLOBSV0R/1R)

The observation registers make visible the state of the interrupt request vector that the ICOLL is sending out on IVL[6:0] and the winning (prioritized) interrupt vectors for destinations A, B, C, and D.

A IVL1

B IVL2

C IVL3

D IVL6

Each observation bit corresponds to a specific interrupt source listed in Table 42. This register is read only and is primarily for debug purposes.

7.2.5.1. Interrupt Collector Observe 0 Register

HW_ICLOBSVZ0R X:$F30F

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	0	**Reserved –** Must be written with 0.
20:14	**IVB**	R		
13:7	**IVA**	R		
6:0	**REQ**	R		

Table 60. ICOLL Observe 0 Register Description

7.2.5.2. Interrupt Collector Observe 1 Register

HW_ICLOBSVZ1R X:$F310

BITS	LABEL	RW	RESET	DEFINITION
23:14	**RSRVD**	R	0	**Reserved –** Must be written with 0.
13:7	**IVD**	R		
6:0	**IVC**	R		

Table 61. ICOLL Observe 1 Register Description

8. USB CONTROLLER

The STMP35xx includes a Universal Serial Bus (USB) version 2.0 device controller. The USB controller is used to download digital music data or program code into external memory and to upload voice recordings from memory to the PC. Program updates can also be loaded into the flash memory area using the USB interface.

The reader should refer to the USB Implementer's Forum website www.usb.org for detailed specifications and information on the USB protocol, timing and electrical characteristics.

The USB 2.0 controller comprises both a programmed I/O (PIO) interface and a DMA interface, see Figure 10. "USB Interface Block Diagram" on page 14. Both of these interfaces, as implemented in the ARC High Speed USB core, are designed to meet a VSI Alliance Basic Virtual Component Interface BVCI, see www.vsi.org. The BVCI used by the USB controller has a target (PIO) and initiator (DMA) data bus of 32-bits. In addition, the initiator address bus is also 32 bits wide.

In integrating this core into the STMP35xx, the 24 bit nature of the DSP is mapped onto the 32 bit nature of the BVCI by a SigmaTel developed USB interface block. In addition, the STMP35xx only makes use of a 16 bit subset of the BVCI initiator's 32 bit address. For the following discussion, the USB interface block is broken into a target mapping function and an initiator mapping function. A third portion of the USB interface block maps the ARC core USB controller to the UTMI PHY interface. see Figure 20. "USB 2.0 Device Controller" on page 62.

8.1. USB Programmed I/O (PIO) Target Interface

The programmed I/O interface uses three 24 bit registers, **HW_USBARCACCESS**, **HW_USBARCDATALOW,** and **HW_USBARCDATAHIGH**. A hardware state machine then translates PIO requests from the DSP into target bus cycles for the ARC USB 2.0 Device Controller core. see Figure 21. "USB 2.0 PIO Target Interface" on page 63. To write data to a register in the USB device controller, software first loads the DATAHIGH and DATALOW registers with the upper and lower 16 bit parts of the 32 bit data to be written. It then loads the ARC access register, **HW_USBARCACCESS**, fields with the address, the Read/Write bit and sets the Kick bit to one. Setting the Kick bit starts the state machine which transfers the 32 bit data into a USB controller PIO register. All transfers to and from the USB controller are exactly 32 bits wide.

The Kick bit will remain set until the state machine finishes the target write cycle. While the Kick bit is high, software must not modify any field in the **HW_USBARCACCESS**, **HW_USBARCDATALOW,** or **HW_USBARCDATAHIGH** registers. Fortunately the maximum number of cycles the Kick bit can be high is ***TBD*** DCLKs. As long as the software path length is guaranteed to be at least this long until the next PIO operation is initiated, software does not have to poll the kick bit.

In order to read a PIO register from the USB controller, software loads the ARC access register, **HW_USBARCACCESS**, fields with the address, the Read/Write bit

and sets the Kick bit to one. The Read/Write bit is set to one for reads and zero for writes.

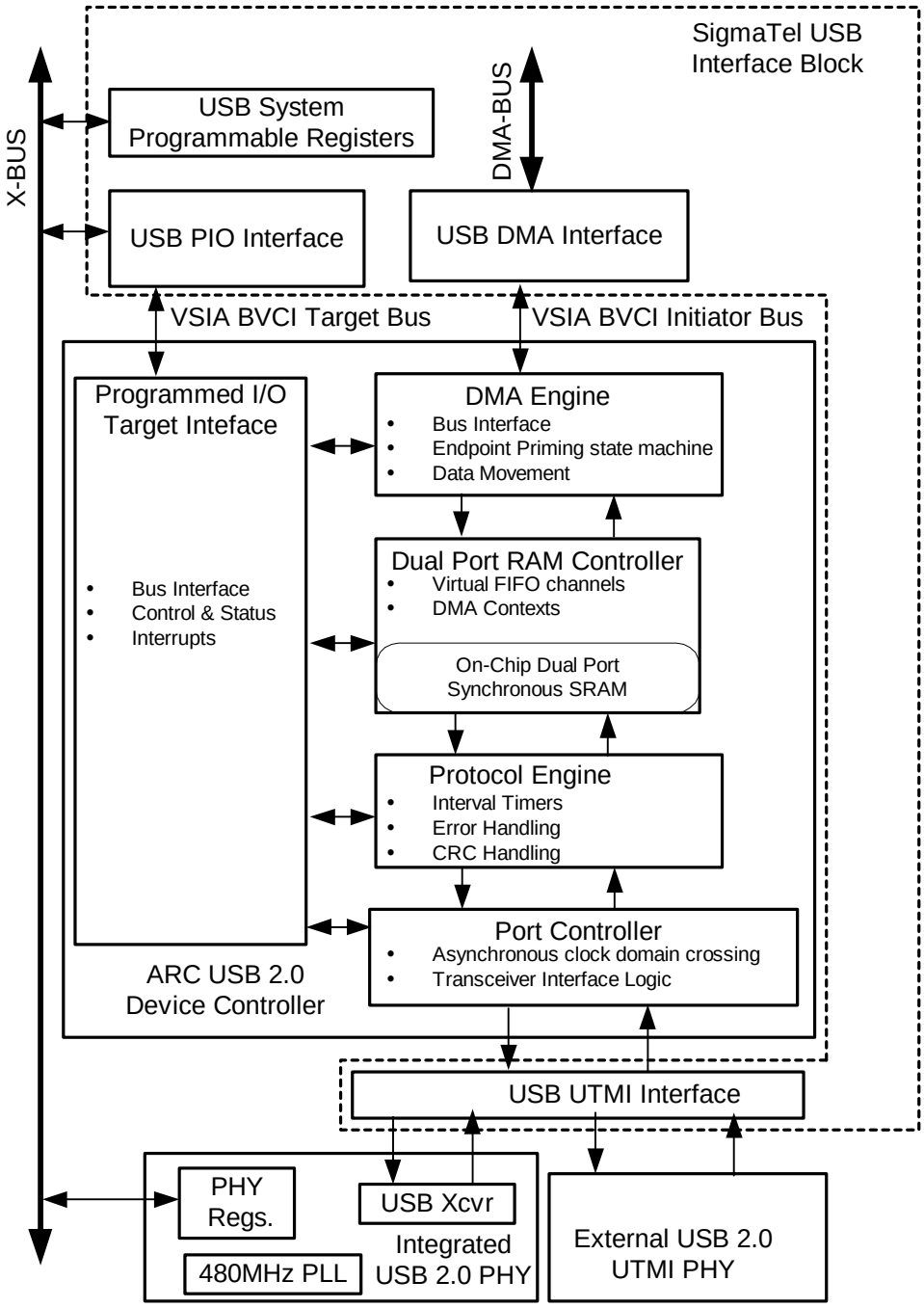


Figure 20. USB 2.0 Device Controller

The target bus and the intitiator busses emanating from the USB Device Controller do not share any resources in common. Thus no arbitration or other startup delay is encounter in performing PIO operations to/from the USB Device Controller.

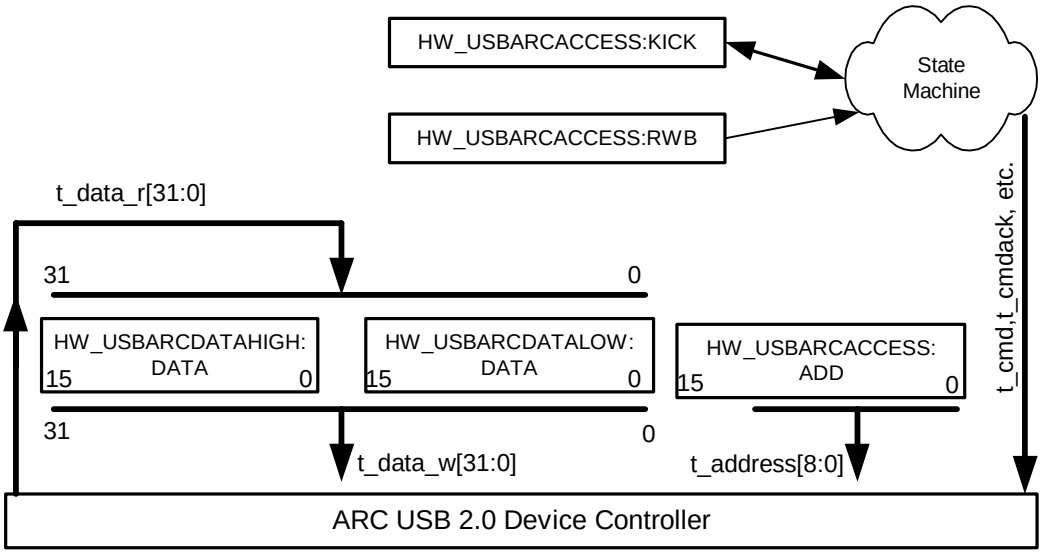


Figure 21. USB 2.0 PIO Target Interface

C versions of generalized PIO read and PIO write routines are given below, see Figure 22. “USB 2.0 PIO Target Interface Sample Code” on page 63. These routines are provided to facilitate understanding. Actual software uses tightly coded assembly routines to manipulate the USB Controller’s programmable registers.

```
//This function unpacks the 48-bit double word into two 16-bit words and writes
// it back to the ARC register
void _reentrant write_usb_reg(USHORT usRegAdd, DWORD dwData)
{
    HW_USBARCDATALOW.B.DATA =
                (WORD)(dwData & (WORD)(HW_USBARCDATALOW_DATA_SETMASK));
    HW_USBARCDATAHIGH.B.DATA =
                (WORD)((dwData >> 16) & (DWORD)(HW_USBARCDATAHIGH_DATA_SETMASK));
    usRegAdd &= (USHORT)HW_USBARCACCESS_ADD_SETMASK;
    HW_USBARCACCESS.B.ADD = usRegAdd;
    HW_USBARCACCESS.B.RWB = 0;
    HW_USBARCACCESS.B.KICK = 1;
}

// This function reads the ARC 32-bit register and packs it in a 48-bit double word
// Use this function in combination with write_usb_reg() to read-modify-write
// an ARC register
void _reentrant read_usb_reg(USHORT usRegAdd, DWORD * dwData)
{
    usRegAdd &= HW_USBARCACCESS_ADD_SETMASK;
    HW_USBARCACCESS.B.ADD = usRegAdd;
    HW_USBARCACCESS.B.RWB = 1;
    HW_USBARCACCESS.B.KICK = 1;
    while(HW_USBARCACCESS.B.KICK); // wait for state machine
    *dwData = (DWORD)(HW_USBARCDATALOW.B.DATA & HW_USBARCDATALOW_DATA_SETMASK);
    *dwData |= (DWORD)(HW_USBARCDATAHIGH.B.DATA << 16);
}
```

Figure 22. USB 2.0 PIO Target Interface Sample Code

8.2. USB DMA Interface

Most software interactions with the USB device controller occur with data or control structures that have been DMA transferred to the on-chip RAM.

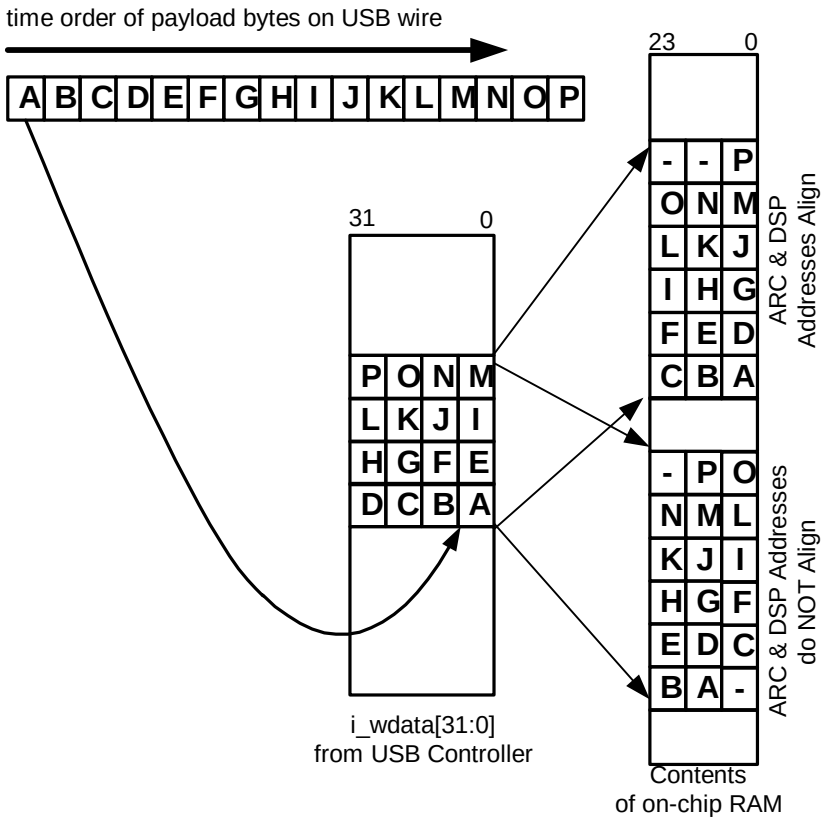


Figure 23. USB 2.0 32 bit to 24 bit Packing Example

The USB DMA interface has to handle data transfers to and from a 32 bit interface to the ARC USB Device Controller. It has to map these data transfers to and from a 24-bit on-chip RAM DMA bus. Figure 23, above shows one example of a stream of bytes coming across the USB cable, being DMAed across the 32-bit BVCI initiator bus and from there being packed into the 24 bit wide on-chip memory. The DMA interface data flow that performs this packing is shown below, see Figure 24. “USB 2.0 DMA Data Flow” on page 65.

From these figures one sees that all DMA address pointers are referenced to the 32 bit data bus world of the ARC USB DMA engine. When the data lands in on-chip RAM or is taken from on-chip RAM, then software has to access it with full knowledge of how the USB DMA transferred the bytes. The figure above shows that depending on where the starting address for the data is assigned in the ARC address space determines how the data aligns with 24 bit DSP RAM. In particular, one divides the ARC byte address by three to obtain a 24 bit word address in the DSP RAM. The remainder of this division determines the alignment in the DSP RAM. A remainder of zero implies a perfectly aligned transfer point. A remainder of one yields the alignment shown in the lower right hand portion of Figure 23. Software must be constantly aware of this alignment and must deal with it as it arises.

For example, the eight end points implemented in the USB Device controller require 16 Queue Head structures to be accessed by the DMA from on-chip RAM. Each queue head is 64 bytes long or 21 remainder 1 DSP words in length. The remainder of one causes the alignment each successive queue head to shift by one byte.

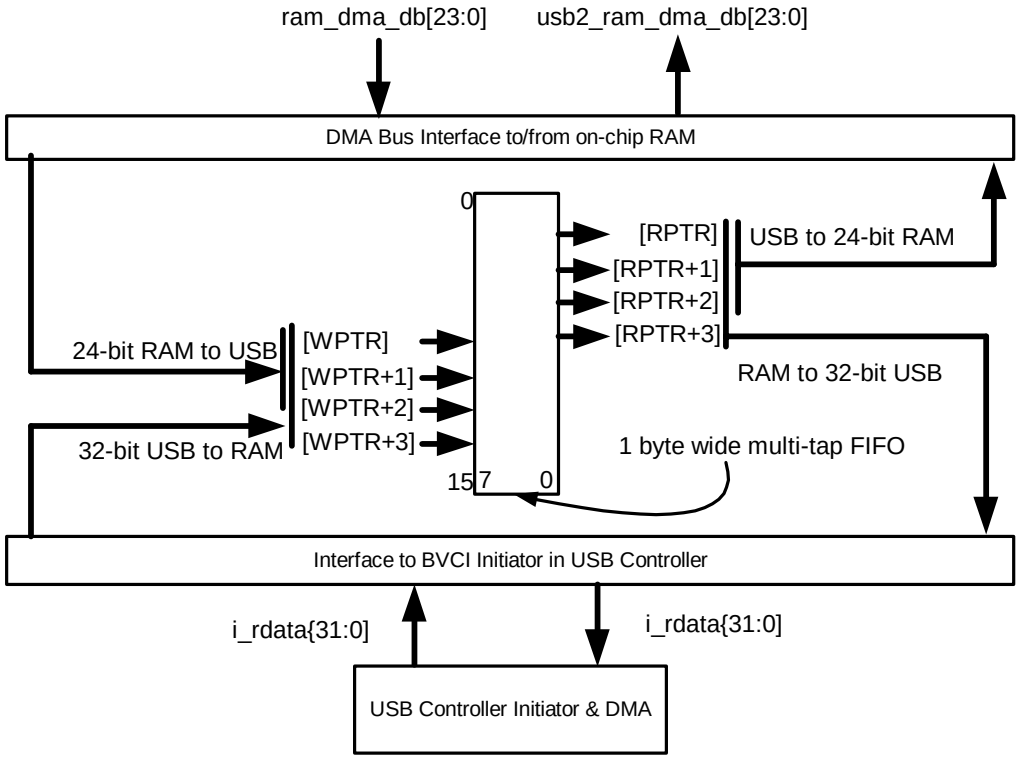


Figure 24. USB 2.0 DMA Data Flow

The general strategy used by software is have pack and unpack routines to access data structures shared with the USB controller, i.e. queue heads, transfer descriptors. In addition, software uses pack and unpack routines to access the packet headers, etc.

8.3. USB UTMI Interface

The SigmaTel developed UTMI interface logic allows multiplexing between the integrated USB 2.0 High Speed PHY and an off-chip UTMI compliant USB 2.0 PHY, see Figure 28. “USB 2.0 PHY at the System on Chip Level” on page 78. The **HW_USBCSR_UTMI_EXT** bit selects the external PHY when set to one.

ADD MATERIAL HERE ABOUT UTMI TEST MODE

8.4. USB Device Controller Core

The USB Device Controller is an instantiation of the ARC VUSBHS-DEV High Speed USB Device Controller Core. This proprietary core, the intellectual property it represents and the copyrighted documentation for the core are the property of ARC International. For detailed information about the device controller core, the reader is

referred to the appropriate ARC documentation. The core synthesized as a device controller only and implements eight endpoints, including endpoint zero.

8.5. USB Controller Flowcharts

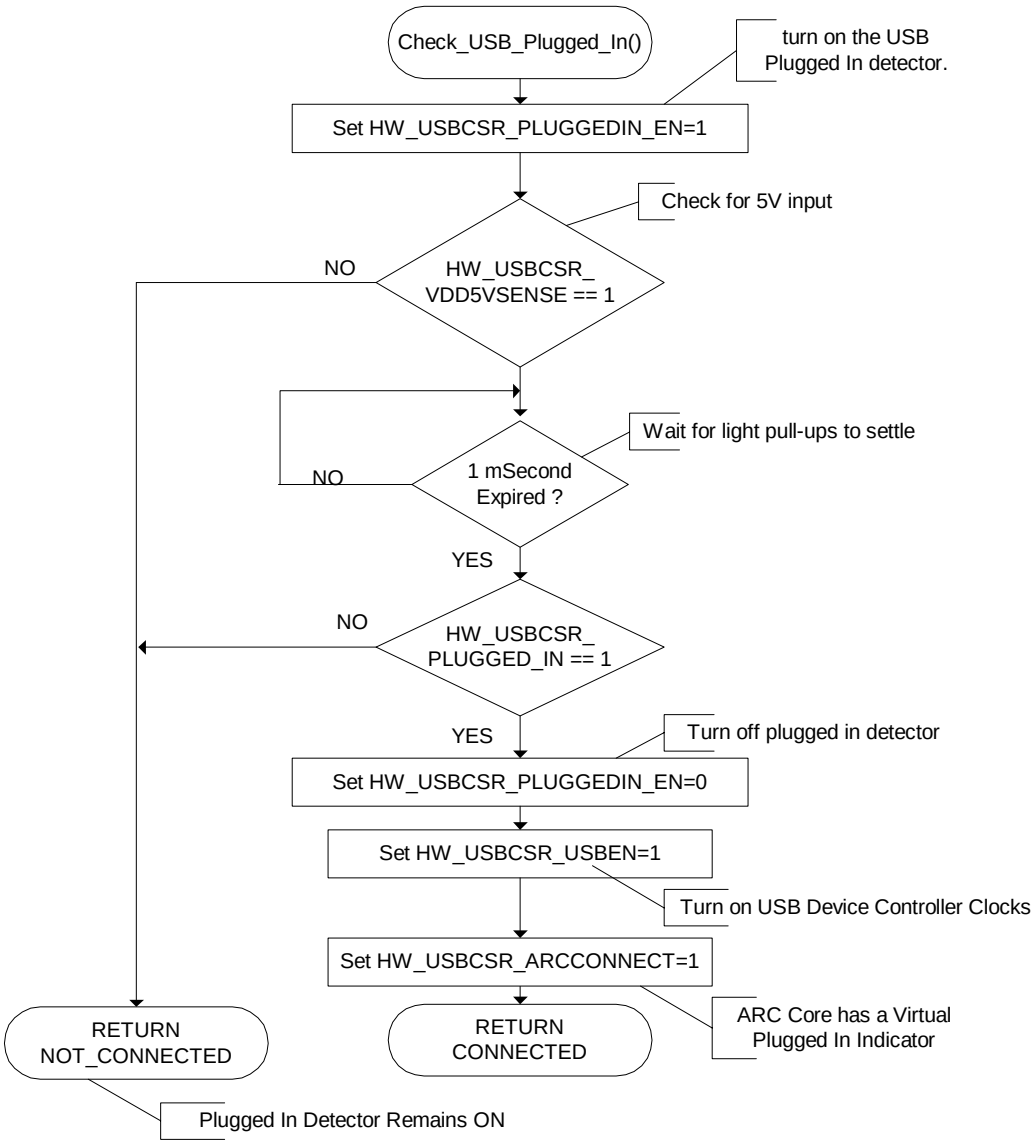


Figure 25. USB 2.0 Check_USB_Plugged_In Flowchart

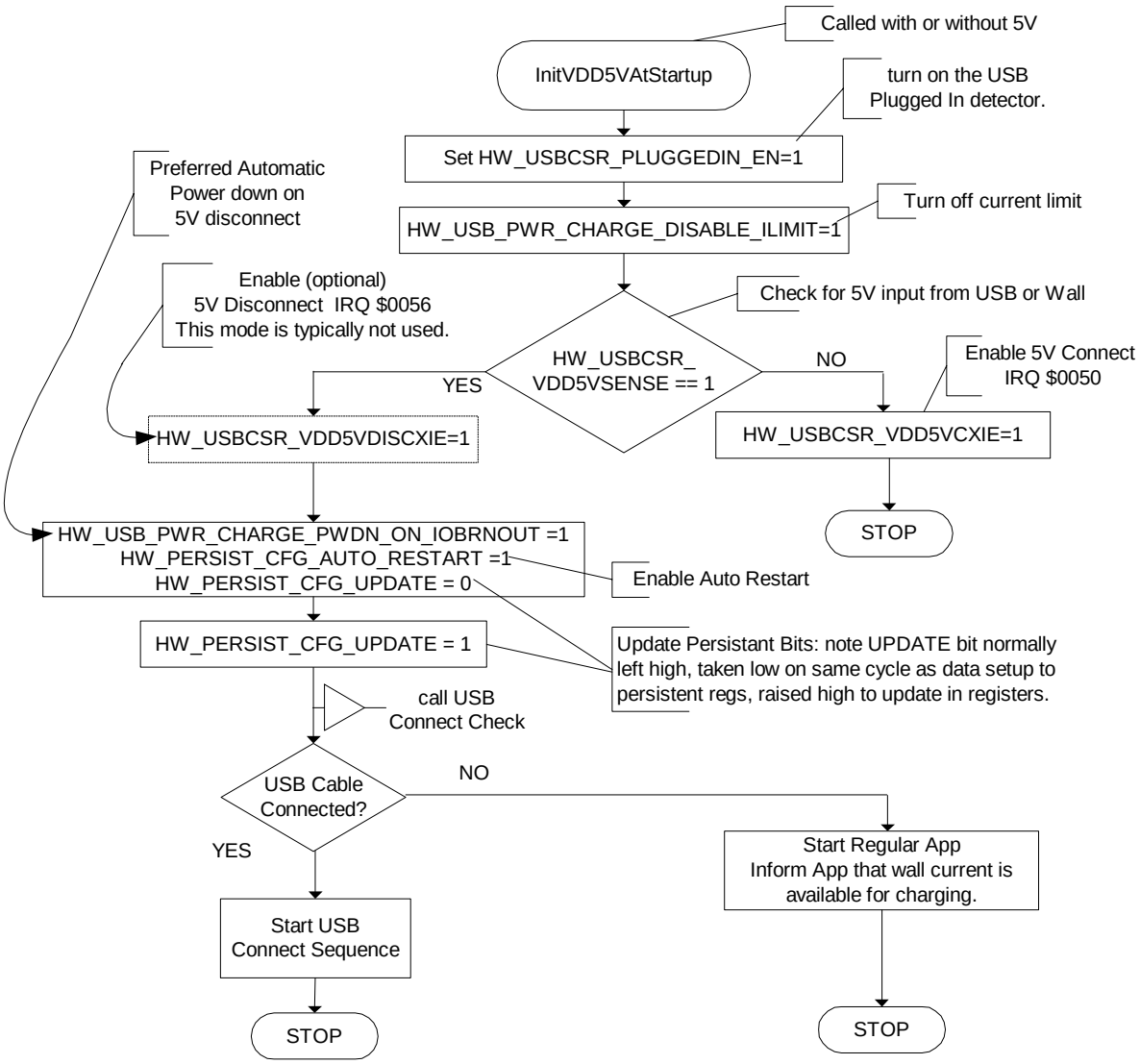


Figure 26. USB 2.0 InitializeVDD5VSense Flowchart

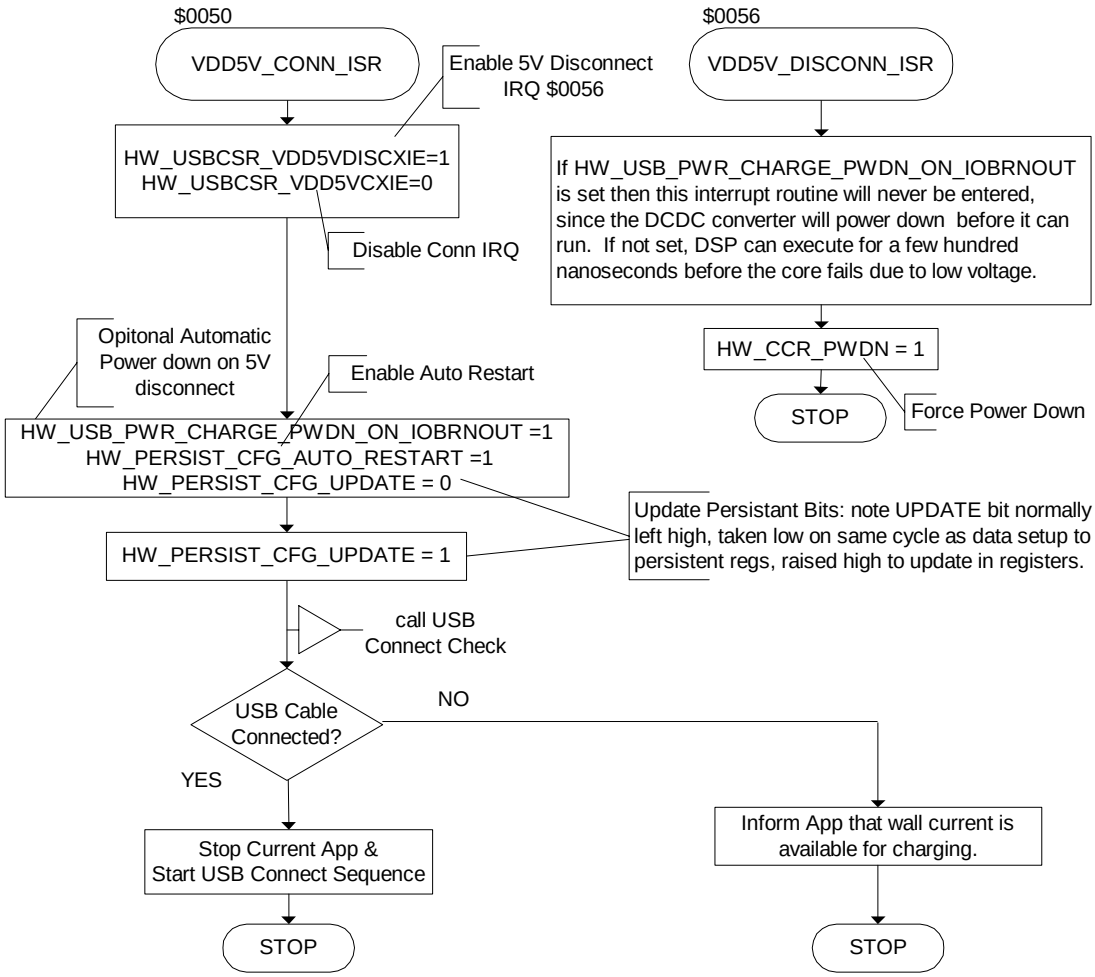


Figure 27. USB 2.0 VDD5V Conn, Disconn ISR Flowchart

8.6. USB Interface Registers

The following subsections describe the programmable registers of the SigmaTel USB Interface Block.

8.6.1. USB Control Status Register

This register handles the overall configuration and control of the USB interface. It provides specific interrupt status and interrupt enables for interrupt events arising from the USB Controller as well as interrupts arising from monitoring the 5V sense comparator and the cable plugged in monitoring circuit in the integrated PHY.

HW_USBCSR X:$F200

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
VDD5VSENSE	HOSTDISCONNECT									PLUGGED_IN	PLUGGEDIN_EN	ARCCONNECT	UTMI_EXT	SUSPF	SUSP	CLKOFF	VDD5VDISCXIE	VDD5VDISCXIRQ	VDD5VCXIE	VDD5VCXIRQ	WAKEUPIE	WAKEUPIRQ	USBEN

Table 62. HW_USBCSR

BITS	LABEL	RW	RESET	DEFINITION
23	**VDD5VSENSE**	R	0	**USB 5Volt Vdd5V Sense State –** This read only bit returns a one when the VDD5V pin is more than one Vt above the VDDIO voltage. This situation indicates that a 5V power source either from a wall transformer or from the USB VBUS has been connected to the STMP35xx. It returns a zero otherwise. ***ADD CROSS REF TO POWER CHARGER***
22	**HOSTDISCONNECT**	R	0	**Host Disconnect –** This disconnect signal reports the state of the Hub Disconnect detector in the integrated PHY.
21:14	**RSRVD**	R	0	**Reserved –** Must be written with 0.
13	**PLUGGED_IN**	R	0	**Cable Plugged In –** This read only bit returns a zero when the integrated PHY's plugged-in detector reports that both D_P and D_N are high. This can only occur when the cable is unplugged and the 200KΩ integrated pull-ups are enabled onto D_P and D_N. This detector is powered down at reset. Power control is located in bit **HW_USBCSR_PLUGGEDIN_EN** and must be turned on for at least 1mSecond prior to relying on the state of this bit.
12	**PLUGGEDIN_EN**	RW	0	**USB plugged-in detector Enable –** Set this bit to one to power up the plugged in detector and to switch in the two 200KΩ pull-up resistors.
11	**ARC_CONNECT**	RW	0	**ARC Connect –** The ARC core has an input that is designed to be hooked up to a VBUS Sensor to tell it when to go to powered state. In this implementation, the DSP detects this situation using the VDD5VSENSE bit above. The DSP therefore virtualizes this indication by controlling the ARCCONNECT bit. When the DSP sets this bit to one, the ARC core moves its internal state machine from the "detached" state to the "powered" state.
10	**UTMI_EXT**	RW	0	**UTMI EXTERNAL Interface Enable –** When set to one, this bit causes the external USB 2.0 PHY to be selected instead of the integrated PHY. Set this bit to zero for normal operation.
9	**SUSPF**	RW	0	**Force USB Suspend mode –** This bit controls a mux which determines the drive source for the utmi_suspendm signal. When set to one, the mux select the SUSP bit below as the source for driving utmi_suspendm. Set this bit to zero for normal operation.
8	**SUSP**	RW	0	**USB Suspend –** This bit provides direct programmable control of the utmi_suspendm signal when enabled by the SUSPF bit above.

Table 63. USB Control Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
7	**CLKOFF**	RW	1	**USB clock off –** When set to one, clocks to the USB module will be set to zero. Set CLKOFF to zero for normal operation of the USB interface.
6	**VDD5VDISCXIE**	RW	0	**USB 5V Vdd5V pin Disconnect Interrupt Enable–** When this bit is set to one, then **VDD5VDISCXIRQ** below will cause an interrupt at vector $0056 when it is set to one.
5	**VDD5VDISCXIRQ**	RW	0	**USB 5V Vdd5V Disconnect Interrupt –** This bit is set to one whenever a falling edge is detected on the **VDD5SENSE** comparator. This bit can be reset by writing a one directly to it.
4	**VDD5VCXIE**	RW	0	**USB 5V Vdd5V Connect Interrupt Enable–** When this bit is set to one, then **VDD5VCXIRQ** below will cause an interrupt at vector $0050 when it is set to one.
3	**VDD5VCXIRQ**	RW	0	**USB 5V Vdd5V Connect Interrupt –** This bit is set to one whenever a rising edge is detected on the **VDD5SENSE** comparator. This bit can be reset by writing a one directly to it.
2	**WAKEUPIE**	RW	0	**Wakeup Interrupt Enable –** When this bit is set to one, then **WAKEUPIRQ** below will cause an interrupt at vector $0054 when it is set to one.
1	**WAKEUPIRQ**	RW	0	**Wakeup Interrupt –** This bit is set to one whenever the synchronized versions of utmi_linestate indicate a K-state on the line, i.e linestate[0] == D_P ==0 & linestate[1] == D_N ==1. This bit can be reset by writing a one directly to it. Activity on. NOTE: this interrupt bit monitors the raw state of the USB signals and will continue to be set as transitions occur on the USB. Software should reset **WAKEUPIE** and then clear **WAKEUPIRQ** as soon as this interrupt is detected.
0	**USBEN**	RW	0	**USB Enable Bit –** The USBEN bit enables the USB port. This bit must be set before any other USB registers are written to.

Table 63. USB Control Status Register Description (Continued)

8.6.2. USB DMA Offset Register

All ARC core DMA addresses have an offset added to them. This addition takes place after the 32-bit ARC byte address is divided by three to obtain a DSP 24-bit word address. USB DMA buffers can be placed in any *one* of the three processor memory spaces (P,X,Y).

HW_USBDMAOFF X:$F201

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
						MEM		ADD															

Table 64. HW_USBDMAOFF

BITS	LABEL	RW	RESET	DEFINITION
23:18	**RSRVD**	R	0	**Reserved –** Must be written with 0.
17:16	**MEM**	RW	0	**MEM –** Selects the memory space (P,X or Y) to which or from which all DMA transfers occur.
15:0	**ADD**	RW	0	**USB DMA ADDRESS OFFSET –** All DMA addresses generated in the USB controller have this 16 bit offset added to them, after the division by 3.

Table 65. USB DMA Offset Address Register Description

8.6.3. USB ARC ACCESS

This register is used in programmed I/O (PIO) access to ARC core internal registers, see 8.1. “USB Programmed I/O (PIO) Target Interface” on page 61.

HW_USBARCACCESS X:$F202

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
KICK							RWB								ADD								

Table 66. HW_USBARCACCESS

BITS	LABEL	RW	RESET	DEFINITION
23	**KICK**	RW	0	**KICK –** This bit is set to a one to initiate either a read or write programmed I/O cycle to the ARC Core BVCI bus target. This bit will be automatically reset when the transfer is complete. Software must not read or modify the data or address registers until the KICK bit has been reset by hardware.
22:17	**RSRVD**	R	0	**Reserved –** Must be written with 0.
16	**RWB**	RW	0	**Read Write Bit –** This bit determines whether a programmed I/O cycle to the ARC core will be a read or a write transfer. When this bit is a one, an ARC core PIO register will be read into the data registers. When this bit is a zero, the data registers will be written to an ARC core PIO register. **RWB** = 1 =READ **RWB** = 0 =WRITE
15:9	**RSRVD**	R	0	**Reserved –** Must be written with 0.
8:0	**ADD**	RW	0	**ARC Register Address –** The ARC core target interface receives **ADD**[8:0] as an address for its internal programmed I/O registers.

Table 67. USB ARC Access Register Description

8.6.4. USB ARC Data Low Register

All programmed I/O data transfers to and from the ARC core’s internal registers are 32 bits wide. This register holds the lower order 16 data bits for such transfers, see 8.1. “USB Programmed I/O (PIO) Target Interface” on page 61.

HW_USBARCDATALOW X:$F203

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								DATA															

Table 68. HW_USBARCDATALOW

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**DATA**	RW	0	**DATA –** The lower 16 bits of a data read/write operation to an ARC internal register.

Table 69. USB ARC Data Low Register Description

8.6.5. USB ARC Data High Register

All programmed I/O data transfers to and from the ARC core's internal registers are 32 bits wide. This register holds the higher order 16 data bits for such transfers, see 8.1. "USB Programmed I/O (PIO) Target Interface" on page 61.

HW_USBARCDATAHIGH X:$F204

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								DATA															

Table 70. HW_USBARCDATAHIGH

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**DATA**	RW	0	**DATA –** The higher 16 bits of a data read/write operation to an ARC internal register.

Table 71. USB ARC Data High Register Description

8.6.6. USB UTMI Test Control Status Register

HW_USBUTCSR X:$F205

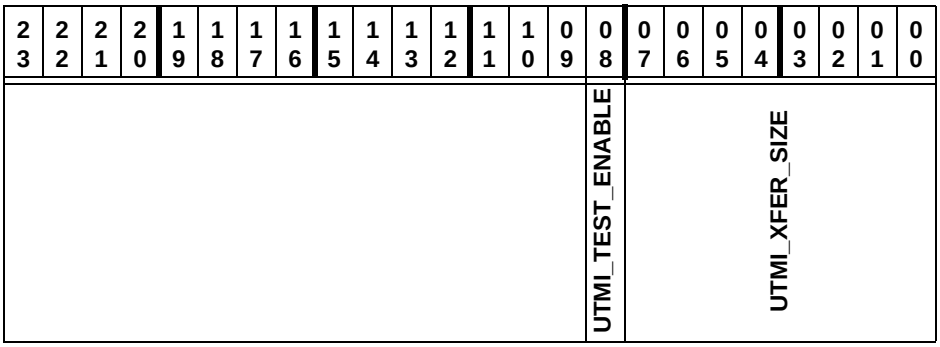


Table 72. HW_USBUTCSR

BITS	LABEL	RW	RESET	DEFINITION
23:9	**RSRVD**	R	0	**Reserved –** Must be written with 0.
8	**UTMI_TEST_ENABLE**	RW	0	**UTMI TEST ENABLE –**
7:0	**UTMI_XFER_SIZE**	RW	0	**UTMI TRANSFER SIZE –** Load this field with the number of DMA transfers to make during UTMI loop back test mode.

Table 73. USB UTMI Test Control Status Register Description

8.6.7. USB UTMI1 Register

HW_USBUTMI1 X:$F206

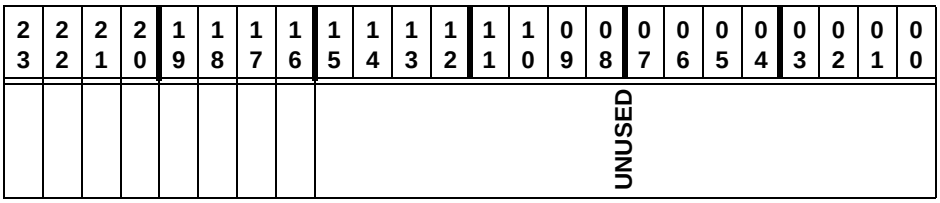


Table 74. HW_USBUTMI1

	LABEL	RW	RESET	DEFINITION
20:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**UNUSED**	RW	0	**UNUSED Bit –** This field is currently unused in the design.

Table 75. USB UTMI1 Register Description

8.6.8. USB UTMI2 Register

HW_USBUTMI2 X:$F207

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								UNUSED															

Table 76. HW_USBUTMI2

BITS	LABEL	RW	RESET	DEFINITION
20:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**UNUSED**	RW	0	**UNUSED Bit –** This field is currently unused in the design.

Table 77. USB UTMI2 Register Description

8.6.9. USB UTMI Sense Register

This register provides hardware debug and design verification access to various signals on the internal UTMI interface.

HW_USBUTMISENSE X:$F208

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								UTMI_RESET	UTMI_XCVR_SELECT	UTMI_TERM_SELECT	UTMI_LINESTATE		UTMI_OPMODE		UTMI_TXVALIDH	UTMI_TXVALID	UTMI_TXREADY	UTMI_RXVALIDH	UTMI_RXVALID	UTMI_RXACTIVE	UTMI_RXERROR	UTMI_DATAOE	UTMI_DATABUS16_8

Table 78. HW_USBUTMISENSE

BITS	LABEL	RW	RESET	DEFINITION
20:16	**RSRVD**	R		**Reserved –** Must be written with 0.
15	**UTMI_RESET**	R		**UTMI Reset Signal –** T
14	**UTMI_XCVR_SELECT**	R		**UTMI High Speed Transceiver Select –** T
13	**UTMI_TERM_SELECT**	R		**UTMI High Speed Terminator Select –** T
12:11	**UTMI_LINESTATE**	R		**UTMI Linestate[1:0] –** T
10:9	**UTMI_OPMODE**	R		**UTMI Opmode[1:0] –** T
8	**UTMI_TXVALIDH**	R		**UTMI Transmit Valid High –** T
7	**UTMI_TXVALID**	R		**UTMI Transmit Valid Low –** T
6	**UTMI_TXREADY**	R		**UTMI Transmit Ready –** T
5	**UTMI_RXVALIDH**	R		**UTMI Receive Valid High –** T
4	**UTMI_RXVALID**	R		**UTMI Receive Valid low –** T
3	**UTMI_RXACTIVE**	R		**UTMI Receive Active –**
2	**UTMI_RXERROR**	R		**UTMI Receive Error –**
1	**UTMI_DATAOE**	R		**UTMI Data Output Enable –**
0	**UTMI_DATABUS16_8**	R		**UTMI Data Bus 16 or 8 –** The USBEN bit enables the USB port. This bit must be set before any other USB registers are written to.

Table 79. USB UTMI Sense Register Description

8.6.10. USB Read Test Register

HW_USBREADTEST X:$F209

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TESTVALUE																							

Table 80. HW_USBREADTEST

	LABEL	RW	RESET	DEFINITION
15:0	**TESTVALUE**	R	$ABCDEF	**Read Only Test Value –** This field always reads back the same constant value $ABCDEF

Table 81. USB UTMI1 Register Description

8.6.11. USB State Machine Sense Register

This register provides hardware debug and design verification access to various state machines in the ARC core interface.

HW_USBSTATEMACHINES X:$F20A

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
				DMA_STATE									IRESP_STATE						CMD_STATE				

Table 82. HW_USBSTATEMACHINES

BITS	LABEL	RW	RESET	DEFINITION
20:18	**RSRVD**	R		**Reserved –** Must be written with 0.
17:12	**DMA_STATE**	R		**DMA State Machine State**
11	**RSRVD**	R		**Reserved –** Must be written with 0.
10:8	**IRESP_STATE**	R		**Interrupt Response State Machine State**
7:5	**RSRVD**	R		**Reserved –** Must be written with 0.
4:0	**CMD_STATE**	R		**PIO Command State Machine State**

Table 83. USB State Machine Sense Register Description

8.6.12. USB ARC Unused Signals Sense Register

This register provides hardware debug and design verification access to various unused signals of the ARC core.

HW_USBARCUNUSED X:$F20B

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
													SERIAL_SIGS						PHILIPS_SIGS				

Table 84. HW_USBARCUNUSED

BITS	LABEL	RW	RESET	DEFINITION
20:11	**RSRVD**	R		**Reserved –** Must be written with 0.
10:5	**SERIAL_SIGS**	R		**Unconnected Serial PHY Interface Signals**
4:0	**PHILIPS_SIGS**	R		**Unconnected Philips PHY Interface Signals**

Table 85. USB ARC UNUSED Signal Sense Register Description

8.6.13. USB Laser Fuse Sense Register

This register provides hardware debug and design verification access to various unused signals of the ARC core.

HW_USBLASERFUSE X:$F20C

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		LASER_FUSE							ARC_UNCONNECTED														

Table 86. HW_USBSTATEMACHINES

BITS	LABEL	RW	RESET	DEFINITION
20:11	**RSRVD**	R		**Reserved –** Must be written with 0.
21:20	**LASER_FUSE**	R		**USB Laser Fuse –** When LASER_FUSE[1] is a one, then the USB controller core is limited to Full Speed operation only. When LASER_FUSE[0] is a one, then the USB controller clock is gated off and the function is not available.
19:15	**RSRVD**	R		**Reserved –** Must be written with 0.
14:0	**ARC_UNCONNECTED**	R		**Various Unconnected Signals from ARC USB 2.0**

Table 87. USB State Machine Sense Register Description

9. INTEGRATED USB 2.0 PHY (HS,FS)

The STMP35xx contains an integrated USB 2.0 PHY macro-cell capable of connecting to PC host systems at the USB Full Speed (FS) rate of 12Mbits/Second or at the USB 2.0 High Speed (HS) rate of 480Mbits/Second. The integrated PHY provides a standard UTMI interface. This allows the STMP35xx to alternatively connect to an external UTMI compliant USB 2.0 PHY chip.

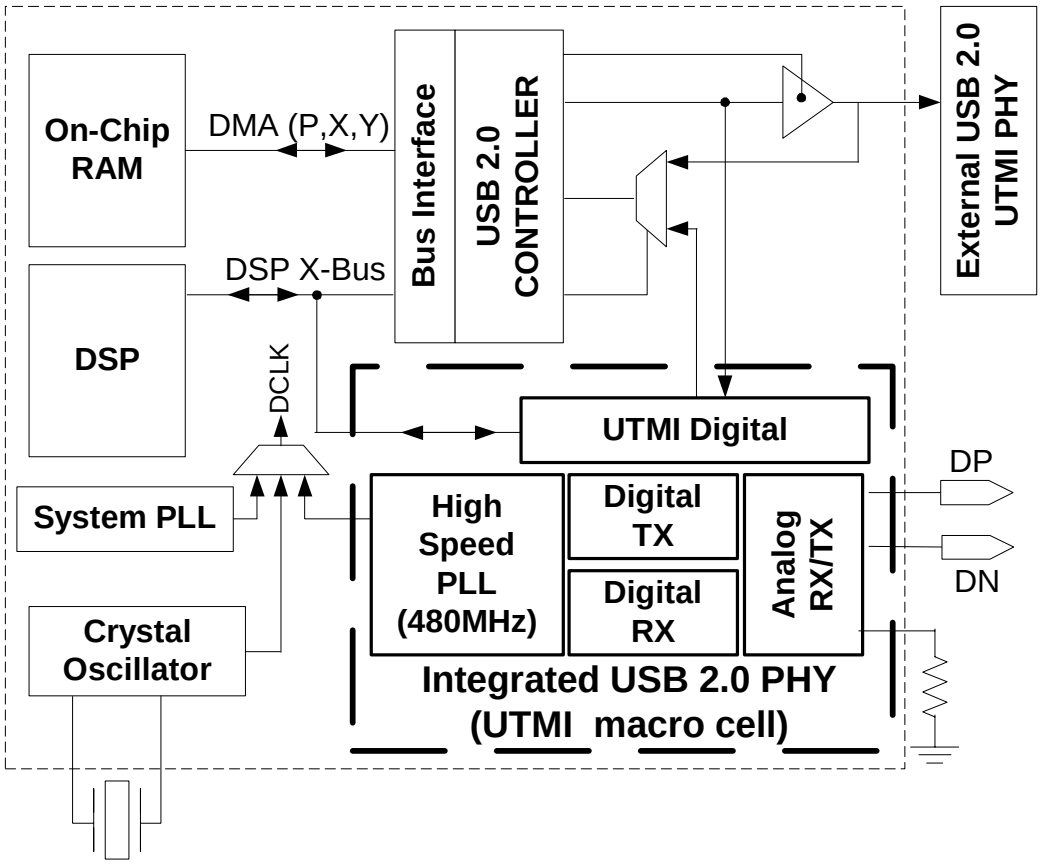


Figure 28. USB 2.0 PHY at the System on Chip Level

The following subsections describe the external interfaces, internal interfaces, major blocks, and programable registers that comprise the integrated USB 2.0 PHY.

9.1. External Signals

DP,DN – These pins connect directly to a USB device connector.

Precision Calibration Resistor. This pin connects a 620Ω +/- 1% resistor to ground. The key on chip resistors, i.e. the 45Ω high speed termination resistor and the 1500Ω pull up resistor contain digitally controlled trimming and calibration circuits to match their impedance to the external precision resistor for USB 2.0 specification compliance.

9.2. UTMI Internal Signals

utmi_clk – Used to clock receive and transmit data to and from the USB controller. The internal UTMI interface is 16 bits wide resulting in a 30MHz UTMI_CLK sourced by the PHY. This clock can either be derived from a divide by sixteen off of the USB 2.0 PHY PLL or can be divided by two from DCLK. see Figure 33. “USB 2.0 PHY PLL Block Diagram” on page 88.

utmi_reset – From chip wide master reset distribution.

utmi_xcvr_select – Selects between the High Speed and Full Speed transceivers (0 = High Speed, 1 = Full Speed).

utmi_term_select – Selects between High Speed and Full Speed terminations. (0 = High Speed Termination, 1 = Full Speed Termination).

utmi_suspend – Not implemented in the internal UTMI interface. The availability of PHY control registers that are directly accessible by DSP instructions obviates the need for this signal.

utmi_line_state[1:0] – These signals directly reflect the current state of the single ended receivers (DP and DN) in the PHY.

DN LS[1]	DP LS[0]	UTMI_LINE_STATE[1:0]
0	0	00 = SE0, single ended zero
0	1	01 = ‘j’ State
1	0	10 = ‘K’ State
1	1	11 = SE1, single ended one

Table 88. USB PHY line_state[1:0]l

utmi_op_mode[1:0] – These input signals tell the PHY whether to encode and decode transmissions with bit stuffing and NRZI or to simply pass the data straight through.

OP_MODE	UTMI_OP_MODE[1:0]
00	Normal operation
01	Non-driving
10	Disable bit stuffing and NRZI encoding
11	Reserved

Table 89. USB PHY op_ode[1:0]l

utmi_tx_data[15:0] – These PHY inputs convey the transmit data from the USB controller to the PHY.

utmi_tx_valid – This signal indicates when tx_data[7:0] has a valid data output byte.

utmi_tx_validh – This signal indicates when tx_data[15:0] has a valid data output byte.

utmi_tx_ready – The USB controller uses this signal in conjunction with the current state of tx_valid to determine when a data cycle has been accepted into the PHY serializer. When utmi_tx_rdy and utmi_tx_valid are both asserted at the rising edge of utmi_clk then a new data word should be presented by the USB controller to the PHY.

utmi_rx_data[16:0] – These signals convey the received data from the PHY to the USB controller.

utmi_rx_valid – When high, this signal tells the USB controller that utmi_rx_data[7:0] contains a valid byte of data.

utmi_rx_validh – When high, this signal tells the USB controller that utmi_rx_data[15:8] contains a valid byte of data.

utmi_rx_active – Indicates that the USB PHY receive machine has detected SYNC and is active.

utmi_rx_error – Zero indicates no error detected. One indicates that a receive error has been detected.

usb_plugged_in_detect – Digital output from the receiver's analog plugged-in detector circuit. This signal is captured within the USB controller and made available from there to the DSP.

9.3. UTMI and Digital Circuits

The UTMI provides a 16 bit interface to the USB Controller. This interface is clocked at 30MHz. There are four parts to the UTMI/Digital circuits block. The UTMI block, the Digital Transmitter, the Digital Receiver and the Programmable Registers block.

9.3.1. UTMI Block

This block handles the line_state bits, reset buffering, suspend distribution, transceiver speed selection, and transceiver termination selection. The PLL supplies a 60MHz signal to all of the digital logic. The UTMI block does a final divide by two to develop the 30MHz clock used in the interface.

9.3.2. Digital Transmitter Block

The digital transmitter block receives the 16 bit transmit data from the USB controller, and handles the tx_valid, tx_validh and tx_ready handshake. In addition, it contains the transmit serializer which converts the 16 bit parallel words at 30MHz to a single bit stream at 480Mbit for High Speed or 12Mbit for Full Speed. It does this while implementing the bit stuffing algorithm and the NRZI encoder that are used to remove the DC component from the serial bit stream. The output of this encoder is sent to the Full Speed (FS) or High Speed (HS) drivers in the analog transceiver section's transmitter block.

9.3.3. Digital Receiver Block

The digital receiver block receives the raw serial bit stream either from the HS differential transceiver or from the FS differential transceiver. The HS input goes to a very fast DLL which uses one of eight identically spaced phases of the 480MHz clock to pick a sample point. As the phase of the USB host transmitter shifts relative to the local PLL, the receiver section's HS DLL tracks these changes to give a reliable sample of the incoming 480Mbit/Second bit stream. Since this sample point shifts relative to the PLL phase used by the digital logic, a rate matching elastic buffer is provided to cross this clock domain boundary. Once the bit stream is in the local clock domain, an NRZI decoder and Bit Unstuffer restores the original payload data bit stream and passes it to a de-serializer and holding register. The Receive state machine handles the rx_valid, rx_validh and handshake with the USB controller. The handshake is not interlocked in that there is no rx_ready signal coming from the controller. The controller must take each 16 bit value as presented by the PHY. The

Receive state machine provides an rx_active signal to the controller that indicates when it is inside a valid packet (SYNC detected, etc.).

9.3.4. *Programmable Registers Block*

The PHY contains four 24 bit programmable registers that are read and written via the DSP's X bus, see 9.7. "Programmable Registers" on page 92.

9.4. Analog Transceiver

The analog transceiver section comprises an analog receiver and an analog transmitter, see Figure 29 below.

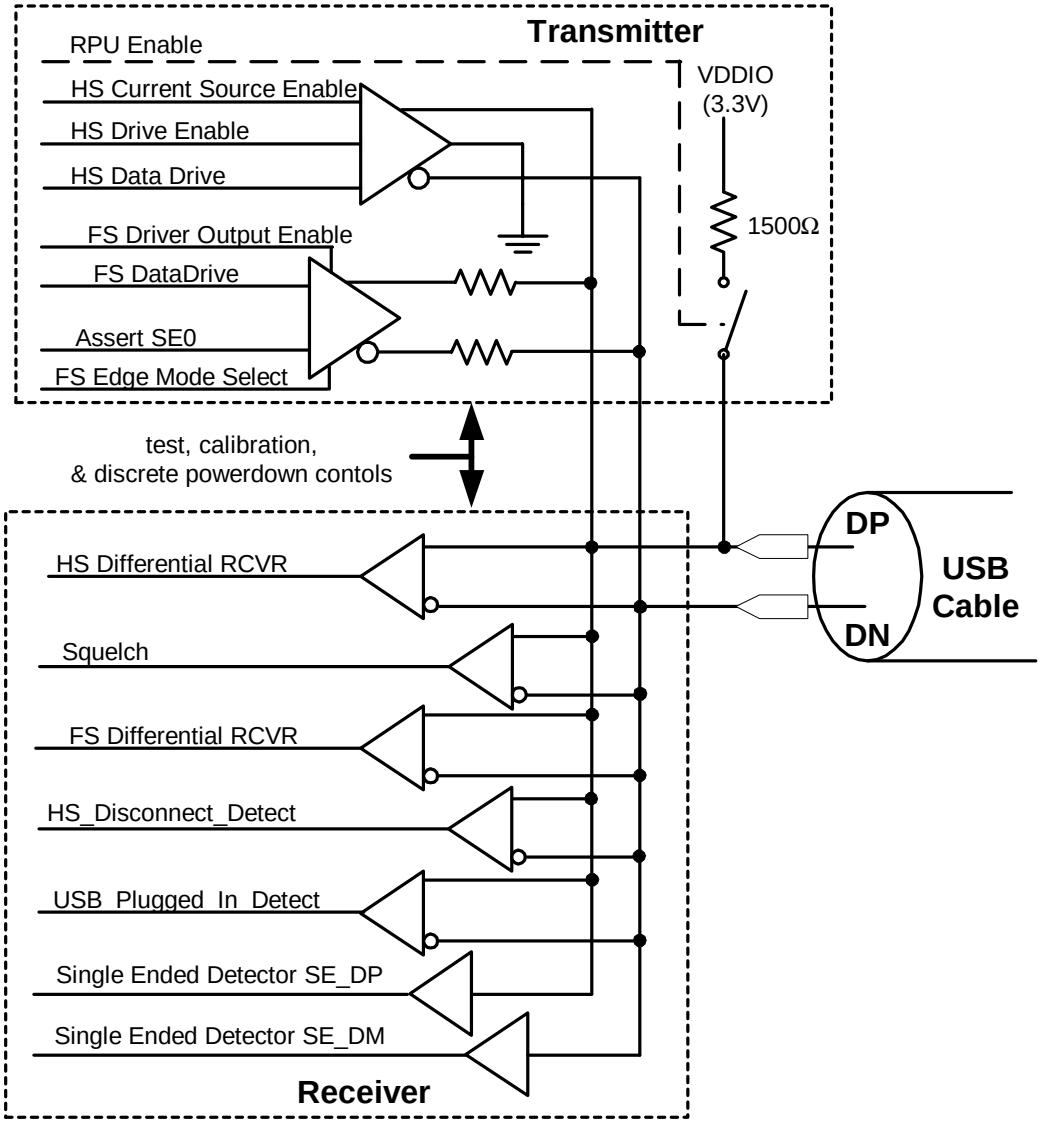


Figure 29. USB 2.0 PHY Analog Transceiver Block Diagram

9.4.1. Analog Receiver

The analog receiver comprises five differential receivers and two single ended receivers, described below.

9.4.1.1. HS Differential Receiver

The high speed differential receiver is both a differential analog receiver and threshold comparator. Its output is a one if the differential signal is greater than a 0V threshold. Its output is zero otherwise. Its purpose is to discriminate the +/- 400mV differential voltage resulting from the high speed drivers current flow into the dual 45Ω terminations found on each leg of the differential pair. The envelope or squelch detector, below, ensures that the differential signal has sufficient magnitude to be valid. The HS differential receiver tolerates up to 500mV of common mode offset.

9.4.1.2. Squelch Detector

The squelch detector is a differential analog receiver and threshold comparator. Its output is a one if the differential magnitude is less than a nominal 100mV threshold. Its output is zero otherwise. Its purpose is to invalidate the HS differential receiver when the incoming signal is simply to low to receive reliably.

9.4.1.3. FS Differential Receiver

The full speed differential receiver is both a differential analog receiver and threshold comparator. The crossover voltage falls between 1.3V and 2.0V. Its output is a one when the D_P line is above the crossover point and the D_N line is below the crossover point.

9.4.1.4. HS Disconnect detector

This Host side function is not used in STMP35xx applications but is included to make a complete UTMI macro-cell. It is a differential analog receiver and threshold comparator. Its output is a one if the differential magnitude is greater than a nominal 575mV threshold. Its output is zero otherwise.

9.4.1.5. USB Plugged-In Detector

The USB Plugged-In detector looks for both D_P and D_N to be high. There is a pair of large on-chip pull-up resistors (200KΩ) that hold both D_P and D_N high when the USB cable not attached. The USB Plugged-In detector signals a zero in this case.

The host/hub interface that is *upstream* from the STMP35xx contains a 15KΩ pull-down resistor that easily over-rides the 200KΩ pull-up. When plugged in, at least one signal in the pair will be low which will force the Plugged-In detector's output high.

9.4.1.6. Single Ended D_P Receiver

The single ended D_P receiver output is high whenever the D_P input is above its nominal 1.8V threshold.

9.4.1.7. Single Ended D_N Receiver

The single ended D_N receiver output is high whenever the D_N input is above its nominal 1.8V threshold.

9.4.2. Analog Transmitter

The analog transmitter comprises two differential drivers, one for high speed signaling and one for full speed signalling. It also contains the switchable1500Ω pull up resistor.

9.4.2.1. Switchable High Speed 45Ω Termination Resistors

High speed current mode differential signalling requires good 90Ω differential termination at each end of the USB cable. This results from switching in 45Ω terminating resistors from each signal line to ground at each end of the cable. Since each signal is parallel terminated with 45Ω at each end, each driver sees a 22.5Ω load. This is much too low of a load impedance for full speed signalling levels hence the need for switchable high speed terminating resistors. Switchable trimming resistors are provided to tune the actual termination resistance of each device, see Figure 30. “USB 2.0 PHY Transmitter Block Diagram” on page 84. The HW_USBPHYTX_TXCAL45DP bit field, for example, allows one of 16 trimming resistor values to be placed in parallel with the 45Ω terminator on the D_P signal. The calibration operation is described below, see 9.4.2.5. “Resistor Calibration Mode” on page 85.

9.4.2.2. Full Speed Differential Driver

The full speed differential drivers are essentially “open drain” low impedance pull down devices which are switched in a differential mode for full speed signalling, i.e. either one or the other device is turned on to signal the “J” state or the “K” state. These drivers are both turned on, simultaneously, for high speed signalling. This has the effect of switching in both 45Ω terminating resistors. The tx_fs_hiz signal originates in the digital transmitter section. The hs_term signal which also controls these drivers comes from the UTMI.

9.4.2.3. High Speed Differential Driver

The high speed differential driver receives a 17.78mA current from the constant current source and essentially steers it down either the D_P signal or the D_N signal or alternatively to ground. This current will produce approximately a 400mV drop across the 22.5Ω termination seen by the driver when it is steered onto one of the signal lines. The approximately 17.78mA current source is referenced back to the integrated voltage band gap circuit. The Iref, IBias and V to I circuits are shared with the integrated battery charger.

9.4.2.4. Switchable 1500Ω DP Pull up resistor

The STMP35xx contains a switchable 1500Ω pull-up resistor on the D_P signal. This resistor is switched on to tell the Host/Hub controller that a full speed capable device is on the USB cable, powered on, and ready. This resistor is switched off at power on reset so the host doesn’t recognize a USB device until DSP software enables the announcement of a full speed device. This pull-up also includes 16 switchable parallel trimming resistors, see HW_USBPHYTX_TXCAL1500.

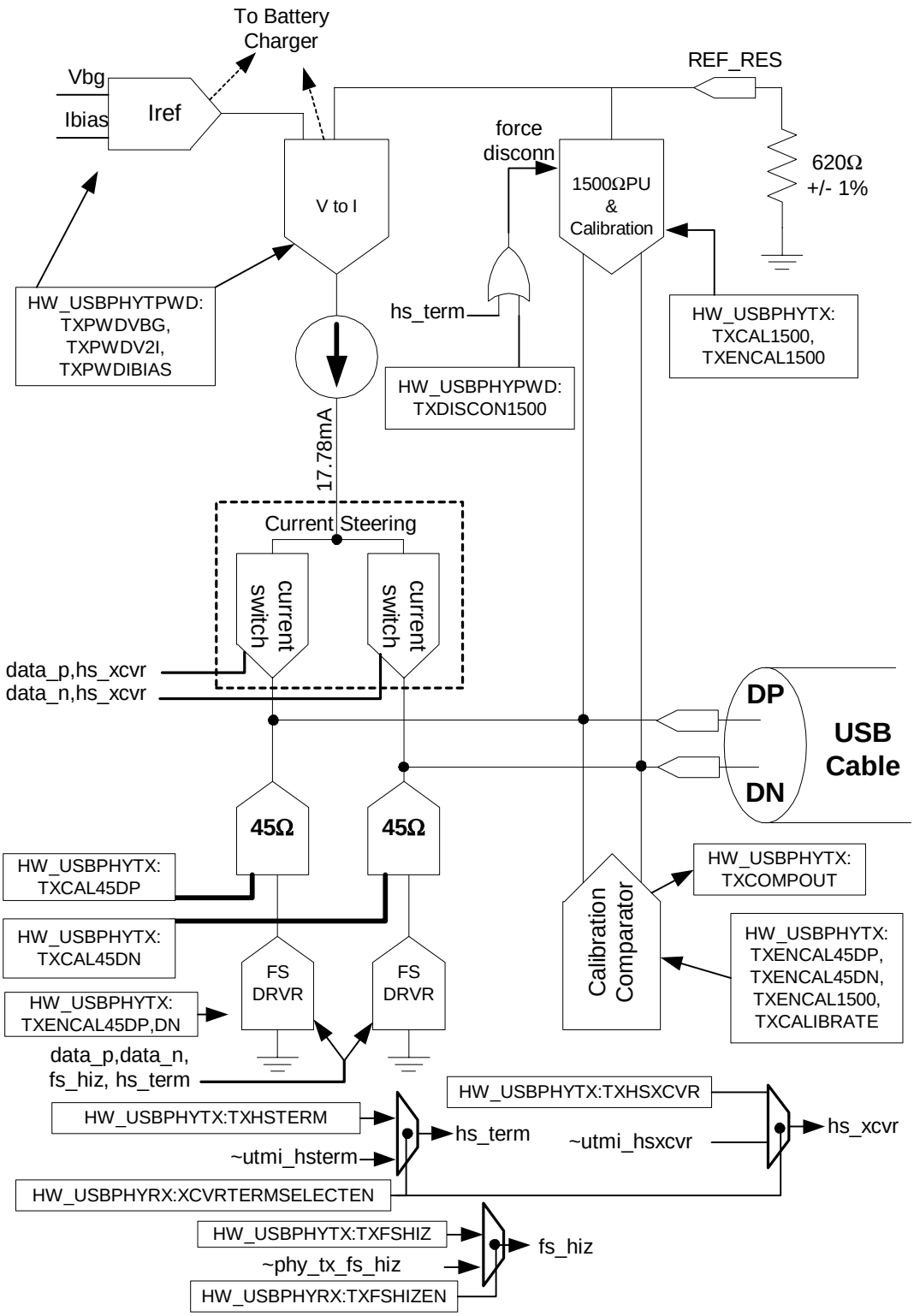


Figure 30. USB 2.0 PHY Transmitter Block Diagram

The following table summarizes the response of the PHY analog transmitter to various states of UTMI input and key transmit/receive state machine states.

UTMI OPMODE	UTMI TERM	UTMI XCVR	T/R	FUNCTION	45 Ω HIZ	1500Ω HIZ	
00=Normal	0	0	X	HS	0	1	
	1	1	T	FS	0	0	
	1	1	R	FS	1	0	**SUSPEND**
	1	0	R	CHIRP	1	0	
	1	0	T	CHIRP	1	0	
	0	1	X	DISCONNECT	1	1	
01=NoDrive	0	0	T	HS	1	1	
	0	0	R	HS	1	1	
	1	1	X	FS	1	1	
	1	0	X	CHIRP	1	1	
	0	1	X	DISCONNECT	1	1	**POR**
10=NoNRZI NoBitStuff	0	0	X	HS	0	1	
	1	1	T	FS	0	0	
	1	1	R	FS	1	0	
	1	0	R	CHIRP	1	0	
	1	0	T	CHIRP	1	0	
	0	1	X	DISCONNECT	1	1	
11= Invalid	0	0	T	HS	1	1	
	0	0	R	HS	1	1	
	1	1	X	FS	1	1	
	1	0	X	CHIRP	1	1	
	0	1	X	DISCONNECT	1	1	

Table 90. USB PHY Terminator States

9.4.2.5. Resistor Calibration Mode

The analog transmitter section includes a calibration comparator that can monitor the D_P or D_N voltage as desired. Setting **HW_USBPHYTX_ENCAL45DP** selects the D_P signal. The comparator output is visible in **HW_USBPHYTX_TXCOMPOUT**. To calibrate the 45Ω D_P terminator, first set the field **HW_USBPHYTX_TXCAL45DP** to all ones ($F). The flowchart of Figure 31, below, shows how to search for the proper trimming resistor to calibrate the D_P terminator.

Essentially one first puts the chip into termination resistor calibration mode for the D_P terminator. One starts with the largest value of trimming select, i.e. all ones. One has to make several precise minimum delay calculations to allow the mixed signal components to stabilize. The comparator output is sampled and then checked. If the comparator has not tripped, then one reduces the value of the trimming select field and tries again. This repeats until the trip point is reached.

While this flowchart shows how to calibrate the D_P terminator, calibration of the D_N terminator is accomplished in a similar manner, substituting *DN bit fields for *DP backfields.

Calibrating the 1500Ω pull-up is done in a similar manner. The flow chart of Figure 32 shows how to accomplish this task.

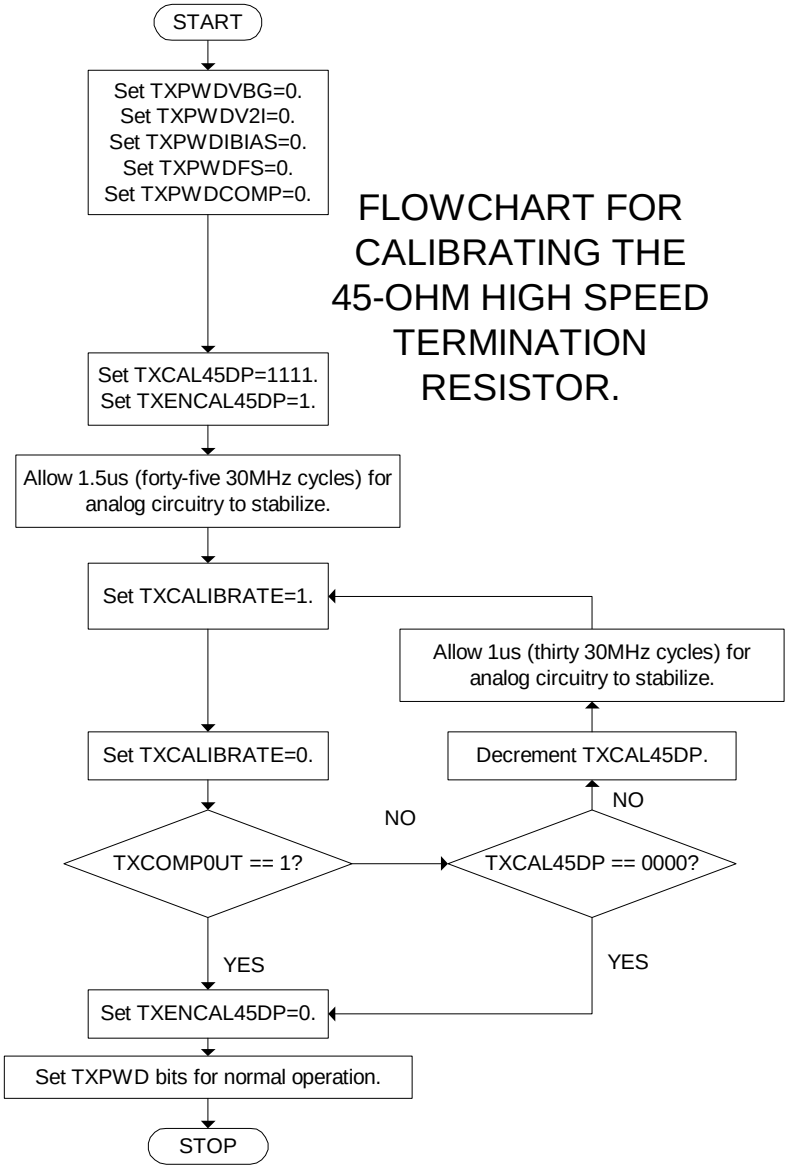


Figure 31. 45Ω Calibration Flow Chart

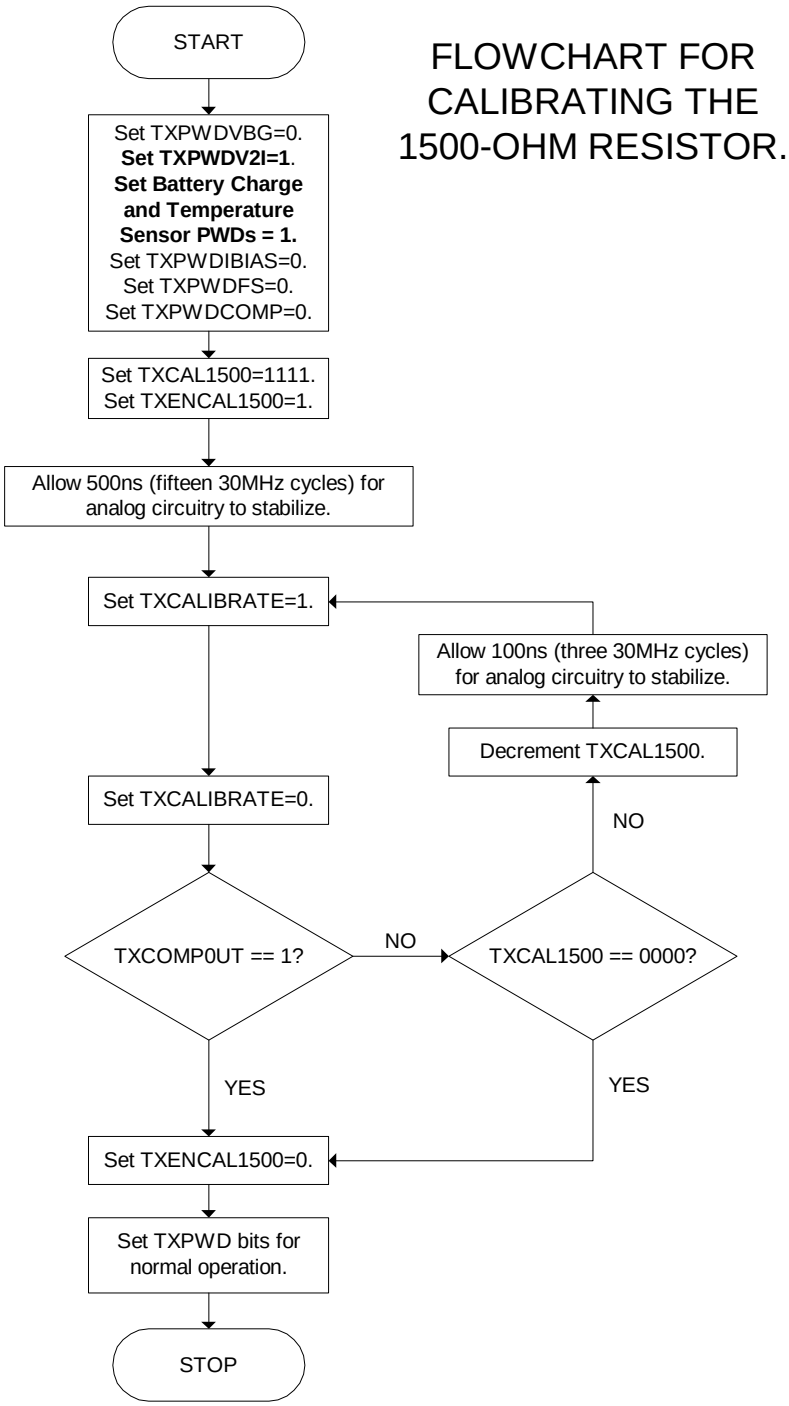


Figure 32. 1500Ω pull up resistor Calibration Flow

9.5. USB 2.0 PHY 480MHz PLL

The STMP35xx includes a 480MHz PLL to clock the high speed transceiver. This PLL can also be used for generating the system wide DCLK. In addition, it can be left powered down and the system wide DCLK can be used to drive 60MHz into the Full Speed PHY to operate the USB.

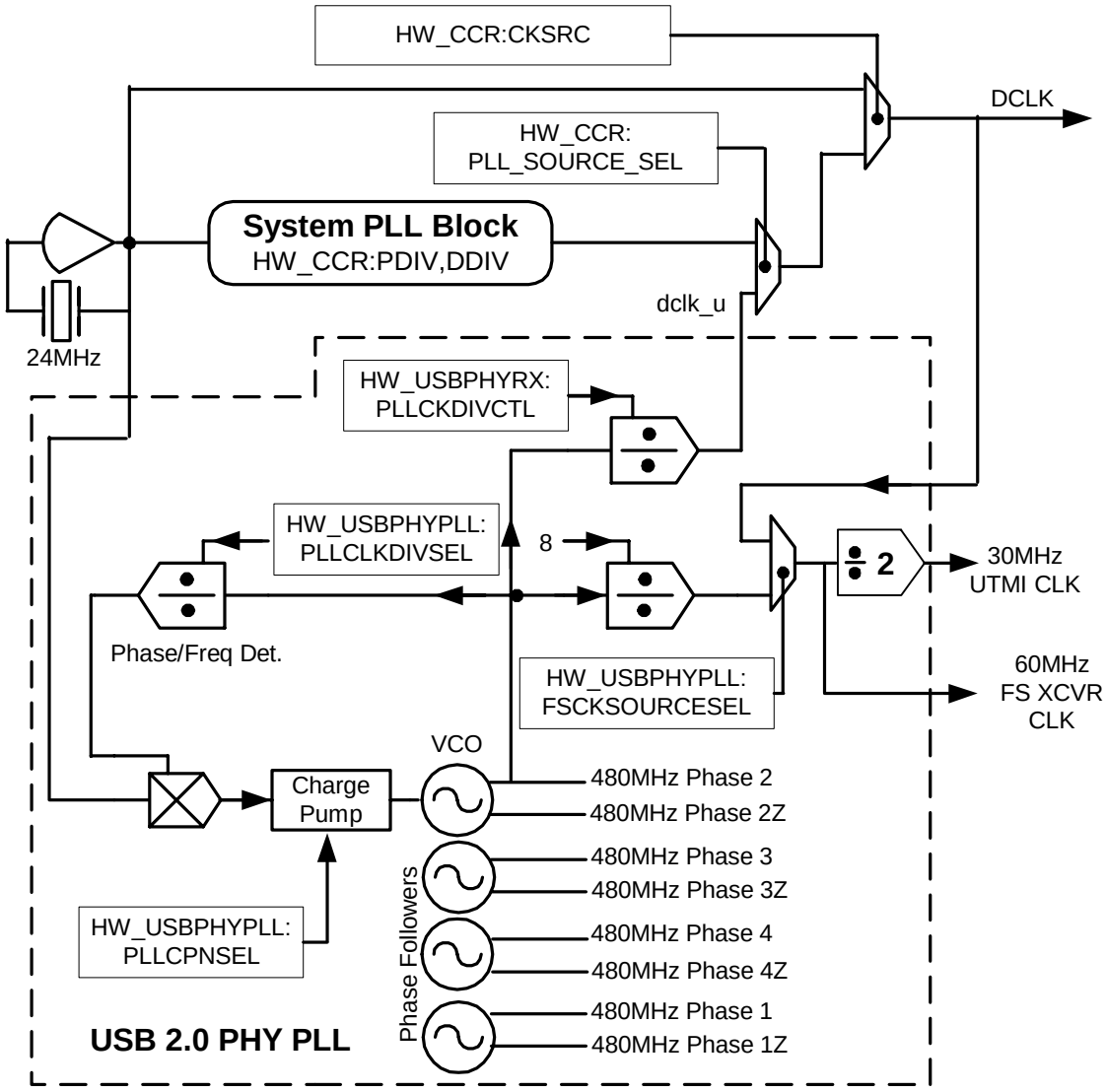


Figure 33. USB 2.0 PHY PLL Block Diagram

9.5.1. 480MHz VCO and Phase Followers

The heart of the PLL is the VCO which can operate from 120MHz to 720MHz. The VCO frequency is determined by the output of the charge pump, in standard fashion. The VCO produces a 480MHz clock and its exact out of phase component. In the design, these are identified as vco_clk2 and vco_clk2z. In addition, three

phase followers are included to produce a precise eight phase clock at 480MHz. These eight phases are used in the high speed digital receiver to operate the DLL that tracks the incoming 480Mbit/second USB receive digital stream. vco_clk2 is also used as a single phase 480MHz digital clock for various clock dividers and other circuits. The VCO and various of its phase followers can be selectively powered down to reduce the overall energy requirements of the STMP35xx.

9.5.2. PFD and Charge Pump

The phase/frequency detector (PFD) and charge pump (CP) are used to lock the VCO to the reference oscillator. For the STMP35xx the reference is the integrated crystal oscillator. The most common reference crystal frequencies are 24MHz and 20MHz. Selective power down and control of the PFD, the CP and various loop filter parameters can be controlled in **HW_USBPLL**. The charge pump gain (current) should be adjusted for different feedback settings, see **HW_USBPLL_PLLCPNSEL**.

9.5.3. Feedback Divider

The feedback divider is connected to dived the VCO frequency by one of several constant dividers. The loop goal is to lock to 480MHz from several different reference frequencies. If a 24MHz crystal is used with a divide by 20, then the loop will stabilize to 480MHz. Similarly if a 20MHz crystal is used with a divide by 24, then the loop will stabilize again at 480MHz. With a 20MHz crystal and a divide by 6 the loop will stabilize to 120MHz. Of course the charge pump gain must be set appropriately. The feedback divider is programmed in **HW_USBPLL_PLLCLKDIVSEL**.

9.5.4. DCLK_U Generation

The high speed PLL supplies a post divider output of the VCO that can be used to drive the DCLK chip wide clock net. The divider value is set in **HW_USBRX_PLLCKDIVCTL** which defaults to a divide by eight. With the VCO locked to 480MHz, this produces a 60MHz clock which is driven out on the signal dclk_u. Dclk_u is wired to the system PLL where a 2:1 mux selects either the system PLL or the high speed PLL to drive DCLK. The **HW_CCR_PLLSOURCESEL** bit selects the desired PLL. Of course, the **HW_CCR_CKSRC** bit selects either the crystal oscillator or the selected PLL to drive the DCLK clock net.

9.5.5. 60MHz Full Speed USB PHY Clock Generation

A fixed divide by eight post divider is included to generate 60MHz from a 480MHz VCO. This clock is used within the full speed USB transceiver. It is also further divided by two to produce the 30MHz clock used in the UTMI interface for the integrated USB 2.0 PHY. For test purposes, the chip wide DCLK can be selected instead of this high speed PLL version of the 60MHz signal, see **HW_USBPHYPLL_FSCKSOURCESEL**.

9.6. Integrated USB 2.0 PHY Initialization Flow Charts

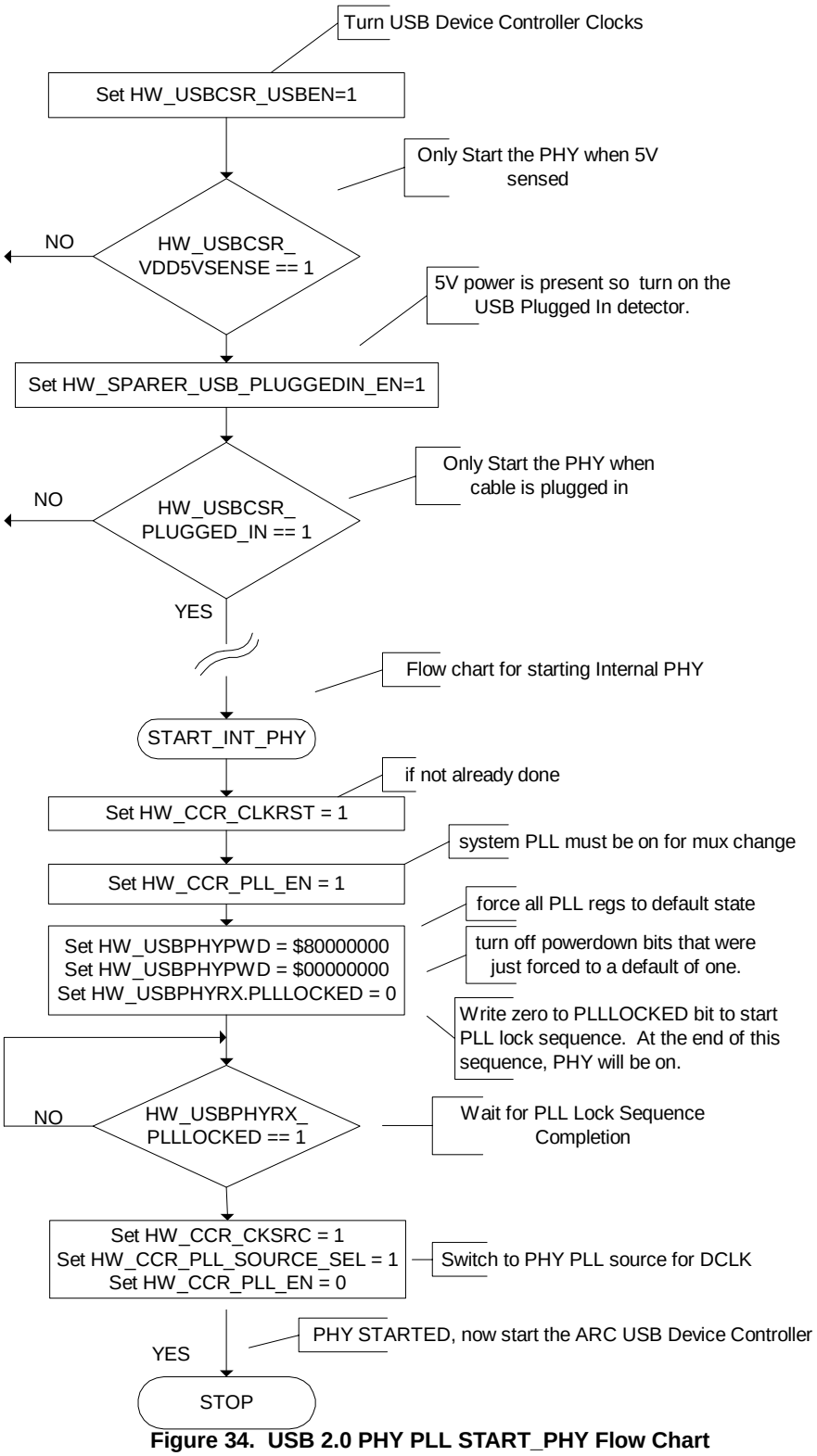


Figure 34. USB 2.0 PHY PLL START_PHY Flow Chart

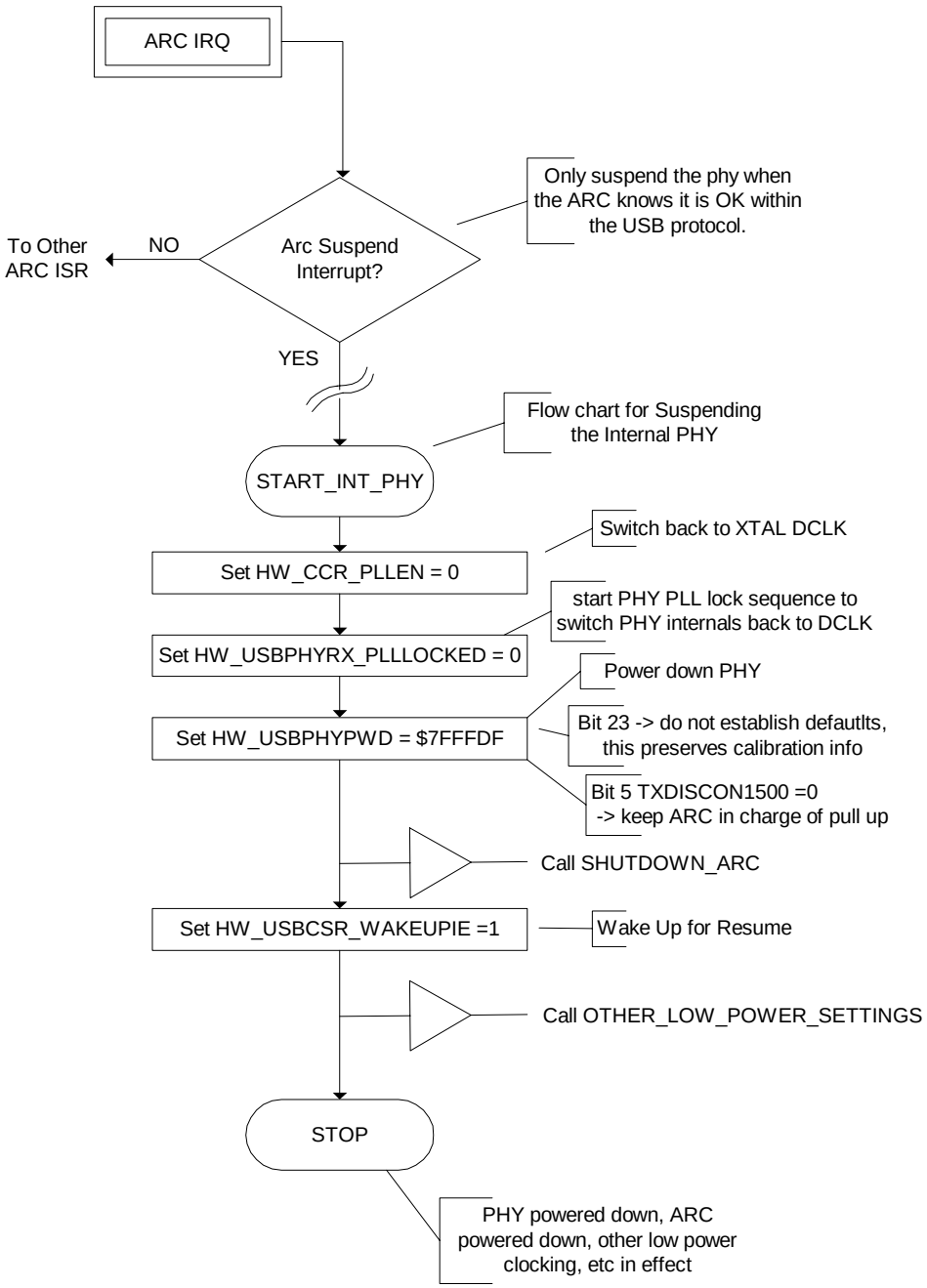


Figure 35. USB 2.0 PHY PLL Suspend Flowchart

9.7. Programmable Registers

The USB 2.0 Integrated PHY contains four registers that are directly programmable from the DSP, as follows:

9.7.1. USB PHY Analog Power Control

HW_USBPHYPWD X:$F210

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
REGRESET	PWDIBIAS		RXPWDRX	RXPWDDIFF	RXPWD1PT1	RXPWDENV	RXPWDDISCONDET		TXPWDCOMP	TXPWDVBG	TXPWDV2I	TXPWDIBIAS	TXPWDFS			PLLPWDCP	PLLPWDVCO	TXDISCON1500					

Table 91. HW_USBPHYPWD

BITS	LABEL	RW	RESET	DEFINITION
23	**REGRESET**	RW	0	Writing a one to this bit resets the Analog Control register to its default (reset) state. This bit always reads a Zero.
22	**PWDIBIAS**	RW	1	Powers down the bias generation circuit for the USB 2.0 PHY. This should be set only when all of the other power down bits (PWD) for the USB PHY are set to zero. This bit must be set to zero (low) during battery charge operation.
21	**RSRVD**	R	0	**RESERVED** -- must be written with zero.
20	**RXPWDRX**	RW	1	Set to one to power down the entire USB PHY receiver block except for the full speed differential receiver. Set to zero for normal operation.
19	**RXPWDDIFF**	RW	1	Set to one to power down the USB PHY high speed differential receiver. Set to zero for normal operation.
18	**RXPWD1PT1**	RW	1	Set to one to power down the USB 1.1 style full speed differential receiver. Set to zero for normal operation.
17	**RXPWDENV**	RW	1	Set to one to power down the USB 2.0 PHY receiver envelope detector (squelch signal). Set to zero for normal operation.
16	**RXPWDDISCONDET**	RW	1	Set to one to power down the USB 2.0 PHY receiver disconnect detector. Set to zero for normal operation.
15	**RSRVD**	R	0	**RESERVED** -- must be written with zero.
14	**TXPWDCOMP**	RW	1	Set to one to power down the USB 2.0 PHY transmit calibration comparator. Set to zero during calibration and set to one after calibration is complete.
13	**TXPWDVBG**	RW	1	Set to one to power down the USB 2.0 PHY transmit voltage band gap buffer amp as well as the V-to-I converter and the current mirror. Note these circuits are shared with the battery charge circuit, see ***ADD CROSS REF***, setting this bit to one will not power down these circuits unless the corresponding bit in the battery charger is also set for power down. Set to zero for normal operation and for calibration.

Table 92. USB PHY Analog Power Control

BITS	LABEL	RW	RESET	DEFINITION
12	**TXPWDV2I**	RW	1	Set to one to power down the USB 2.0 PHY transmit V-to-I converter and current mirror.Note these circuits are shared with the battery charge circuit, see ***ADD CROSS REF***, setting this bit to one will not power down these circuits unless the corresponding bit in the battery charger is also set for power down. Set to zero for normal operation and for calibration.
11	**TXPWDIBIAS**	RW	1	Set to one to power down the USB 2.0 PHY current bias block for the transmitter. This bit should only be set when the USB is in suspend mode. This effectively powers down the entire USB transmit path. Note these circuits are shared with the battery charge circuit, see ***ADD CROSS REF***, setting this bit to one will not power down these circuits unless the corresponding bit in the battery charger is also set for power down. Set to zero for normal operation and for calibration.
10	**TXPWDFS**	RW	1	Set to one to power down the USB 2.0 PHY full speed drivers. This turns off the current starvation sources and puts the drivers into a Hi-Z output.
9:8	**RSRVD**	R	00	**RESERVED** -- must be written with zero.
7	**PLLPWDCP**	RW	1	Set to one to power down the USB 2.0 PHY charge pump in the PLL. Should be used in conjunction with PLLPWDVCO to completely power down the PLL. Set to zero for normal operation.
6	**PLLPWDVCO**	RW	1	Set to one to power down the USB 2.0 PHY VCO in the PLL. This bit only powers down the VCO section, use in conjunction with PLLPWDCP. Set to zero for normal operation.
5	**TXDISCON1500**	RW	0	Set to zero to connect the integrated 1500Ω pull up resistor tied to the DPLUS USB pad. This defaults to “disconnected” to allow the DSP and USB controller to perform the appropriate initialization before connecting to the USB DPLUS & DMINUS.
4:0	**RSRVD**	R	00000	**RESERVED** -- must be written with zero.

Table 92. USB PHY Analog Power Control

9.7.2. USB PHY Analog Transmit Control

HW_USBPHYTX X:$F211

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TXCOMPOUT	TXFSHIZ	TXENCAL45DP		TXCAL_45DP				TXSKEW	TXHSTERM	TXENCAL45DN		TXCAL45DN				TXCALIBRATE	TXHSXCVR	TXENCAL1500		TXCAL1500			

Table 93. HW_USBPHYTX

BITS	LABEL	RW	RESET	DEFINITION
23	**TXCOMPOUT**	RW	0	The calibration comparator output is latched to this bit. This bit should be erased for every new calibration. This bit can set to zero or set to one by a normal write from the DSP. In addition, it is loaded with the state of the calibration comparator's output whenever **HW_USBPHT_TXCALIBRATE** is set to one. It continuously copies the comparator to this bit as long **TXCALIBRATE** is set to one, i.e. when **TXCALIBRATE** is one, writing to this bit has no effect.
22	**TXFSHIZ**	RW	1	Set to zero to tri-state the full speed driver. The actual tristate control is muxed between the normal source coming from the PHY transmitter logic and this bit. The mux is controlled by **HW_USBPHYRX_TXFSHIZEN**. Set to one for normal operation.
21	**TXENCAL45DP**	RW	0	Set to one for the time you wish to compare the 45Ω D_P termination resistor to the reference resistor. This bit should be set to one each time a new value of **HW_USBPHYTX_TXCAL45DP** is set in order to compare the resulting resistance. NOTE: only one of the following bits can be set to one for any calibration operation: **HW_USBPHYTX_TXENCAL1500**, **HW_USBPHYTX_TXCAL45DN** & **HW_USBPHYTX_TXENCAL45P**. Set to zero when D_P calibration is completed. The result of a comparison can be seen in **HW_USBPHYTX_TXCOMPOUT**.
20	**RSVRD**	R	0	**RESERVED** -- must be written with zero.
19:16	**TXCAL45DP**	RW	0110	Decode to select a 45Ω resistance for the D_P output pin. 0000= Maximum resistance. Resistance is centered by design at 0110. Perform calibration routine by initially setting to 1111 and counting down until the comparator trips.
15	**TXSKEW**	RW	0	Test mode bit to skew the transmit signal in order to test the receiver sensitivity.

Table 94. USB PHY Analog Transmit Control

BITS	LABEL	RW	RESET	DEFINITION
14	**TXHSTERM**	RW	0	Set to one to connect the high speed 45Ω terminations. This forces both of the FS drivers to zero so that the 45Ω resistors are connected between D_P/D_N and ground. It also disconnects the 1500Ω resistor from VDD so that the pull-up is not there. When first communicating with a hub, the PHY needs to be in Full Speed mode, i.e. leaving this bit at zero. Only after the proper high speed handshaking should this bit be set to one. When the PHY enters Suspend mode, this bit needs to revert to zero so that Full Speed terminations are re-established. 0--> full speed termination, 1 --> high speed termination. This bit is muxed with the inverse of UTMI_TERM. The mux is controlled by **HW_USBRX_TERMSELECTEN**.
13	**TXENCAL45DN**	RW	0	Set to one for the time you wish to compare the 45Ω D_N termination resistor to the reference resistor. This bit should be set to one each time a new value of **HW_USBTX_TXCAL45DN** is set in order to compare the resulting resistance. NOTE: only one of the following bits can be set to one for any calibration operation: **HW_USBTX_TXENCAL1500**, **HW_USBPYTX_TXCAL45DN** & **HW_USBPHYTX_TXENCAL45P**. Set to zero when D_N calibration is completed. The result of a comparison can be seen in **HW_USBPHYTX_TXCOMPOUT**.
12	**RSVRD**	R	0	**RESERVED** -- must be written with zero.
11:8	**TXCAL45DN**	RW	0110	Decode to select a 45Ω resistance for the D_N output pin. 0000= Maximum resistance. Resistance is centered by design at 0110. Perform calibration routine by initially setting to 1111 and counting down until the comparator trips.
7	**TXCALIBRATE**	RW	0	Set to one to effect calibration of any of the three precision resistances and set back to zero to read the result of calibration in **HW_USBPHYTX_TXCOMPOUT**. When set to one, it causes the calibration comparator output to continuously update the state of **HW_USBPHYTX_TXCOMPOUT**. Set to zero for normal operation. NOTE: only one of the following bits can be set to one for any calibration operation: **HW_USBTX_TXENCAL1500**, **HW_USBPYTX_TXCAL45DN** & **HW_USBPHYTX_TXENCAL45P**.
6	**TXHSXCVR**	RW	0	Set to one to enable the high speed transceiver. This enables the data lines to control the current steer block. Set to zero for full speed operation. This bit is muxed with the inverse of UTMI_XCVR_SELECT. The mux is controlled by **HW_USBRX_XCVRSELECTEN**.
5	**TXENCAL1500**	RW	0	Set to one for the time you wish to compare the 1500Ω R_{PU} resistor to the reference resistor. This bit should be set to one each time a new value of **HW_USBTX_TXCAL1500** is set in order to compare the resulting resistance. NOTE: only one of the following bits can be set to one for any calibration operation: **HW_USBTX_TXENCAL1500**, **HW_USBPYTX_TXCAL45DN** & **HW_USBPHYTX_TXENCAL45P**. Set to zero when R_{PU} calibration is completed. The result of a comparison can be seen in **HW_USBPHYTX_TXCOMPOUT**.
4	**RSVRD**	R	0	**RESERVED** -- must be written with zero.
3:0	**TXCAL1500**	RW	1000	Decoded to select a 1500Ω resistance for the R_{PU} Output. 0000= Maximum resistance. Resistance is centered by design at 1000. Perform calibration routine by initially setting to 1111 and counting down until the comparator trips.

Table 94. USB PHY Analog Transmit Control

9.7.3. USB PHY Analog PLL Control

HW_USBPHYPLL X:$F212

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PLLCLKDIVRSTZ	PLLVCOKSTART	PLLCPSHORTLFR	PLLPFDRST					PLLCLKDIVSEL				PLLCPNSEL				PLLVCOCLK24	PLLVCOCLK2	PLLCPDBLIP		PLLV2ISEL			

Table 95. HW_USBPHYPLL

BITS	LABEL	RW	RESET	DEFINITION
23	**PLLCLKDIVRSTZ**	RW	1	This bit should normally be set to one. It can be momentarily set to zero and then back to one to provide a pulse to load the clock divider bits whenever HW_USBPHPLL_PLLCLKDIVSEL is changed. DCLK should always be switch back to the 24MHz crystal oscillator source before changing the PLL frequency.
22	**PLLVCOKSTART**	RW	0	This test bit is provided for the very unlikely event that the VCO does not start oscillation. This theoretically possible but highly unlikely event can only happen in a noiseless system-- an unlikely scenario. This bit is normally set to zero. To kick start the VCO, perform a zero to one transition on this bit followed by a one to zero transition.
21	**PLLCPSHORTLFR**	RW	0	This normally low test mode bit is used to short the charge pump resistor for a highly under-damped response. Set to one to short the resistor. The resistor should only be shorted in test mode.
20	**PLLPFDRST**	RW	0	This bit can be used to reset the PFD. This bit is set to zero for normal operation. To reset the PFD, perform a zero to one transition on this bit followed by a one to zero transition. This transition is not needed for normal operation.
19:16	**RSVRD**	R	0000	**RESERVED** -- must be written with zero.
15:12	**PLLCLKDIVSEL**	RW	0000	The PLLCLKDIVSEL bit-field is used to select the PLL feedback divide ratios. See Table 97, “PLL Clock Divider Values,” on page 98 for recommend values for 24MHz and 20MHz crystals. 0000 DIV20 0001 DIV24 0010 DIV30 0011 DIV16 0100 DIV12 0101 DIV10 0110 DIV8 0111 DIV6 1XXX reserved, do not use

Table 96. USB PHY Analog PLL Control

BITS	LABEL	RW	RESET	DEFINITION
11:8	**PLLCPNSEL**	RW	0000	These bits are set in conjunctions with the HW_USBPHYPLL_PLLCLKDIVSEL bits to maintain a constant loop filter damping factor for the different divide ratios. They can also be used independently to speed up or slow down the activity of the PLL. Recommended settings are found in Table 97, “PLL Clock Divider Values,” on page 98. 0000 ip current (default) 0001 ip current 0010 1.50 * ip current 0011 0.75 * ip current 0100 0.50 * ip current 0101 0.50 * ip current 0110 0.40 * ip current 0111 0.40 * ip current 1XXX reserved do not use
7	**PLLVCOCLK24**	RW	0	Set to one to disable four of the eight phases of PLL clock output, i.e. turn off vco_clk1, vco_clk1z, vco_clk3, and vco_clk3z. Only vco_clk2, vco_clk2z, vco_clk4, and vco_clk4z remain enabled when this bit is set. Disabling these clock phases when they are not needed can save approximately 1mA. Set to zero to enable all eight phases out of the PLL. Note this bit overlaps with **PLLVCOCLK2** in disabling these phases.
6	**PLLVCOCLK2**	RW	0	Set to one to disable six of the eight phases of PLL clock output, i.e. turn off vco_clk1, vco_clk1z, vco_clk3, vco_clk3z, vco_clk4, and vco_clk4z. Only vco_clk2 and vco_clk2z remain enabled when this bit is set to one. vco_clk2 is also used as the digital clock for the 480MHz digital clock domain and the various digital clock domains resulting from divisions of the 480MHz oscillator. Disabling these clock phases when they are not needed can save approximately 1.5mA. Set to zero to enable driving all phases out of the PLL. Note: this bit overlaps with **PLLVCOCLK24** in disabling some phases. Setting this bit provides useful power reductions when the high speed PLL is used to drive DCLK in applications that do not use the USB.
5	**PLLCPDBLIP**	RW	0	Set this bit to one to double the charge pump current to speed up lock time. It can be used in conjunction with HW_USBPHYPLL_PLLCPNSEL to change the loop performance. At start up time it can be set to one to shorten the lock time. During normal operation, this be should be set to zero for lowest overall tracking jitter.
4	**RSVRD**	R	0	**RESERVED** -- must be written with zero.
3:0	**PLLV2ISEL**	RW	0000	These bits can be used to extend the frequency range of the PLL. 0000 Nominal frequency range (default) 0001 Lower the useful frequency range 0010 Lowest useful frequency range 0011 Highest useful frequency range 01XX Reserved, do not use 1XXXf Reserved, do not use

Table 96. USB PHY Analog PLL Control

.

PLLCLKDIVSEL	DIVIDER	FREQUENCY FOR 24MHZ CRYSTAL OSCILLATOR	FREQUENCY FOR 20MHZ CRYSTAL OSCILLATOR	PLLCPNSEL
0000	DIV20	480MHz	400MHz	0000,0001
0001	DIV24	576MHz	480MHz	0000,0001
0010	DIV30	720MHz	600MHz	0010
0011	DIV16	384MHz	320MHz	0011
0100	DIV12	288MHz	240MHz	0100
0101	DIV10	240MHz	200MHz	0101
0110	DIV8	192MHz	160MHz	0110
0111	DIV6	144MHz	120MHz	0111
1XXX	unused	unused	unused	

Table 97. PLL Clock Divider Values

9.7.4. USB PHY Analog Receive Control

HW_USBPHYRX X:$F213

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PLLLOCKED	REGRXDBYPASS	FSCKSOURCESEL	HOSTMODETEST	PLLCKDIVCTL				TXFSHIZEN	XCVRTERMSELECTEN	PLLLKTIMECTL		DEBUGMODE				DISCONADJ				ENVADJ			

Table 98. HW_USBPHYRX

BITS	LABEL	RW	RESET	DEFINITION
23	**PLLLOCKED**	R	PLL LOCK STATE	This bit is set whenever the PLL achieves its lock state, i.e. whenever the PLL lock counter counts down to zero. This is determined by design and from characterization to be a maximum time period. The PLL includes a lock counter to time this period. The initial value of the lock counter is taken from the **HW_USBPHYRX_PLLLKTIMECTL** bit field. What is written to this bit is indirectly related to what is read.
		W	N/A	Software must write a zero to this bit position to initiate a new PLL lock count. After powering up the USB PHY or changing the PLL feedback divider, software must write a zero to this bit to restart the lock count. **WARNING:** To modify other bits in this register without initiating a PLL lock cycle, software must write a ONE to this bit position.
22	**REGRXDBYPASS**	RW	0	Set this bit to one to use the output of the D_P single ended receiver in place of the full speed differential receiver. This test mode is intended for lab use only. For normal operation, set this bit to zero.

Table 99. USB PHY Analog RX Control

BITS	LABEL	RW	RESET	DEFINITION
21	**FSCKSOURCESEL**	RW	0	The UTMI interface of the PHY and the digital full speed transceiver receive a 60MHz clock from the PHY's PLL, see Figure 33. "USB 2.0 PHY PLL Block Diagram" on page 88. The source of this clock is either a divide by eight from the 480MHz PLL or from the chip wide DCLK. The FSCKSOURCESEL bit selects the 480MHz divided by eight or the DCLK as the source. Set this bit to zero to use the 480MHz PLL divided by eight to drive the Full Speed transceiver and the divide by two that generates the UTMI clock. Set this bit to one to use the chip wide DCLK to drive the Full Speed transceiver and the divide by two that generates the UTMI clock.
20	**HOSTMODETECT**	RW	0	Set this bit to one to put the USB 2.0 PHY into host mode. This mode is not supported in the STMP35xx, however the functionality is built into the PHY and will be characterized for subsequent product use of the USB 2.0 PHY in host mode. Set this bit to zero for normal operation.
19:16	**PLLCKDIVCTL**	RW	0111	The USB 2.0 PHY PLL can be used as the clock generator to drive DCLK for the entire chip. These bits select the divide ratio used for driving DCLK. The USB 2.0 PHY PLL frequency is adjusted using HW_USBPHYPLL_PLLCLKDIVSEL. For some of the values of PLLCKDIVCTL, the PLL must be running at 120MHz maximum frequency, these values are marked with a †. 0000 DIV1† (PLL freakiness ≤ 120MHz pass through) 0001 DIV2† (PLL freakiness ≤ 120MHz) 0010 DIV3† (PLL freakiness ≤ 120MHz) 0011 DIV4 (480MHz ÷ 4 = 120MHz = DCLK) 0100 DIV5 (480MHz ÷ 5 = 96MHz = DCLK) 0101 DIV6 (480MHz ÷ 6 = 80MHz = DCLK) 0110 DIV7 (480MHz ÷ 7 = 68.57MHz = DCLK) 0111 DIV8 (480MHz ÷ 8 = 60MHz = DCLK) 1000 DIV9 (480MHz ÷ 9 = 53.33MHz = DCLK) 1001 DIV10 (480MHz ÷ 10 = 48MHz = DCLK) 1010 DIV11 (480MHz ÷ 11 = 43.64MHz = DCLK) 1011 DIV12 (480MHz ÷ 12 = 40MHz = DCLK) 11XX reserved, do not use
15	**TXFSHIZEN**	RW	0	This bit is set to one to override the normal control by the PHY transmitter of the full speed driver's Hi Z control signal. Set to zero for normal operation, i.e. controlled by the PHY transmitter.
14	**XCVRTERMSELECTEN**	RW	0	This bit is set to one to override the normal control by the ARC core of the xcvr_select and the term_select signals. Set to zero for normal operation.

Table 99. USB PHY Analog RX Control

BITS	LABEL	RW	RESET	DEFINITION
13:12	**PLLLKTIMECTL**	RW	10	These bits determine the number of clocks counted by the lock timer before it reports the locked condition in **HW_USBPHYRX_PLLLOCKED**. The maximum lock time across temperature and process variations is 90 µseconds with register default settings. Lock is defined here as stabilizing within 1% of final frequency. This field selects a 10-bit, 11-bit, 12-bit or 13-bit counter. The clock input to this counter comes from the PLL reference clock source, i.e. the crystal oscillator. For a 24MHz crystal, the lock times selected are as follows: 00 1024 clocks or 42.67 µS 01 2048 clocks or 85.33 µS 10 4096 clocks or 170.67 µS 11 8192 clocks or 341.33 µS
11:8	**DEBUGMODE**	RW	0000	These bits are used for debug only. These bits **must** be set to zero for normal operation. 0000 Normal operation. 0001 HS DLL bypass 0010 HS elastic buffer bypass 0011 Sync detect count reduction 0100 511 test pattern generation and detection 01XX undefined 1XXX undefined
7:4	**DISCONADJ**	RW	0000	This bit field adjusts the trip point for the disconnect detector. FIELD TRIP LEVEL VOLTAGE 0000 0.57500 volts 0001 0.56875 volts 0010 0.58125 volts 0011 0.58750 volts 01XX reserved, do not use 1XXX reserved, do not use
3:0	**ENVADJ**	RW	0000	This bit field adjusts the trip point for the envelope detector. FIELD TRIP LEVEL VOLTAGE 0000 0.12500 volts 0001 0.11875 volts 0010 0.13125 volts 0011 0.13750 volts

Table 99. USB PHY Analog RX Control

The **HW_USBPHYRX_PLLCKDIVCTL** bit field selects the divide ratio used to generate DCLK when the USB 2.0 PHY PLL is used to drive DCLK to the entire chip. The PHY PLL is of course adjustable and must be set to less than 120MHz for several of the DIVCTL settings. The following table shows the DCLK values available when the PLL is set at 480MHz and when it is set at 120MHz. The divider value DIV1 should only be used in test modes.

PLLCKDIVCTL	DIVIDE BY	DCLK FREQUENCY FOR 480MHZ USB 2.0 PHY PLL	DCLK FREQUENCY FOR 120MHZ USB 2.0 PHY PLL	
0000	DIV1	N/A	120MHz	
0001	DIV2	N/A	60MHz	
0010	DIV3	N/A	40MHz	
0011	DIV4	120MHz	30MHz	
0100	DIV5	96MHz	24MHz	
0101	DIV6	80MHz	20MHz	
0110	DIV7	68.57MHz	17.1MHz	
0111	DIV8	60MHz	15MHz	reset
1000	DIV9	53.33MHz	13.33MHz	
1001	DIV10	48MHz	12MHz	
1010	DIV11	43.64MHz	10.9MHz	
1011	DIV12	40MHz	10MHz	
11XX	unused	unused	unused	

Table 100. PLL Clock Divider Values

10. PARALLEL EXTERNAL MEMORY CONTROLLER (EMC)

The chip includes an external memory controller that has two major functional modes: SmartMedia/NAND and CompactFlash. The SmartMedia/NAND flash interface provides a state machine that provides all of the logic necessary to perform DMA functions between on-chip RAM and the flash. The CompactFlash interface supports all three major CompactFlash modes: Memory, I/O and IDE. These modes can be used to communicate with standard CompactFlash (CF) devices such as CF Flash and the IBM MicroDrive. The CF Memory mode can be used to communicate with standard ATA/ATAPI devices like CD-ROM and Hard drives.

This documentation will detail the configuration and operation of the external memory interface. It will discuss the standard use of the interface for flash applications. Additional information about other specific applications (CD-ROM, DRAM, NOR Flash, etc.) will be available in future application notes.

10.1. EMC Overview

The external memory controller has several major features:

- DMA data transfers allow minimal CPU overhead
- 32-bit SmartMedia/NAND addressing supports future devices up to 4Gbyte
- Multiple device support with four SmartMedia/NAND and two CF chip selects
- Configured timing can be set to support various devices

The external memory controller can be described as two fairly independent devices in one: a SmartMedia/NAND flash interface and a CompactFlash/NOR flash/IDE interface. Both interfaces share the same device pins, some registers and the DMA engine.

Both interfaces uses memory mapped registers to setup and control the transactions. Data is always sent through the DMA – there are no data registers that correspond to the interface data bus. Transactions are always started with a kick bit. The interface sets up the control lines and transfers data to/from the internal RAM. Once the transaction is complete the interface signals the DSP with either a polled flag or an interrupt. The kick bit, polling bit and interrupt configurations are all contained in the registers.

10.1.1. EMC External Pins

The EMC supports several different modes of operation, and the pins used in each mode overlap substantially. The following table shows how the pins relate to each other.

SMARTMEDIA/ NAND PIN NAME	CF+/COMPACTFLASH PIN NAME, MEMORY MODE	CF+/COMPACTFLASH PIN NAME, I/O MODE	CF+/COMPACTFLASH PIN NAME, TRUE IDE MODE
SM_D7 - SM_D0	CF_D7 - CF_D0	CF_D7 - CF_D0	CF_D7 - CF_D0
SM_WEn	CF_WEn	CF_WEn	CF_WEn
SM_REn	CF_OEn	CF_OEn	CF_OEn
*	CF_A10	CF_A10	*
SM_ALE	CF_A9	CF_A9	*
SM_CLE	CF_A8	CF_A8	*
SM_SEn	CF_A7	CF_A7	*
SM_CE0n	CF_A6	CF_A6	*
SM_CE2n	CF_A5	CF_A5	*
SM_CE3n	CF_A4	CF_A4	*
*	CF_A3	CF_A3	*
*	CF_A2	CF_A2	CF_A2
*	CF_A1	CF_A1	CF_A1
*	CF_A0	CF_A0	CF_A0
SM_CE1n	CF_CE0n	CF_CE0n	CF_CE0n
*	CF_CE1n	CF_CE1n	CF_CE1n
*	*	CF_IOWRn	CF_IOWRn
*	*	CF_IORDn	CF_IORDn
*	CF_WPn	CF_WPn	CF_WPn
*	CF_REGn	CF_REGn	CF_REGn
*	CF_RESETn	CF_RESETn	CF_RESETn
*	CF_BVD1/IREQ	CF_BVD1	CF_BVD1
SM_WPn	*	*	*
*	CF_READY	CF_READY	CF_READY
*	CF_CDn	CF_CDn	CF_CDn
SM_READY	CF_WAITn	CF_WAITn	CF_WAITn

Table 101. Mapping of pins to CF+/CompactFlash and SmartMedia/NAND Card/Device Pins

Note: Pins marked with (*) are not used by the module and are available for use in GPIO mode. See the GPIO module documentation for more information. I/O mode and IDE modes are not normally used in any application. Their pin names are not included in the pin list at the end of this document.

10.2. General Use of the External Memory Interface

The external memory interface has some functions that are shared between both the SmartMedia/NAND and the CompactFlash modes. In particular, common registers control the functions that communicate with the DSP such as the DMA and transaction start/done functions. Each of the memory interfaces also have their own registers that set up their specific parameters. This section will focus on the common registers.

10.2.1. Common Flash Registers

10.2.1.1. Flash Control Register

HW_FLCR X:$F000

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		SRST		NB														MMD		IRQP	TCIE	RW	KICK

Table 102. HW_FLCR

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**SRST**	RW	0	**Flash module software reset –** This bit is itself reset by a hardware reset only; a software reset does not clear the bit.
20:19	**RSRVD**	R	0	**Reserved –** Must be written with 0.
18:6	**NB**	RW	$001	**Number of bytes to transfer**
5:4	**MMD**	R	0	**External interface type** 00 SmartMedia/NAND 10 Reserved 01 CompactFlash 11 Reserved
3	**IRQP**	RW	0	Reading a one indicates a pending interrupt; writing back a one **de-asserts** the interrupt; writing zero has no effect.
2	**TCIE**	RW	0	Setting this bit enables the transaction-complete interrupt.
1	**RW**	RW	0	Setting this bit initiates an external memory transfer is a read; otherwise it is a write.
0	**KICK**	RW	0	Setting this bit initiates an external memory transfer; it automatically clears when the transfer completes.

Table 103. Flash Control Register Description

The Flash Control Register sets up basic operating parameters. After the transaction has been set up in all of the appropriate registers (transaction type, timing, address, etc.) the software triggers the kick bit. This will initiate the transaction. The status bit will indicate when the transaction has been completed. The DSP can either poll the status bit or the register can be configured to send an interrupt on transaction completion.

The Flash Start Address Low and High Registers determine the addresses of both the internal XRAM, YRAM and PRAM and the external device. Multiple cycle transactions (reading/writing multiple bytes sequentially) will pack/unpack data from the DSP XRAM, YRAM and PRAM's 24-bit words into the external device's 8-bit interface. The LSB is packed/unpacked first. Each byte is packed with its MSB at bit 7, 13, or 23 for bytes 1, 2 or 3.

The address bus's behavior depends on the specific external device type. SmartMedia/NAND devices don't use the address bus. They are externally selected with chip select lines (CE0-3) and internally with multi-byte accesses over the I/O lines (see more in the SmartMedia/NAND section, below). CompactFlash devices may use the Address pins in Memory or I/O mode or with a combination of control pins (CE0, CE1, CF_A0, A1, A2) in IDE Mode. In all cases the mode configuration registers determine how the target device is addressed.

10.2.1.2. Flash Control 2 Register

HW_FLCR2 X:$F004

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																	CKGT		NDMA			ASEL	

Table 104. HW_FLCR2

BITS	LABEL	RW	RESET	DEFINITION
23:7	**RSRVD**	RW	0	**Reserved –** Must be written with 0.
6	**CKGT**	RW	0	Turns clocks to Flash module off.
5	**RSRVD**	R	10	**Reserved –** Must be written with 0.
4	**NDMA**	RW	0	Inverts data from the Flash to the system.
3:2	**RSRVD**	R	00	**Reserved –** Must be written with 0.
1:0	**ASEL**	RW	00	Memory space to use for DMA transfers 00 X space 10 P space 01 Y space 11 Reserved

Table 105. Flash Control 2 Register Description

The Flash Control 2 Register sets up additional operating parameters.

10.2.1.3. Flash Start Address Low Register

HW_FLSALR X:$F001

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
XA																							

Table 106. HW_FLSALR

BITS	LABEL	RW	RESET	DEFINITION
23:0	**XA**			Lower 24 bits of external memory starting address

Table 107. Flash Start Address Low Register Description

10.2.1.4. Flash Start Address High Register

HW_FLSAHR X:$F002

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																XA							

Table 108. HW_FLSAHR

BITS	LABEL	RW	DEFINITION
23:8	**DA**	R	Starting address for DMA transfer.
7:0	**XA**	R	Upper 8 bits of external memory starting address

Table 109. Flash Start Address High Register Description

10.3. External Memory Interface with SmartMedia/NAND Flash Devices

The external memory interface will commonly be used with SmartMedia/NAND flash devices. SmartMedia are small removable cards that contain one or two NAND Flash devices. Alternatively, the system designer can use non-removable NAND flash chips. Both devices use the same pins. The SmartMedia/NAND electrical interface uses an 8-bit data/address bus and 8 control lines. Multiple devices share the same bus, with chip selects used for selection.

10.3.1. SmartMedia/NAND Pins

- SM_D0 - SM_D7: 8-bit I/O interface. This bus is used to send addresses (multiple bytes required to send a complete address), data and commands to the SmartMedia/NAND device. Data, device ID and status information are received from the Flash over this bus.
- SM_WEn: Write Enable. This active low output is used to indicate a write cycle to the Flash. The external memory interface will write to the bus during a WE~ cycle.
- SM_REn: Read Enable. This active low output indicates a read cycle to the Flash. The flash device writes to the bus during a RE~ cycle.
- SM_ALE: Address Latch Enable. This active high output indicates that the data on the I/O bus is part of an address. The chip outputs one byte of a Flash memory address during this cycle.
- SM_CLE: Command Latch Enable. This active high output indicates that the data on the I/O bus is a flash command. The chip outputs a SmartMedia/NAND command during this cycle. See the data sheet for your SmartMedia/NAND device for more information on the valid commands.
- SM_SEn: Spare Enable Select. This active low output is used to indicate that the ECC data should be sent with a regular 512 Byte Flash Block. This command is not typically used since the ECC should always be used. It is recommended that the SE~ line be tied to ground (asserted) to ensure that the spare area is always used.
- SM_CE0n – SM_CE3n: Chip Enable Lines. Active low outputs that select each SmartMedia or NAND Flash. Connect one to each of the flash devices.
- SM_WPn: Write Protect. This active low output is used prevent accidental data corruption during non-write cycles.

- SM_READY: Ready/ Busy~. This active low input indicates that the flash device is busy. No new transactions to any devices should occur unless this line is not asserted. The Ready/Busy~ line uses open-drain drivers and requires an external pull-up resistor. All SmartMedia/NAND Flash devices busy pins should be connected to this pin.

The basic unit of access of a SmartMedia/NAND device is the 8 bit byte. All bytes must be written in complete 528 byte *“pages”*. A *“block”* is made up of 32 pages. A 64MByte NAND Flash has 4096 blocks, see Figure 36. “NAND Flash Device Block Diagram” on page 108. When an address is sent to a NAND Flash it is sent in three or four bytes depending on the size of the device.

CYCLE	I/O 7	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	I/O 0
ADDR 0	A7	A6	A5	A4	A3	A2	A1	A0
ADDR 1	A16	A15	A14	A13	A12	A11	A10	A9
ADDR 2	A24	A23	A22	A21	A20	A19	A18	A17
ADDR 3	0	0	0	0	0	0	0	A25

Table 110. SmartMedia/NAND Device Address Bytes

Transfers to and from a NAND Flash device are made in packets, starting with a control byte which may be followed by one, three or four bytes of address. The command and address portion can be followed by up to 528 bytes of data for a read or program (write) command, see Figure 37. “NAND Flash Device Command & Data Sequences” on page 109. When a NAND Flash contains more than 32MBytes of storage the *optional* four address byte format has to be used. Pay careful attention to the device address bit mapping in Table 110, “SmartMedia/NAND Device Address Bytes,” on page 107. There is no address bit 8. This reflects the growth of page sizes from 256 bytes to 512 bytes as NAND flash technology evolved. Three read commands are provided to allow starting at offsets of zero bytes, 256 bytes or 512 bytes into a 528 byte page.

The reader is cautioned to pay careful attention to the mapping of Flash Start Address Low and High register bits to the four address bytes. Recall the “missing” address bit 8 in the NAND device’s interpretation of these address bytes.

CYCLE	I/O 7	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	I/O 0
ADDR 0	AL7	AL6	AL5	AL4	AL3	AL2	AL1	AL0
ADDR 1	AL15	AL14	AL13	AL12	AL11	AL10	AL9	AL8
ADDR 2	AL23	AL22	AL21	AL20	AL19	AL18	AL17	AL16
ADDR 3	AH7	AH6	AH5	AH4	AH3	AH2	AH1	AH0

Table 111. Flash Address Low/High mapping to SmartMedia/NAND

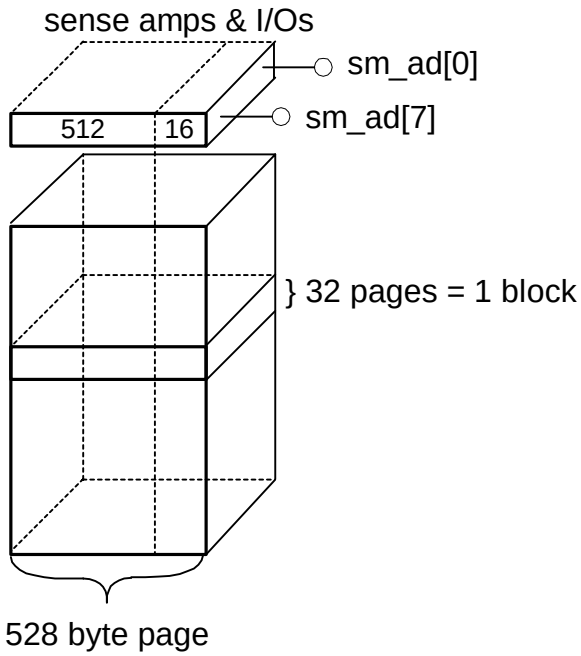


Figure 36. NAND Flash Device Block Diagram

The EMC NAND FLASH controller supports six initial commands (iCMD) and two packet ending commands (eCMD) as shown in the figure below. Initial commands include:

- The RESET command initializes the NAND flash internal state machine to idle.
- The Read Status command is useful to tell when a page has been programmed or a block has been erased.
- The Read Data command is used to fetch a page of data from the Flash. One should generally fetch all 528 bytes and apply any error correction stored in the extra 16 bytes to the 512 real data bytes.
- The Program Data command writes 528 data bytes to a page in the NAND Flash.
- Each device has a built in vendor ID and device ID that can be read from the device with a READ ID command.

Ending commands include:

The AUTO ERASE START command $60.

The AUTO PAGE PROGRAM command $10.

The appropriate ending command is automatically sent by hardware at the end of the program sequence and the erase block commands.

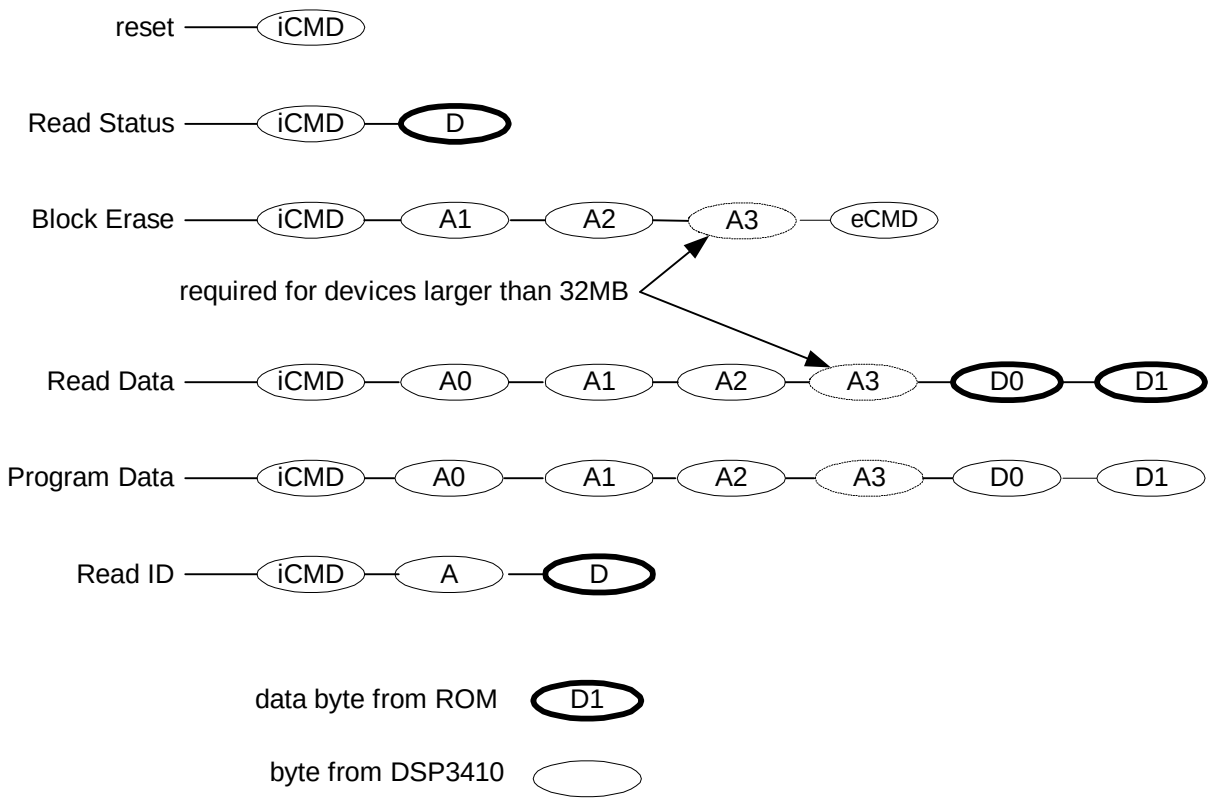


Figure 37. NAND Flash Device Command & Data Sequences

As the above figure shows, all packets sent to the NAND device start with a command byte followed by the appropriate number of address bytes. Either data bytes are then sent to the device or data./status/ID bytes are received from the device. Note the first address byte sent on the Block Erase command is Address Byte 1.

All timing references in the NAND Flash controller are based on DCLK. Since DCLK is quite variable, the timing parameters must be recalculated and reloaded whenever the frequency of DCLK changes. For a timing references, see Figure 38. “NAND Flash Write Timing” on page 110.

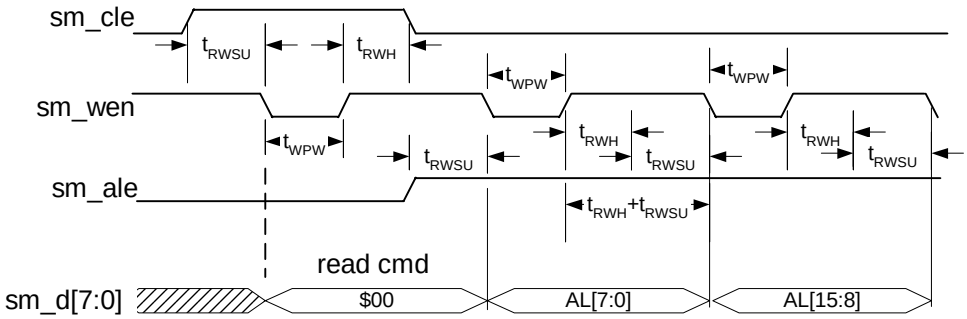


Figure 38. NAND Flash Write Timing

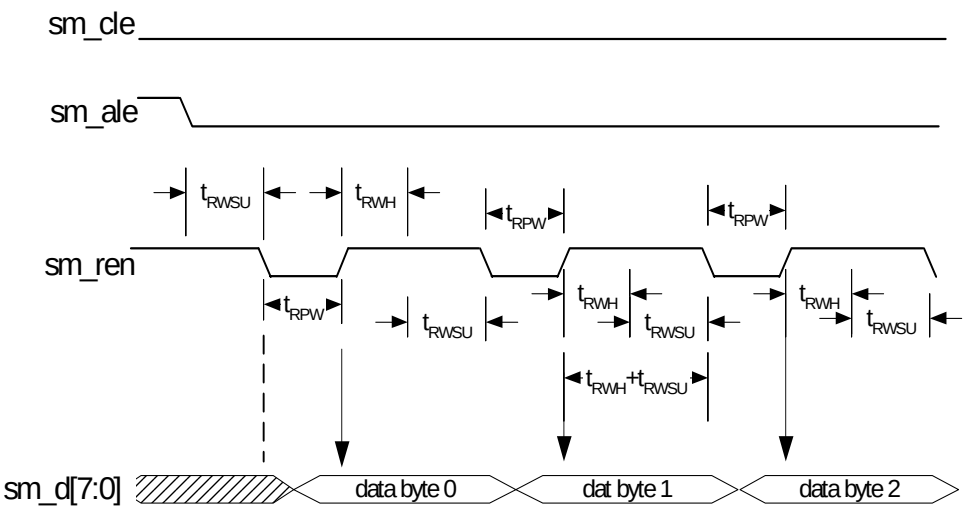


Figure 39. NAND Flash Read Timing

10.3.2. SmartMedia/NAND Registers

10.3.2.1. SmartMedia/NAND Control Register

HW_FLSMCR X:$F010

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
							BPIRQ	TOIRQ	BPIE	TOIE	ICMD								SIZE	WP	SE	CS	

Table 112. HW_FLSMCR

BITS	LABEL	RW	RESET	DEFINITION
23:17	**RSRVD**	R	0	**Reserved –** Must be written with 0.
16	**BPIRQ**	R	0	**Bad programming interrupt pending –** Reading a one indicates a pending interrupt; writing back a one **de-asserts** the interrupt; writing zero has no effect.
15	**TOIRQ**	RW	0	**Timeout interrupt pending –** Reading a one indicates a pending interrupt; writing back a one **de-asserts** the interrupt; writing zero has no effect.
14	**BPIE**	RW	0	**Bad Programming interrupt enable –** Generate BPIRQ if SmartMedia/NAND interface is kicked with bad programming.
13	**TOIE**	RW	0	**SmartMedia/NAND Timeout interrupt enable**
12:5	**ICMD**	R	$FF	**Initial standard SmartMedia/NAND command for transaction** $FF Card Reset $70 Card Status Register Read $60 Card Block Erase. Second command byte is always $D0. $00 Read/Program Data. Start address is in first 256 bytes of 528-byte page. $01 Read/Program Data. Start address is in second 256 bytes of 528-byte page. $50 Read/Program Data. Start address is in redundant area (last 16-bytes of page). $90 Car ID Read
4	**SIZE**	R	0	**Target SmartMedia/NAND device size** 0 <= 32 MBytes 1 > 32 MBytes
3	**WP**	RW	0	**Write Protect –** Controls the output of the SM_WPn pin (active-low) 0 Assert -WP pin 1 De-assert -WP pin
2	**SE**	RW	0	**-SE pin control –** Allows the spare (redundant) area (last 16 bytes of page) to be skipped. 0 assert -SE pin. Redundant area enabled. 1 de-assert -SE pin. Redundant area disabled.

Table 113. SmartMedia/NAND Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
1:0	**CS**	R	0	**External Chip Select** 00 Select Chip 0 01 Select Chip 1 10 Select Chip 2 11 Select Chip 3

Note: The SmartMedia/NAND interface's state machine logic makes a number of checks on the legitimacy of DSP programming before allowing the master state machine out of its IDLE state. If these checks fail, the state machine remains IDLE, the BPIRQ gets set and the transaction is considered done (Kick bit in the HW_FLSMCR register is de-asserted). Optionally, a bad-programming interrupt can be enabled by setting the BPIE bit.

Table 113. SmartMedia/NAND Control Register Description (Continued)

The SmartMedia/NAND Control Register configures the interface and determines the type of transaction. The SmartMedia/NAND portion of the interface was designed to very closely interface to external Flash device. The interface knows how to set the appropriate control and data bits at a device command level. Operations like Reset, accesses to the Status Register and data operations are automatically processed by the interface. The user code just needs to set up some basic timing and device information and make a high level request.

10.3.2.2. SmartMedia/NAND Timer 1 Register

HW_FLSMTMR1R X:$F011

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	TRWH						TWPW						TRPW						TRWSU				

Table 114. HW_FLSMTMR1R

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21:17	**TRWH**	RW	00010	Read/Write Pulse Hold in terms of dclks
16:11	**TWPW**	RW	000110	Write Pulse Width in terms of dclks
10:5	**TRPW**	RW	000110	Read Pulse Width in terms of dclks
4:0	**TRWSU**	RW	00010	Read/Write Strobe Setup in terms of dclks

Table 115. SmartMedia/NAND Timer 1 Register Description

10.3.2.3. SmartMedia/NAND Timer 2 Register

HW_FLSMTMR2R X:$F012

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TWTO																		TWT					

Table 116. HW_FLSMTMR2R

BITS	LABEL	RW	RESET	DEFINITION
23:6	**TWTO**	RW	$001E8	Delay before timing out on RDY pin in terms of dclks
5:0	**TWT**	RW	010000	Delay before examining RDY pin in terms of (dclks x 1024)

Table 117. SmartMedia/NAND Timer 2 Register Description

The SmartMedia/NAND timing registers default to conservative settings on reset.

TIMING PARAMETER	DEFAULT IN DCLK CYCLES	TIME @ 48MHZ (21NS CYCLE)	SAMSUNG SPEC
Trwh	2	42ns	40ns
Twpw	6	125ns	80ns
Trpw	6	125ns	80ns
Trwsu	2	42ns	30ns
Twto	488 * 1024	10.5ms	4ms
Twait	16	336ns	200ns

Table 118. SmartMedia/NAND Timing Specifications

The firmware designer can change the timing to accommodate higher performance or newer flash devices, and should check that the timing values are acceptable if the dclk clock frequency is changed.

10.4. External Memory Interface in CompactFlash Mode

Some applications will require an interface to either a CompactFlash device such as a standard CF memory card or IBM Microdrive. Other applications will interface to a standard IDE/ATAPI device like a CD-ROM or hard disk. The system designer may also want to interface to an external SDRAM, DRAM or NOR Flash for bulk data storage. All of these are possible with the CompactFlash mode of the external Memory Interface and, occasionally, some external glue logic like a CPLD.

This interface mode differs significantly from the SmartMedia/NAND mode in that it doesn't have the intelligence to closely control any device family at a high level. All of the control pins must be configured at the register level. High-level command state machines are implemented in software rather than the memory interface. This requires additional system software development. The benefit is significantly increased flexibility. The CompactFlash mode covers a large number of interfaces.

10.4.1. CompactFlash Modes

The three CompactFlash (CF) modes are Memory Mode, I/O Mode and ATA/IDE Mode. The STMP35xx supports the 8 bit Memory Mode in the 100-pin package. In the 144-pin fpBGA, the STMP35xx supports 16 bit Memory Mode and ATA/IDE mode. It is recommended that all applications using the CompactFlash interface use the Memory Mode and not the I/O mode.

The ATA mode can be used to connect hard disk drives and CD-ROM drives.

10.4.2. CompactFlash Registers

10.4.2.1. CompactFlash Control Register

HW_FLCFCR X:$F008

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MODE16	DASP	VS		CRE	CS		XDDI	CFAI		INCS	ISCS	IFCS	INCE	ISCE	IFCE	RI	XWT	CRST	XATTR	SM		CDP	WP

Table 119. HW_FLCFCR

BITS	LABEL	RW	RESET	DESCRIPTION
23	**MODE16**	RW	0	CompactFlash data bus is 16-bits
22	**DASP**	R	unknown	DASP pin of the CompactFlash card, if connected.
21:20	**VS**	R	unknown	VS2, VS1 voltage sense inputs (if connected)
19	**CRE**	RW	EMC5600_ CARDRSTEN_OUT	Output enable for RESET/-RESET pin
18:17	**CS**	RW	00	Active-high versions of outgoing -CE2,-CE1/-CS1,CS0 chip selects
16	**XDDI**	R	unknown	Incoming -PDIAGN/-STSCHG pin, sampled into dclk domain
15:14	**CFAI**	RW	10	Multi-byte transfer card address increment/toggle 00 next A10-A0 same as previous 01 next A10-A0 equal to previous plus one 10 next A10-A0 equal to previous plus two (CIS) 11 next A10-A0 equal to previous, but A0 toggles
13	**INCS**	RW	0	Reading a one indicates a pending interrupt; writing back a one **de-asserts** the interrupt; writing zero has no effect
12	**ISCS**	RW	0	Reading a one indicates a pending interrupt; writing back a one **de-asserts** the interrupt; writing zero has no effect
11	**IFCS**	RW	0	Reading a one indicates a pending interrupt; writing back a one **de-asserts** the interrupt; writing zero has no effect
10	**INCE**	RW	0	Setting this bit enables the card absence/removal interrupt
9	**ISCE**	RW	0	Setting this bit asserts an interrupt based on the -PDIAGN/-STSCHG pin of the CompactFlash card going low in any PC Card Mode
8	**IFCE**	RW	0	Setting this bit passes on a card-based interrupt from the RDY/-BSY/-IREQ/INTRQ pin of the CompactFlash card. The interrupt occurs if the input pin transitions (either direction) in any PC Card socket mode (Memory, I/O, True IDE)
7	**RI**	R	unknown	RDY/-BSY/-IREQ/INTRQ pin of the CompactFlash card, sampled into dclk domain
6	**XWT**	R	unknown	-WAIT active-low card pin of the CompactFlash card, sampled into dclk domain 0 wait/card is busy 1 card/data is available
5	**CRST**	RW	EMC5600_ CARDRST_OUT	Output value to RESET/-RESET pin of the CompactFlash card. Active-high in Memory or I/O Mode; Active-low in True IDE Mode.
4	**XATTR**	RW	0	0 Access to Attribute Memory or I/O space 1 Access to Common Memory

Table 120. CompactFlash Control Register Description

BITS	LABEL	RW	RESET	DESCRIPTION
3:2	**SM**	RW	2'b00	Card access mode 00 PC Card Memory Mode 01 PC Card I/O Mode 10 True IDE Mode 11 reserved
1	**CDP**	RW	EMC5600_PUP	Card Detect pin (-CD1) pull-up enable/disable
0	**WP**	R	unknown	Write Protect (WP) pin of CompactFlash card, sampled into dclk domain

Table 120. CompactFlash Control Register Description (Continued)

10.4.2.2. CompactFlash Timer1 Register

HW_FLCFTMR1R X:$F009

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TRWH					TWPW							TRPW								TRWSU			

Table 121. HW_FLCTMR1R

BITS	LABEL	RW	RESET	DESCRIPTION
23:19	**TRWH**	RW	00001	Hold time for -CE, An, -REG pins of the CompactFlash card, relative to -WE, -RE, -IOWR or -IORD de-asserting, in dclk cycles
18:12	**TWPW**	RW	0001010	-WE, -IOWR pins of the CompactFlash card pulse width, in dclk cycles
11:5	**TRPW**	RW	0001010	-OE, -IORD pins of the CompactFlash card pulse width, in dclk cycles
4:0	**TRWSU**	RW	00001	Read or write pin of the CompactFlash card setup time, in dclk cycles

Table 122. CompactFlash Timer1 Register Description

10.4.2.3. CompactFlash Timer2 Register

HW_FLCFTMR2R X:$F00A

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TRAQ					THW					TWTO										TWW			

Table 123. HW_FLCFTMR2R

BITS	LABEL	RW	RESET	DESCRIPTION
23:19	**TRAQ**	RW	000101	Delay between -OE/-IORD de-assertion and EMC interface re-acquiring the data bus, in dclk cycles
18:14	**THW**	RW	00001	Hold time between -WAIT de-asserting and read or write strobe (-OE, -WE, etc.) de-asserting, dclk cycles
13:4	**TWTO**	RW	$0C8	Timeout waiting for card-imposed wait (-WAIT low) period, in dclk cycles
3:0	**TWW**	RW	0100	Wait-for-wait time, in dclk cycles. Time for -WAIT to assert, from -OE/ -WE/-IOWR/-IORD assertion edge.

Table 124. CompactFlash Timer2 Register Description

11. GENERAL PURPOSE FLASH CONTROLLER

The General Purpose Flash Interface controller (GP Flash) is a DSP configurable interface to external NAND Flash. This highly configurable and flexible interface can attach to and utilize most readily available NAND Flash devices, including the newer large block erase 1Gbit per die devices. The GP Flash is a DMA device that can autonomously transfer NAND Flash pages to or from on-chip RAM as shown in Figure 40.

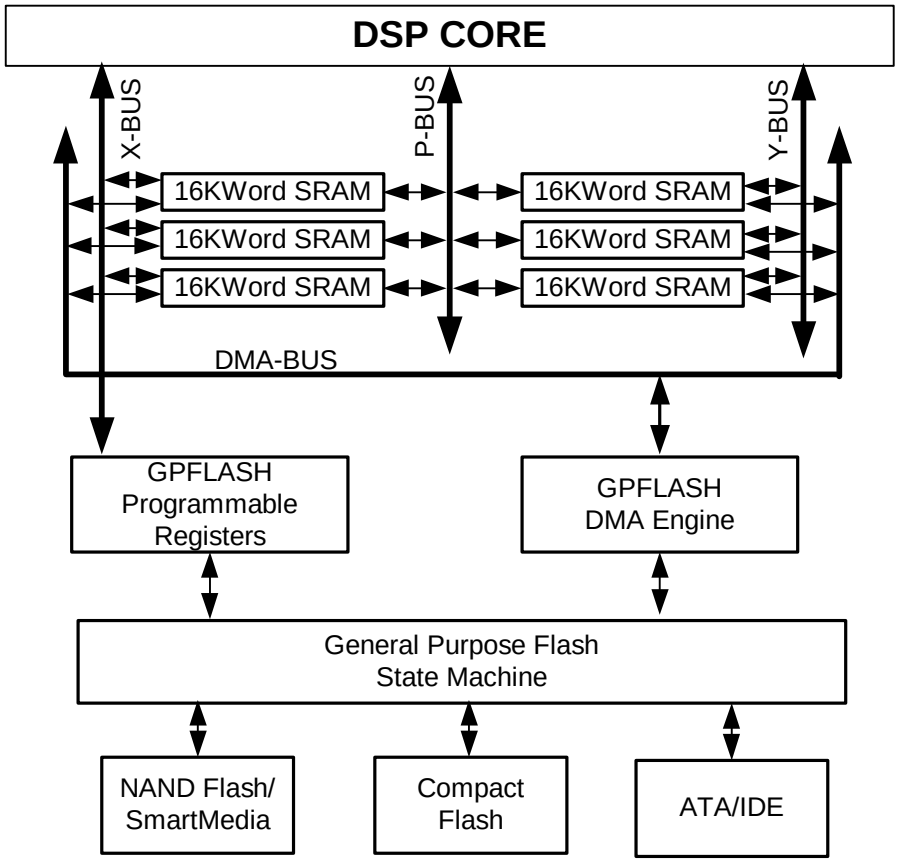


Figure 40. General Purpose Flash Controller

The GP Flash interface provides automatic timing control for the critical data read and write access signal lines. The interface automatically maintains proper CLE and ALE setup and hold time as well as proper read/write strobe to data setup and hold times making automatic transfers via DMA practical. see Figure 41. “GP Flash Command/Address/Write Data Timing” on page 117. Also see Figure 42. “GP Flash Read Timing” on page 117. These diagrams show the fundamental constraints maintained by the interface controller. In addition, it shows the programmable registers and bit fields that control the various critical times in the NAND Flash cycle.

The GP Flash DMA will transfer the data to/form the system X/Y/P memory. The external data bus can be configured to be 8-bit or 16-bit access.

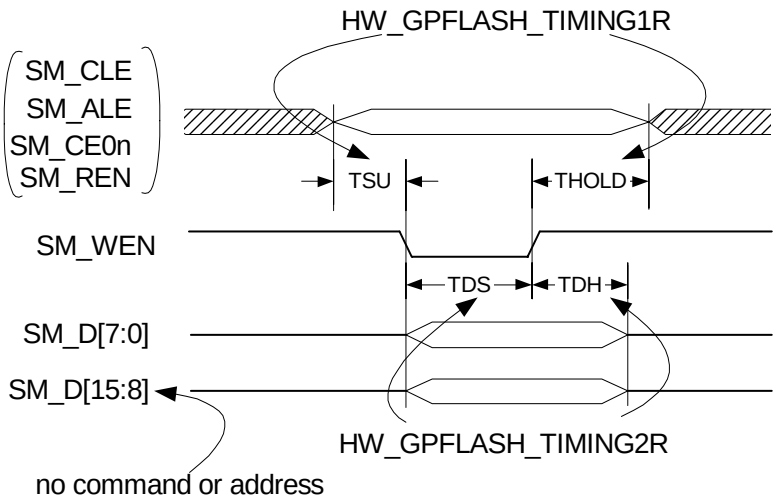


Figure 41. GP Flash Command/Address/Write Data Timing

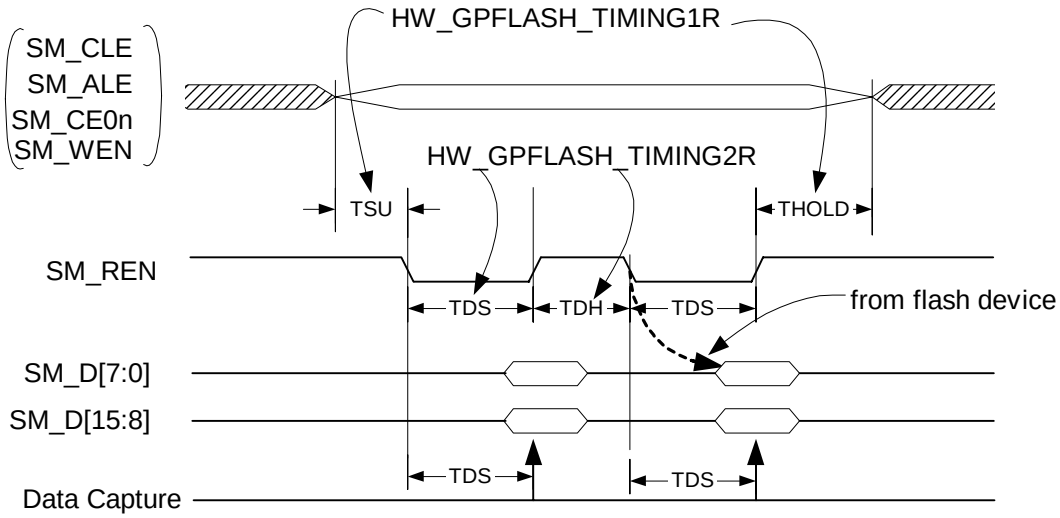


Figure 42. GP Flash Read Timing

Every transfer over the external Flash data bus travels over the DMA bus to or from the on-chip RAM. This includes the address bytes *and* it includes the command byte or bytes sent with CLE asserted. The GPFlash interface controller separates the three phases of Flash device access into three separate DMA transactions, one for the command byte, one for the address bytes and a third one for moving the actual data bytes to or from the device. Thus one must “kick-off” a DMA transaction to send the read command byte itself and another DMA to send the address bytes for the read. see Figure 43. “GP Flash Command Timing” on page 118.

There are two sets of setup and hold control counters. One set is for maintaining setup and hold constraints on the command latch enable (CLE)/address latch enable (ALE). A second set of counters is for managing constraints for the data transfer setup and hold time. The ALE/CLE setup and hold times are controlled in **HW_GPFLASH_TIMING1R** while the data setup and hold times are controlled in **HW_GPFLASH_TIMING2R**.

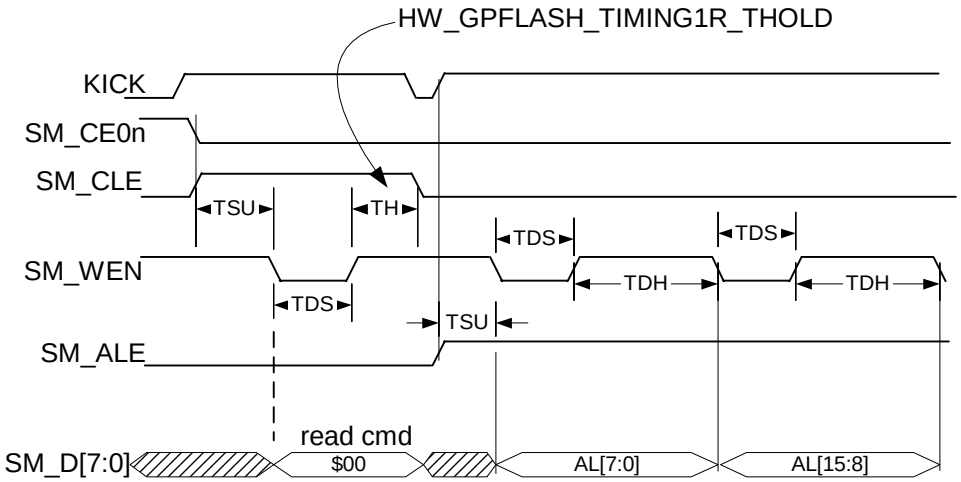


Figure 43. GP Flash Command Timing

The kick bit in **HW_GPFLASHCSR0R** is used to kick off a DMA transfer. Bits in the CSR0R and CSR1R registers permit the definition of a command transfer, an address transfer or a data read or write transfer. All are initiated or kicked-off by setting the KICK bit. Software monitors the state of the KICK bit either by polling or by waiting on a transfer done interrupt. Figure 43 shows the KICK bit being set by software for the command transfer phase and being reset by hardware at the completion of the single byte command transfer. It shows the KICK bit being set again to transfer the address bytes. Of course, the **HW_GPFLASHCSR0R_XFER_TYPE** bits were changed from COMMAND to ADDRESS, and the **HW_GPFLASH_DMA_ADDR_DMA_ADD** field was set to point to the first address byte before this second “kick” was performed. In addition, the **HW_GPFLASH_XFER_SIZE_NUM_BYTES** field was changed from one to four or whatever was appropriate for the device being read.

We see that even to send a single command byte, CSR0, CSR1, have to be setup, the **NUM_BYTES** field has to be set to one, the DMA_ADD field must point to an on-chip RAM byte containing the proper command *and* a DMA must be kicked-off.

Figure 44 shows a complete transaction consisting of a one byte command DMA transfer, a two byte address DMA transfer and a two byte data read DMA transfer. Many NAND Flash devices expect the chip enable to remain low during all three phases, as in Figure 44. The GP Flash state machines respond to the **HW_GPFLASH_CSR0R_CEB_CTRL** bit by leaving the selected chip enable pin in the state indicated by this bit at the end of a DMA transfer. Figure 44 shows the effect of leaving this bit low for the command and address DMA transfers but setting it before the data read DMA transfer. Thus it ends up high (device disabled) after the complete three phase transaction. CEB is brought high after its hold time constraint is met, as specified in **HW_GPFLASH_TIMER1R_THOLD**. More than one data phase DMA transaction can be used while keeping the flash device selected.

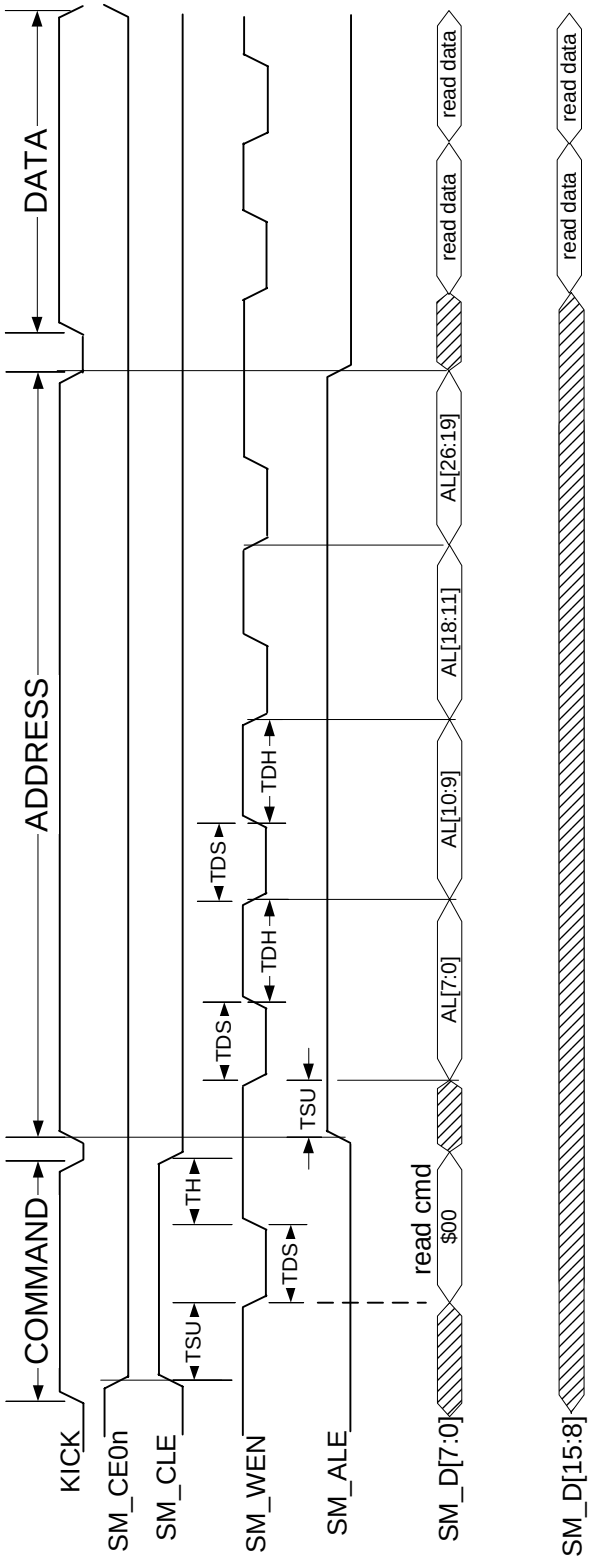


Figure 44. GP Flash Command, Address and Read Data Operations

11.1. Fractional Word Writes

Whenever a write to on-chip RAM ends at a non-word boundary then a fractional word write occurs. This can arise from setting **HW_GPFLASH_CSR1_START_BYTE** to start transferring either to the Intermediate or most significant byte (byte one or byte two) while setting **HW_GPFLASH_XFERSIZER** to an even word size, i.e. byte count is a multiple of three. It can also arise from setting **HW_GPFLASH_CSR1_START_BYTE** to start transferring with the least significant byte (byte zero is the aligned case) while setting **HW_GPFLASH_XFERSIZER** to a value that is not evenly divisible by three.

In either of these cases, the last word of the transfer is only partially filled from the GP FLASH, see Figure 45. A read-modify-write cycle is performed by the GP FLASH hardware for the last word so that the one or two bytes that are not targeted by the GP FLASH transfer remain unmodified in on-chip RAM. Similarly for a partial first word write, i.e. with an unaligned start byte, a read-modify-write cycle is performed to maintain the values of the untargeted bytes.

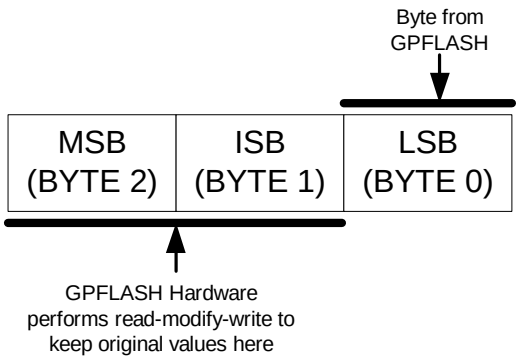


Figure 45. GP Flash First or Last Partial Word Transfer

11.2. Programming Example

There are certain settings that must be made when the GP FLASH module is first initialized. The initialization flow chart provides an example of such an initialization sequence, see Figure 46. “GP Flash Example Initialization Flowchart” on page 121.

First, clocks are turned on and software reset is removed to enable the use of the GP Flash. Next the timing control bit fields are initialized. The NAND Flash device specification should be consulted to determine the timing specifications for the device in absolute time, i.e. nano-seconds. Next these parameters must be converted to DCLK cycle counts by dividing each timing specification by the period of the intended DCLK frequency. The ceiling of this number must be used for minimum specifications. The resulting integer value is then loaded into the appropriate bit field.

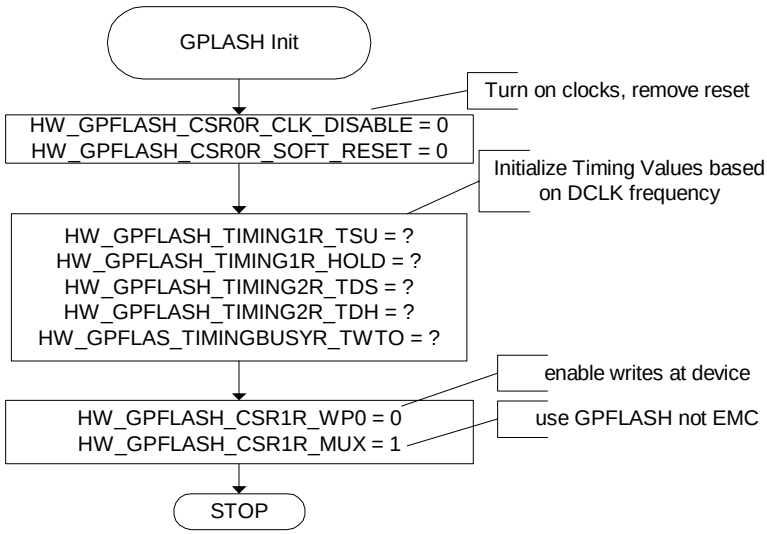


Figure 46. GP Flash Example Initialization Flowchart

Consider the full transfer timing diagram shown above, see Figure 44. “GP Flash Command, Address and Read Data Operations” on page 119. The flowchart of Figure 47 outlines a function designed to generate this sequence and read a page of data bytes from a NAND Flash device and write them into a buffer in on-chip RAM.

Note that the one-byte read command is read from a location in on-chip RAM via DMA and written to the NAND Flash device with the command latch enable (CLE) signal asserted. As shown in the flowchart, three separate transfers are “kicked-off” to effect reading data bytes from a NAND Flash page.

The four byte NAND Flash address is read from a buffer in on-chip RAM and written to the device. For this transfer, the CLE signal has been negated and the address latch enable signal (ALE) is asserted. The ALE/CLE setup and hold timings are automatically guaranteed by the hardware. Finally, the device is configured for a data read transfer, depositing the NAND Flash data into a buffer in on-chip RAM.

The example provided in the flowchart of Figure 47 uses polling to determine when the GP FLASH hardware has completed a transfer that was initiated by setting the KICK bit. While this method functions correctly, it wastes significant time in busy waiting using polling.

It is much better to use the transfer done (XFERDONE) interrupt to monitor completion of each phase of the transaction, see Figure 48. “GP Flash Send Command using XFERDONE Interrupt” on page 123.

This flowchart shows only the command byte transfer. Address and data transfers using XFERDONE interrupts are implemented in a similar manner. Notice that the ISR will have to have a small amount of state information to determine what action to take on a completion interrupt, e.g. after a command transfer, initiate the address transfer, etc.

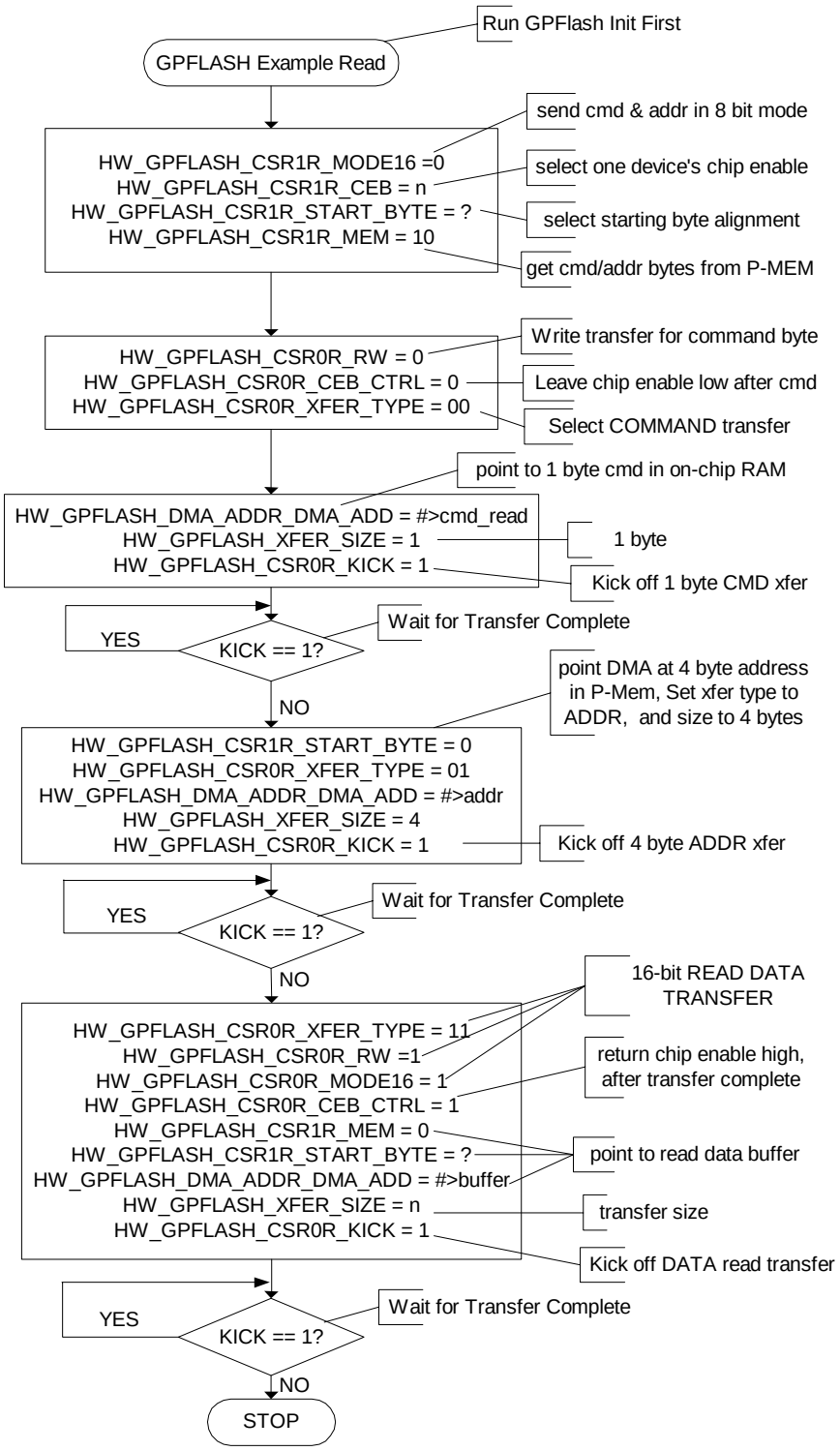


Figure 47. GP Flash Read Data Example Flowchart

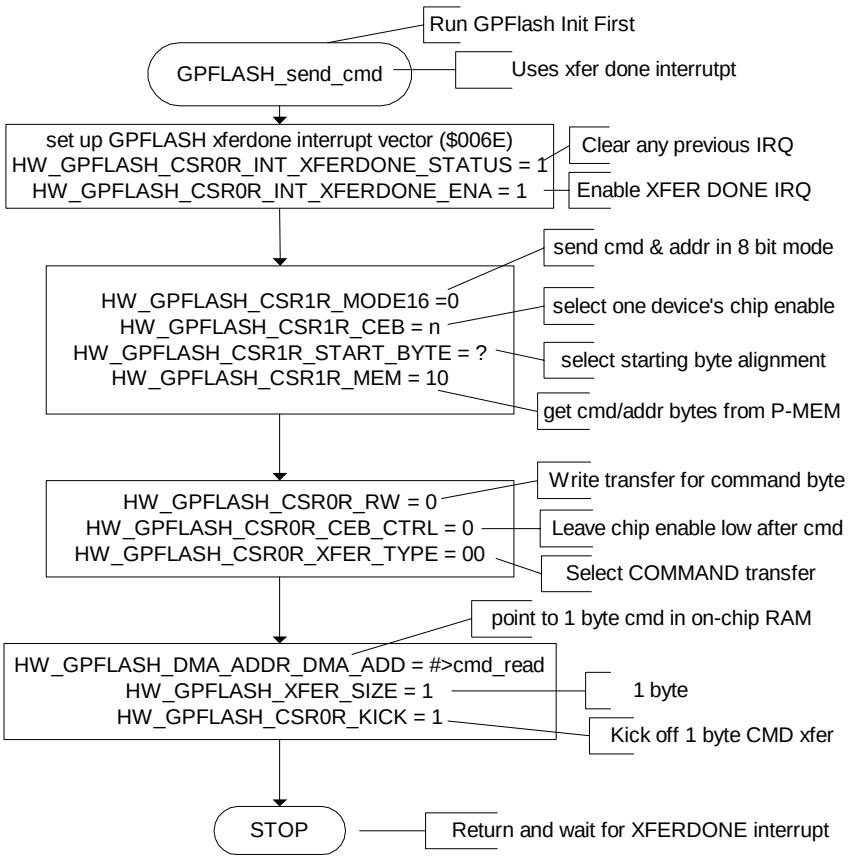


Figure 48. GP Flash Send Command using XFERDONE Interrupt

11.3. Monitoring the NAND Flash Ready/Busy Signal

The GP FLASH module contains several circuits to aid in monitoring the relatively long interval transitions of the NAND Flash device's Ready/Busy signal. This includes an interrupt that can monitor either the rising or falling edge of the busy signal and a TWTO timer that can be set to generate a timeout interrupt if the Flash device hangs and never completes a block erase, etc.

The busy interrupt monitoring interrupt should only be scheduled after a command that will report its availability via the R/B signal long after the command is started. WARNING: not all NAND Flash devices have reliable busy signaling. For some devices, only read status polling from a timer interrupt schedule is practical.

11.4. GP FLASH Programmable Registers

All GP Flash interface controller programmable registers exist in the DSP's X memory space as shown below.

11.4.1. General Purpose Flash Control Status Register 0

This register provides control of the General Purpose Flash Controller.

HW_GPFLASH_CSR0R X:$F0C0

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			EXT_BUSY_VAL	SPARE		XFER_TYPE			BUSY_TIMEOUT_INT_STAT	CLK_DISABLE	CEB_CTRL	SOFT_RESET	BUSY_INT_STATUS	BUSY_TIMEOUT_INT_ENA	BUSY_INT_ENA	BUSY_TIMEOUT_EDGE		BUSY_INT_TYPE		INT_XFERDONE_STATUS	INT_XFERDONE_ENA	RW	KICK

Table 125. HW_GPFLASH_CSR0R

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	000	**Reserved –** Must be written with 0.
20	**EXT_BUSY_VAL**	R	0	This read only bit samples the state of the external busy pin.
19:18	**SPARE**	RW	00	These bits, while present in the register, are currently unconnected in the design and are reserved as patch gates.
17:16	**XFER_TYPE**	RW	00	The GP FLASH breaks up the normal transaction flow to the FLASH into one transaction for sending the command, one transaction for sending any address bytes and one or more commands for transferring read or write data. This field determines which type of transaction is being "kicked off". 00 = command 01 = address 10 = data 11 = reserved
15	**RSRVD**	R	0	**Reserved –** Must be written with 0.
14	**BUSY_TIMEOUT_INT_STAT**	RW	0	Reading a one indicates a pending busy completion TIMED OUT instead of completing normally. This is a sticky bit which is cleared by writing back a one in this bit position. Writing zero has no effect. This bit is ANDed with **BUSY_TIMEOUT_IN_ENA** to generate a BUSY_TIMEOUT_IRQ. BUSY_TIMEOUT_IRQ is ORed with BUSY_INT_IRQ to generate the GPFLASH busy interrupt on source 6 and vector $0072.
13	**CLK_DISABLE**	RW	1	When this bit is set to one, the GP FLASH controller clocks are gated off to reduce power.

Table 126. GP FLASH Command/Status Register 0 Description

BITS	LABEL	RW	RESET	DEFINITION
12	**CEB_CTRL**	RW	1	Setting the chip enable control bit to one allows the chip enable to return to the high state at the end of a transfer. Setting this bit to zero will cause the chip enable to remain low after a transfer. Keeping it low allows multiple DMA transactions to look like a single contiguous transfer to the NAND FLASH device.
11	**SOFT_RESET**	RW	0	Writing a one to this bit position forces a reset to most of the counters and state machines in the General Purpose Flash Controller. This bit must be reset to zero by software for normal operation of the GP FLASH. Programmable registers are **NOT** returned to their default state by **SOFT_RESET**.
10	**BUSY_INT_STATUS**	RW	0	Reading a one indicates a pending busy completion or busy timeout interrupt has occurred. This is a sticky bit which is cleared by writing back a one in this bit position. Writing zero has no effect. This bit is ANDed with **BUSY_IN_ENA** to generate a BUSY_IRQ. BUSY_IRQ is ORed with BUSY_TIMEOUT_IRQ to generate the GP FLASH busy interrupt on source 4 and vector $006E.
9	**BUSY_TIMEOUT_INT_ENA**	RW	0	Setting this bit to one enables the busy timeout state machine and counter. If the trailing edge of busy is not detected then BUSY_TIMEOUT_INT_STAT is set.
8	**BUSY_INT_ENA**	RW	0	Setting this bit to one enables a busy interrupt in **BUSY_INT_STATUS**
7:6	**BUSY_TIMEOUT_EDGE**	RW	00	This field selects the edge behavior of the busy time out interrupt state machine. The 13 bit internal timer counter is reloaded from TXTO at the leading edge. The timer counts down the width of the busy pulse. If the trailing edge is seen then the timeout is suppressed. If the internal timer times out then the **BUSY_TIMEOUT_INT_STAT** bit is set. The leading/trailing edged definition depends on the polarity of the busy signal. 00 = Reload timer on both edges 01 = Reload timer on rising edges, trailing is falling. 10 = Reload timer on falling edge, trailing is rising (NAND). 11 = Test mode
5:4	**BUSY_INT_TYPE**	RW	00	This field selects the edge behavior of the busy interrupt, as follows: 00 = trigger interrupt on both edges 01 = trigger interrupt on rising edge only 10 = trigger interrupt on falling edge only 11 = undefined behavior
3	**INT_XFERDONE_STATUS**	RW	0	Reading a one indicates a pending transfer complete interrupt has occurred. This is a sticky bit which is cleared by writing back a one in this bit position. Writing zero has no effect.
2	**INT_XFERDONE_ENA**	RW	0	Setting this bit to one enables the transfer-complete interrupt.
1	**RW**	RW	0	Setting this bit to one indicates that the external memory transfer is a read; otherwise it is a write.
0	**KICK**	RW	0	Setting this bit to one initiates an external memory transfer; it automatically clears when the transfer completes. It can be polled by software to detect transfer completion or more typically, **HW_GPFLASH_CSR0R_INT_XFERDONE_STATUS** interrupt can be used to detect this condition.

Table 126. GP FLASH Command/Status Register 0 Description (Continued)

11.4.2. General Purpose Flash Control Status Register 1

This register provides control of the General Purpose Flash Controller.

HW_GPFLASH_CSR1 X:$F0C1

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
				WP_CTL	SPARE_AREA_EN	MUX	MODE16			WP1	WP0			CEB				START_BYTE				MEM	

Table 127. HW_GPFLASH_CSR1

BITS	LABEL	RW	RESET	DEFINITION
23:20	**RSRVD**	R	0000	**Reserved –** Must be written with 0.
19	**WP_CTL**	RW	0	This bit controls a multiplexor that selects the source for the SPI MOSI output pin. In order to use the WP1 write protect signal, this bit must be set to one. Setting it to zero returns control of the MOSI driver to the SPI.
18	**SPARE_AREA_EN**	RW	1	The spare area enable pin is essentially a GPIO pin that is intended to drive the SA input pin on NAND FLASH devices that provide this option. It drives SDRAM Address pin 8 when the GP Flash controls the external memory interface pins.
17	**MUX**	RW	0	GP Flash pins are shared with a number of other integrated interface controllers. This bit controls a multiplexor that selects between the GP Flash controller and the External Memory Controller (EMC), as follows: 0 = **HW_GPFLASH_MUX_EMC_ACCESS** 1 = **HW_GPFLASH_MUX_GPFLASH_ACCESS**
16	**MODE16**	RW	0	The GPFLASH controller supports either eight or sixteen bit Flash devices. Set this pin to one for 16 bit devices. 0 = **HW_GPFLASH_8BIT_ACCESS** 1 = **HW_GPFLASH_16BIT_ACCESS**
15:14	**RSRVD**	R	00	**Reserved –** Must be written with 0.
13	**WP1**		0	The write protect 1 bit is essentially a GPIO pin that is intended to connect to the write protect pin of a second NAND Flash device. This pin is shared with the MOSI pin, see **WP_CTL**.
12	**WP0**		0	The write protect 0 bit is essentially a GPIO pin, it is intended to connect to the write protect pin on the NAND Flash.
11:10	**RSRVD**	R	00	**Reserved –** Must be written with 0.

Table 128. GP FLASH Command/Status Register 1 Description

BITS	LABEL	RW	RESET	DEFINITION
9:8	**CEB**	RW	00	The interface allows access to four external NAND Flash devices. The chip enable selector field is decoded to pick one of the four external chip enable pins. Only the selected pin will be driven to zero. 00 = CE0 01 = CE1 10 = CE2 11 = CE3
7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5:4	**START_BYTE**		00	This field the byte alignment of the starting byte 00 = **HW_GPFLASH_START_BYTE_LSB** 01 = **HW_GPFLASH_START_BYTE_ISB** 10 = **HW_GPFLASH_START_BYTE_MSB** 11 = reserved, undefined behavior
3:2	**RSRVD**	R	00	**Reserved –** Must be written with 0.
1:0	**MEM**		00	This field defines the DSP memory space used for DMA transfers as follows: 00 = X Memory **HW_GPFLASH_USE_X_MEMORY** 01 = Y Memory **HW_GPFLASH_USE_Y_MEMORY** 10 = P Memory **HW_GPFLASH_USE_P_MEMORY** 11 = reserved, undefined behavior

Table 128. GP FLASH Command/Status Register 1 Description (Continued)

11.4.3. General Purpose Flash DMA Address Register

This register holds the DMA address used by the General Purpose Flash Controller to access on-chip RAM.

HW_GPFLASH_DMA_ADDR X:$F0C2

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								DMA_ADD															

Table 129. HW_GPFLASH_DMA_ADDR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**DMA_ADD**	RW	$0000	Begin the DMA transfer at this on-chip RAM address in the address space specified in **HW_GPFLASH_CSR0R_MEM**.

Table 130. GP FLASH DMA Address Register Description

11.4.4. General Purpose Flash DMA Transfer Size Register

This register holds the transfer count for DMA transactions kicked off for the General Purpose Flash Controller.

HW_GPFLASH_XFER_SIZE X:$F0C3

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											NUM_BYTES												

Table 131. HW_GPFLASH_XFER_SIZE

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:0	**NUM_BYTES**	RW	$001	This field contains the number of bytes to be transferred by the GP Flash controller, when “kicked”.

Table 132. GP FLASH DMA Transfer Size Register Description

11.4.5. General Purpose Flash Timing 1 Register

This register holds part of the timing values for specifying read or write cycles to or from an external NAND Flash when using the General Purpose Flash Controller.

HW_GPFLASH_TIMING1 X:$F0C4

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
												THOLD								TSU			

Table 133. HW_GPFLASH_TIMING1

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:8	**THOLD**	RW	$01	After the last byte of command, address or data is transferred, the CLE or ALE signals are returned to zero, if they were set. The CLE/ALE hold time constrained is guaranteed by the GPFlash state machines counting **THOLD** DCLKs before releasing the ALE or CLE signals. THOLD can not be set to zero.
7:5	**RSRVD**	R	00	**Reserved –** Must be written with 0.
4:0	**TSU**	RW	$00	Immediately after a transfer is kicked off, CSR0R is examined and the CLE/ALE signals are set as required for the transfer. The GPFlash then counts **TSU** DCLKs before allowing the first read or write enable (REN/WEN) pulse.

Table 134. GP FLASH Timing 1 Register Description

11.4.6. General Purpose Flash Timing 2 Register

This register holds part of the timing values for specifying read or write cycles to or from an external NAND Flash when using the General Purpose Flash Controller. The GPFlash state machines drive the write enable signal low at the same time they drive new data onto the eight or sixteen bit data bus. Setup time for the flash device is guaranteed by holding the write enable low for a sufficient number of DCLKs. Flash device hold time is guaranteed after the write enable is returned to one by keeping the data bus value stable for a sufficient number of DCLKs. These counts are specified in the HW_GPFLASH_TIMING2R register.

In addition, setup time for read data arriving back to the STMP35xx and its necessary hold time is also controlled in this register, see Figure 42. “GP Flash Read Timing” on page 117.

HW_GPFLASH_TIMING2 X:$F0C5

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
										TDH								TDS					

Table 135. HW_GPFLASH_TIMING2

BITS	LABEL	RW	RESET	DEFINITION
23:14	**RSRVD**	R	0	**Reserved –** Must be written with 0.
13:8	**TDH**	RW	01	Number of DCLKS to hold off starting another write or read enable low sequence. In the case of writes, this keeps the data bus stable, as driven to the flash device. In the case of reads, this keeps the device driving the same data to the STMP35xx until its hold time is satisfied. TDH can not be set to a value of zero.
7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5:0	**TDS**	RW	01	Number of DCLKs to hold the write enable or read enable low. Can not be set to zero.

Table 136. GP FLASH Timing 2 Register Description

11.4.7. General Purpose Flash TIMING BUSY Register

This register holds part of the timing values for the General Purpose Flash Controller. Flash devices have a signal called ready/not busy that indicates when the device is busy erasing a block of NAND Flash, when it is programming new values into the NAND Flash array and in some cases, when it is reading the next page of data from the NAND Flash array. There are two ways to determine when the flash device is finished, i.e. no longer busy. One is to perform a device status read command. This is most reliable method across all flash devices. The second method is to monitor the state of the ready/not-busy line. The flash devices have open drain drivers with pull-ups on these signals so that not-BUSY can be wire *OR*ed between all the devices. After the device is commanded to erase a block, the not-BUSY signal will be pulled low. When the block erase is complete, the flash device releases its driver and the signal is pulled up to indicate “READY” for the next operation. The STMP35xx has an interrupt based monitor circuit for this ready/not-busy signal. The not-busy signal is synchronized into the GP Flash interface controller and then both edges are detected. The HW_GPFLASH_CSR0R_BUSY_INT_STATUS bit indi-

cates the arrival of an edge on the ready/not-busy signal. This bit is *"sticky"*, so once it is set software must write a one to it to reset it.

The HW_GPFLASH_CSR0R_BUSY_INT_TYPE field tells the busy monitor circuit which edge it should monitor. Thus software can choose to be interrupted when the device first goes busy or when it leaves the busy state. Not all devices report busy in robust manner, by design. In addition, broken devices may also fail to either go into the busy state or return from it or to even mark these transitions on the ready/not-busy signal. As a result, the busy monitor circuit includes a time out counter that will ultimately interrupt if the DSP on the BUSY interrupt, even if the device failed to do so. The number of DCLKs to wait for the device to go from busy to not busy is specified in the **TWTO** field below. It tracks the time from the fall of the not-busy signal. The field HW_GPFLASH_CSR0R_BUSY_TIMEOUT_EDGE can be used to start timing on the opposite edge.

HW_GPFLASH_TIMINGBUSY X:$F0C6

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
						TWTO																	

Table 137. HW_GPFLASH_TIMINGBUSY

BITS	LABEL	RW	RESET	DEFINITION
23:18	**RSRVD**	R	0	**Reserved –** Must be written with 0.
17:0	**TWTO**	RW	$00001	Number of DCLKs to wait from the falling edge of ready/not-busy until a time-out should be reported.

Table 138. GP FLASH Timing BUSY Register Description

11.4.8. General Purpose Flash Debug Register

This register provides a read only path to view various state machines in the General Purpose Flash Controller.

HW_GPFLASH_DEBUG X:$F0C7

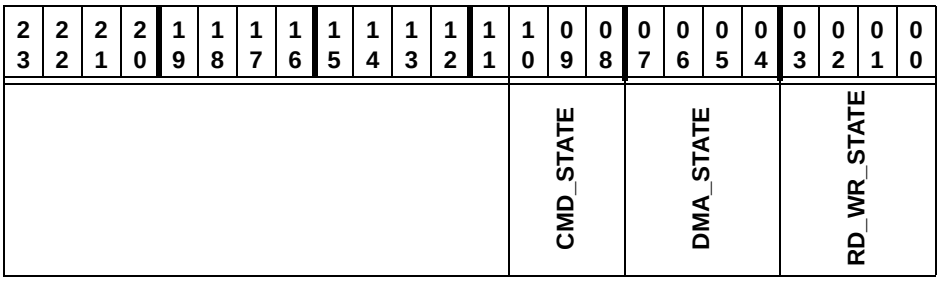


Table 139. HW_GPFLASH_DEBUG

BITS	LABEL	RW	RESET	DEFINITION
23:11	**RSRVD**	R	$0000	**Reserved –** Must be written with 0.
10:8	**CMD_STATE**	R		Read only view of the Command State Machine
7:4	**DMA_STATE**	R		Read only view of the DMA State Machine
3:0	**RD_WR_STATE**	R		Read only view of the Read/Write State Machine.

Table 140. GP FLASH Debug Register Description

11.4.9. General Purpose Flash Busy Timeout Counter Register

This register provides a read only path to view the current value in the Busy Timeout Counter. This counter is preloaded from the **HW_GPFLASH_TIMINGBUSYR** register. It decrements to zero whenever **HW_GPFLASH_CSR0R_BUSY_INT_ENA** is set to one and the leading edge of the busy pulse has been encountered. If the trailing edge of the busy pulse is detected then the counter is reloaded form **HW_GPFLASH_TIMINGBUSYR** and no timeout will be reported.

HW_GPFLASH_TWTOCNT X:$F0C8

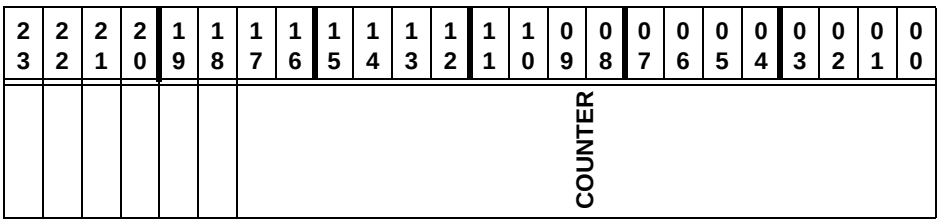


Table 141. HW_GPFLASH_TWTOCNT

BITS	LABEL	RW	RESET	DEFINITION
23:18	**RSRVD**	R	$0000	**Reserved –** Must be written with 0.
17:0	**COUNTER**	R	$00001	Read only view of the Busy Timeout Counter

Table 142. GP FLASH Busy Timeout Counter Register Description

12. FLASH ECC ACCELERATOR

The FLASH ECC Accelerator provides a forward error correction function for improving the reliability of various storage media that can be attached to the STMP35xx. Modern high density NAND Flash devices, for example, presume the existence of forward error correction algorithms. This allows much higher yield and therefore lower cost storage devices by allowing some soft or hard bit errors within the bits of a flash page. The FLASH ECC block comprises two forward error correction algorithms, one based on the Samsung SSFDC hamming code algorithm for single bit error correction and a second, more robust, algorithm for multi-bit error correction Reed-Solomon block codes, see Figure 49. Having a DMA based hardware accelerator for this function allows the DSP to focus on more signal processing algorithms for enhanced functionality or to operate at lower clock frequencies and voltages for improved battery life.

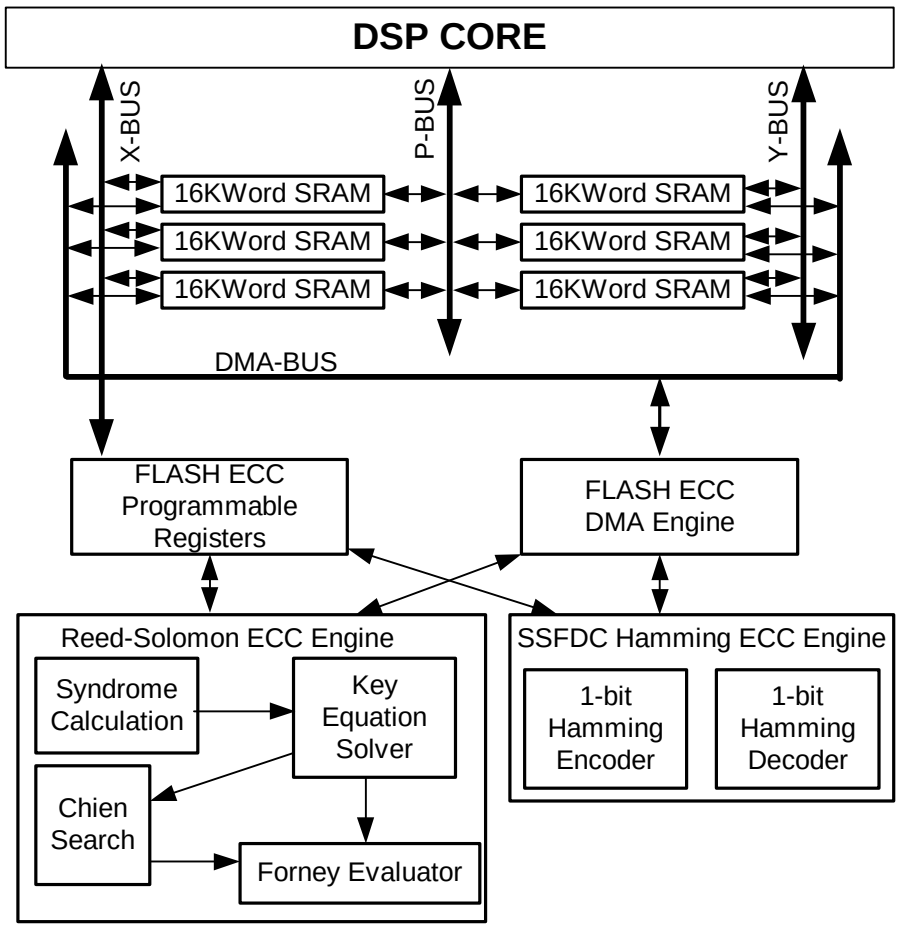


Figure 49. FLASH ECC ACCELERATOR ENGINES

The 1-bit hamming ECC algorithm, as defined by Samsung, is for use with all SSFDC compliant NAND Flash memories. The error coding is capable of correcting a single bit in error over a 256 byte (2048 bit) data block. It generates a 3 byte parity field which is stored in the spare area at the end of a flash page. The Reed-Solomon algorithm is capable of correcting up to 4 9-bit symbols in a 512 byte block. Thus up to 36 bits in error can be corrected in a 512 byte block, provided they are clustered

within no more than four 9 bit symbols. This algorithm generates 9 bytes (8 symbols) of error code or parity (sometimes called syndromes) per 512 byte block. The parity bytes are stored in the spare area at the end of each NAND Flash page.

The Reed-Solomon algorithm takes a significantly larger number of cycles to correct an error than the SSFDC Hamming algorithm but can correct more errors per page. In either case, the DSP is *not* directly involved in checking for the errors nor is it directly involved in correcting errors that may be found.

12.1. Reed-Solomon ECC Accelerator

Consider the case where there is a 512 byte data block located in the on-chip RAM that needs to be written to a NAND Flash device. Further, assume that a 9 byte Reed-Solomon parity field is to be written into the 16 byte spare area of the 528 byte NAND Flash page.

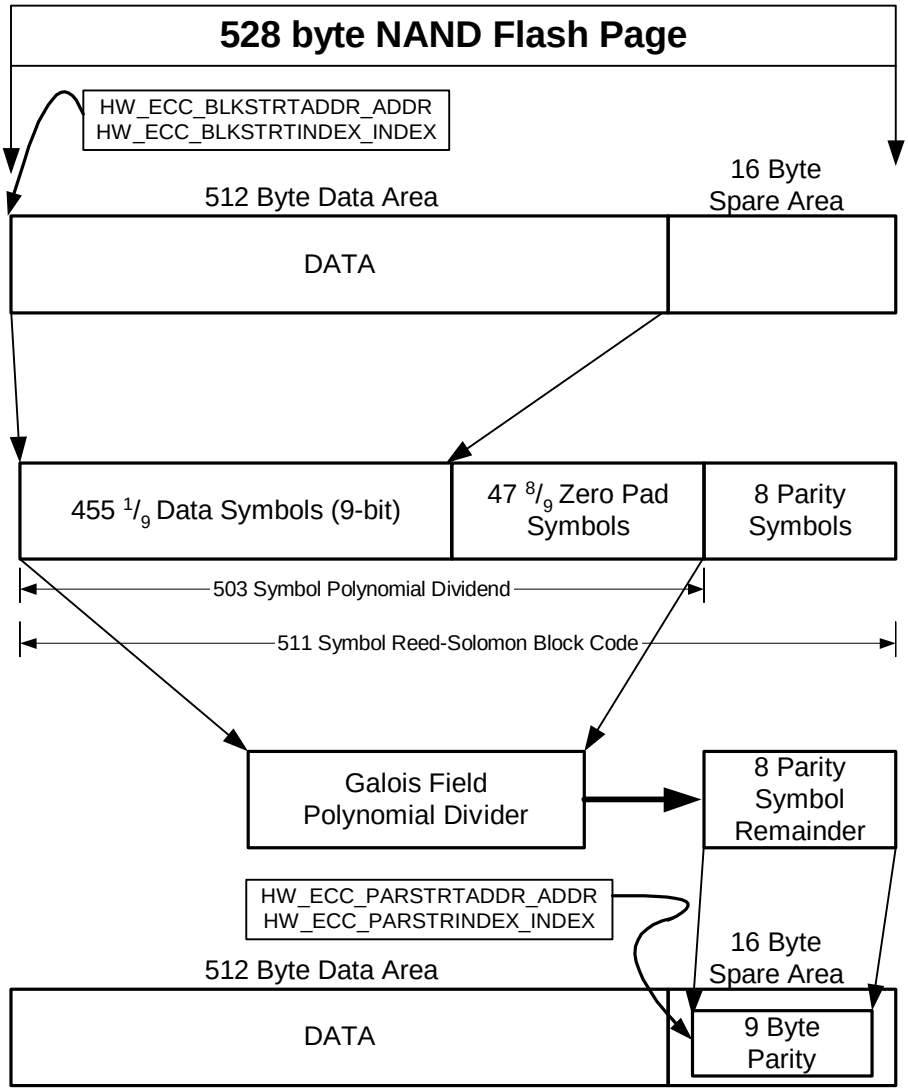


Figure 50. FLASH ECC Reed-Solomon Block Coding- Encoder

Assume that the GPFLASH module will be used to write the resultant 521 bytes of data and parity from on-chip memory to the NAND Flash device. The FLASH ECC module **HW_ECC_BLKSTRTADDR** register is used to point to the data block in on-chip RAM, see Figure 50. “FLASH ECC Reed-Solomon Block Coding- Encoder” on page 133. The Reed-Solomon (RS) algorithm uses 9 bit symbols. Thus a 512 byte data block encompasses 455 1/9 symbols. As the data is read from on-chip RAM, the hardware appends 47 8/9 zero pad symbols to form the basic 511 symbol RS block. This block is treated as a large polynomial and is divided by the hardware using the mathematics of Galois Fields[1]. The hardware retains the 8 symbol (72 bits or 9 bytes) remainder from this division which it then stores as the parity for the block. The parity is stored into on-chip RAM at the address specified in **HW_ECC_PARSTRTADDR**. The GPFLASH DMA can then be started to copy the 521 bytes to the NAND Flash device. Of course, both units can be fully overlapped.

Note that there is an **HW_ECC_BLKSTRTINDEX** register and an **HW_ECC_PARSTRTINDEX** register which allows the data symbols and parity symbols to start on arbitrary bit boundaries within a 24 bit on-chip RAM word.

The RS encoder flowchart shows the detailed steps involved in programming and using the FLASH ECC’s Reed-Solomon encoder, see Figure 51. “FLASH ECC Reed-Solomon Encode Flowchart” on page 135. To use the encoder, one must first turn off the module wide soft reset bit. One then programs the control status register 0 to select the Reed-Solomon encode algorithm, to remove the encoder reset, to enable an interrupt to signal completion of the encode stage, and to specify the DMA wait cycle count. Since the FLASH ECC is a memory to memory DMA device, its DMA utilization is nominally limited only by the encoder’s demand for data. This natural limit may utilize too much DMA bus bandwidth over its command time. As a result, the **HW_ECC_CSR0_DMAWAIT** bit field can specify additional wait cycles to insert between the DMA cycle requests to reduce the FLASH ECC’s short term utilization.

The block start address/index values are next set to point to the data block in on-chip RAM that is to be encoded. The parity start address/index values are set to point to the on-chip RAM buffer that will hold the nine generated bytes until they are written to Flash. Any previous Done Interrupt status is cleared by writing a one to the interrupt bit and the KICK bit is set to one. Software can then poll the KICK bit waiting for it to return to zero, however, this typically takes hundreds of clock cycles. To get full overlap of the DSP and the FLASH ECC module, one uses the “done” interrupt which vectors to $0062, source bit 29. When this interrupt is received, the GPFLASH module can be scheduled to write the entire page to the NAND Flash device.

1.Oliver Pretzel, “Error-Correction Codes and Finite Fields,” Oxford Univ. Press, 1992 ISBN 0-19-269067-1.

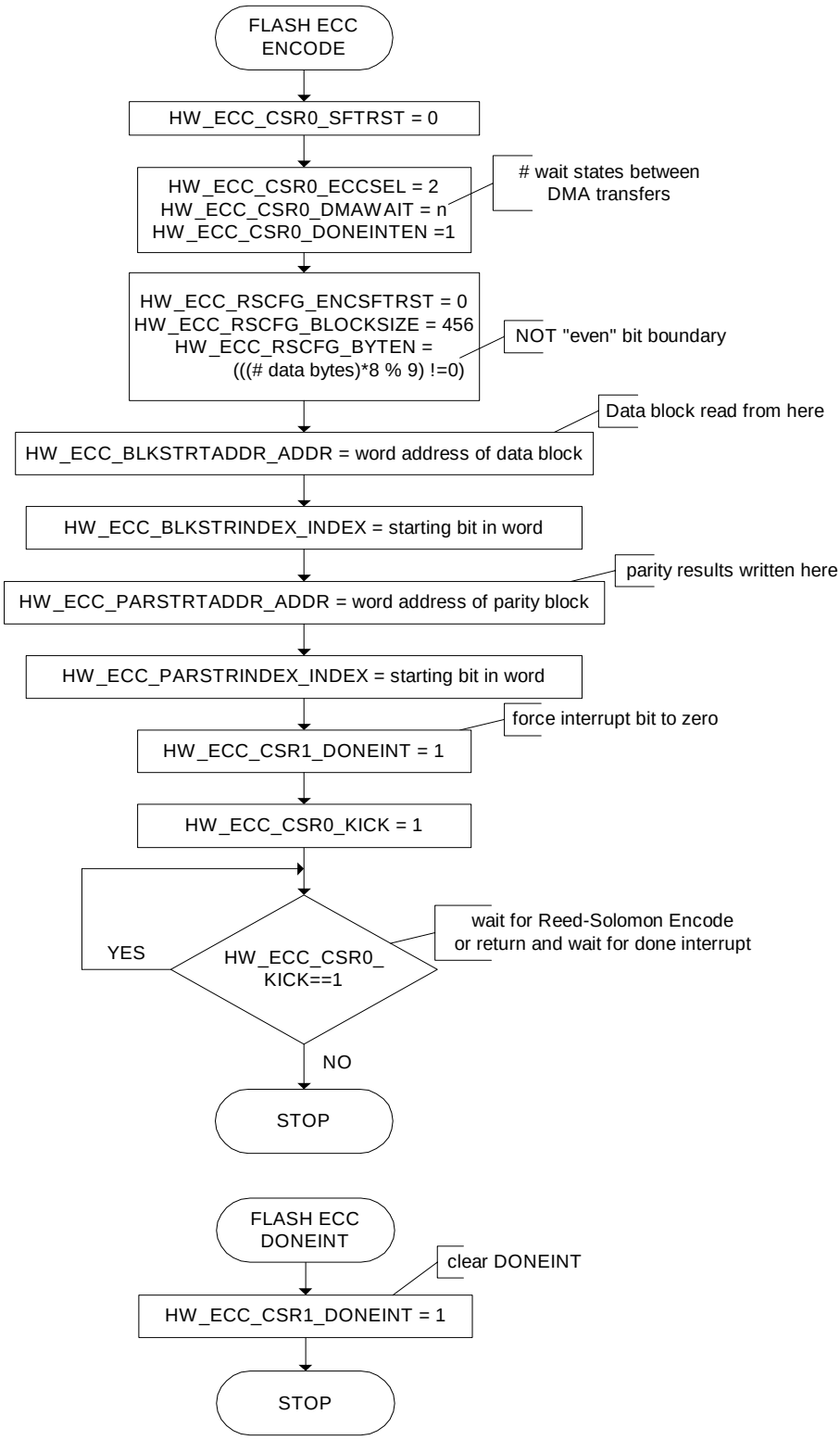


Figure 51. FLASH ECC Reed-Solomon Encode Flowchart

When a page is read from NAND Flash, its RS parity must be checked and if correctable errors are found they must be corrected. This *decoding* process can also be fully overlapped with DSP execution, see Figure 52. “FLASH ECC Reed-Solomon Block Coding- Decoding Phase 1” on page 136. Notice that in this case, both the block start/index registers and the parity start/index registers point into the on-chip RAM buffers where bytes were read from the NAND Flash device. The decoder can be initialized to read the data block, append the zero pad and perform the polynomial division, this time with the supplied parity bytes. If the resulting division yields a zero remainder then no errors are present and the FLASH ECC block can immediately report via its “done” interrupt. If the remainder was non-zero then it further examines the syndrome bytes to correct any errors that may be present, if possible.

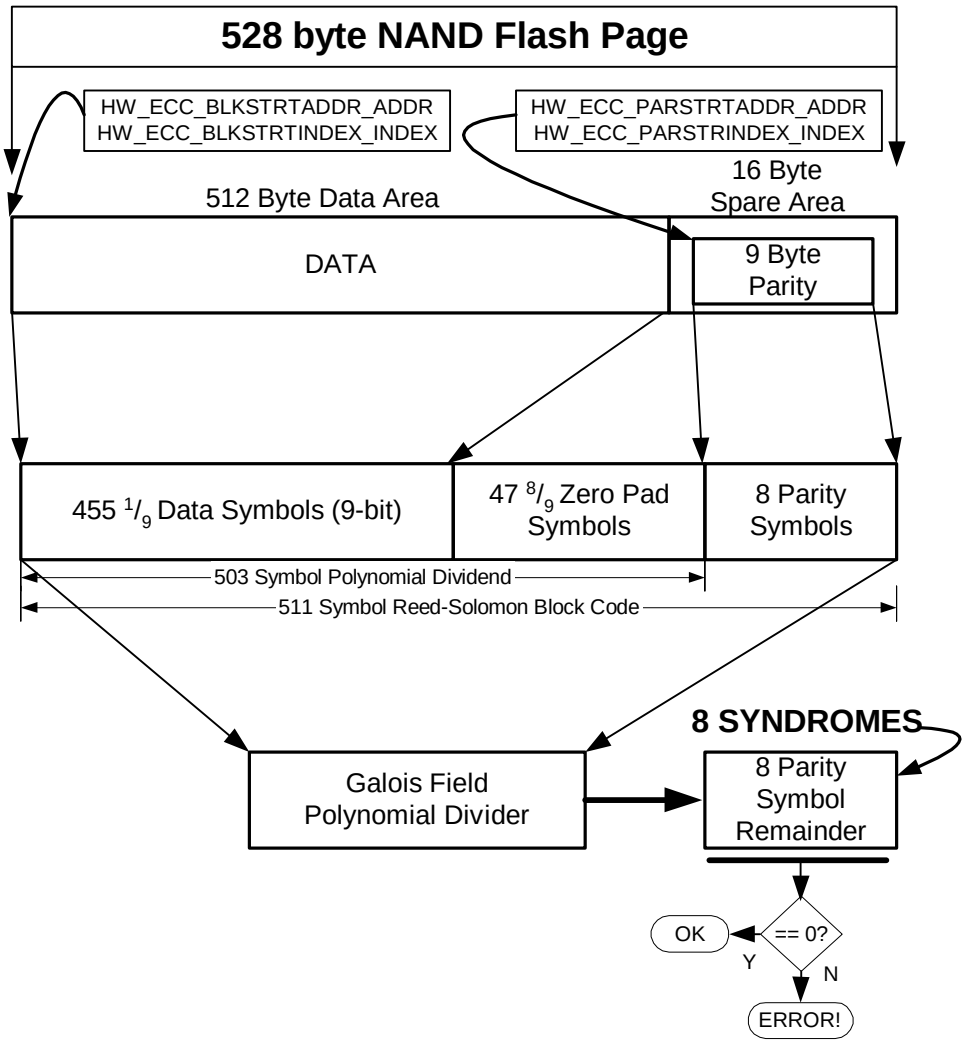


Figure 52. FLASH ECC Reed-Solomon Block Coding- Decoding Phase 1

The RS decoder processes the 511 symbol block code in three phases. Not all of the phases may be necessary, for example when no errors are found or when uncorrectable errors are found. The three phases are:

1. **Syndrome Calculation Phase (SC)**: This is the process of reading in all of the symbols of the block and continuously dividing by the generator polynomial for the field. The eight syndromes are calculated as the remainder of this division and must examined, as described above. This phase takes approximately 700 cycles for a 512 byte data block, with no planned DMA wait states added.
2. **Key Equation Solver Phase (KES)**: Once the eight syndromes have been calculated, a set of eight linear equations in eight unknowns is formed. The process of solving these equations and selecting from the numerous possible solutions constitutes the KES phase. The partial solution is obtained by dividing a polynomial based on the syndromes by a Euclidian polynomial. This division, again using the mathematics of Galois Fields, yields two polynomials, the Error Evaluator (EE) polynomial and the Error Locator (EL) polynomial. The EE polynomial is the remainder of this division and is zero if an uncorrectable, i.e non-solvable case exists. The hardware terminates with an uncorrectable error in this case. This phase takes up to 560 DCLKs, with no planned DMA wait states added.
3. **Chien Search and Forney Evaluator Phase (EVAL)**: This phase takes the EE- and EL polynomials from the KES phase and uses Chien's algorithm for finding the locations of the errors based on the EE-polynomial. The method basically involves substituting all 512 9-bit symbols into the EE-polynomial. All non-zero results of these substitutions represent the locations of the various errors. Another GF division is performed at this point to determine the error value or the correction to apply at the symbol in error location. This phase consumes approximately 550 DCLKs, with no planned DMA wait states added. The EVAL phase terminates either with an uncorrectable error interrupt or simply a "done" interrupt. Done is reported in either case.

The flowchart shows that the initialization of the decoder is quite similar to that for the encoder, see Figure 53. "FLASH ECC Reed-Solomon Decode Flowchart" on page 138. Basic differences are setting ECCSEL to 3 for RS decode and turning off the decoder soft reset instead of the encoder soft reset.

There are a few critical differences, however, first one also enables the uncorrectable error interrupt. In addition one sets the auto correct bit in the RS configuration register. This setting is important because the FLASH ECC module state machines have several stall states, notably at each phase boundary. If the autocorrect bit is not set and errors are detected then the decoder will stall at the end of the SC phase, set the ErrInt bit and wait for it to be cleared. A similar stall occurs at the end of the KES phase. Finally, within the EVAL phase, it will stall as each error location is computed so that the location can be verified. For normal operation, these stalls are bypassed by setting the autocorrect bit.

Finally, one should note that the data buffer pointed to by the block start address/index registers and the parity buffer pointed to by the parity start address/index registers will be read for the purposes of detecting and locating errors and will be written to correct up to four symbols in error. Note that correctable errors in the parity storage will be modified as needed.

The kick operation and done interrupt ISR invocation work as in the encoder case. Note that for the decoder, the ISR must monitor the uncorrectable error status bit to determine that a good data buffer resulted from the decode operation. If uncorrectable errors occurred, it is up to software to determine how to deal with a bad block. One strategy might be to re-read the data from NAND Flash in the hope that enough soft errors will have been removed to make correction successful on a second pass.

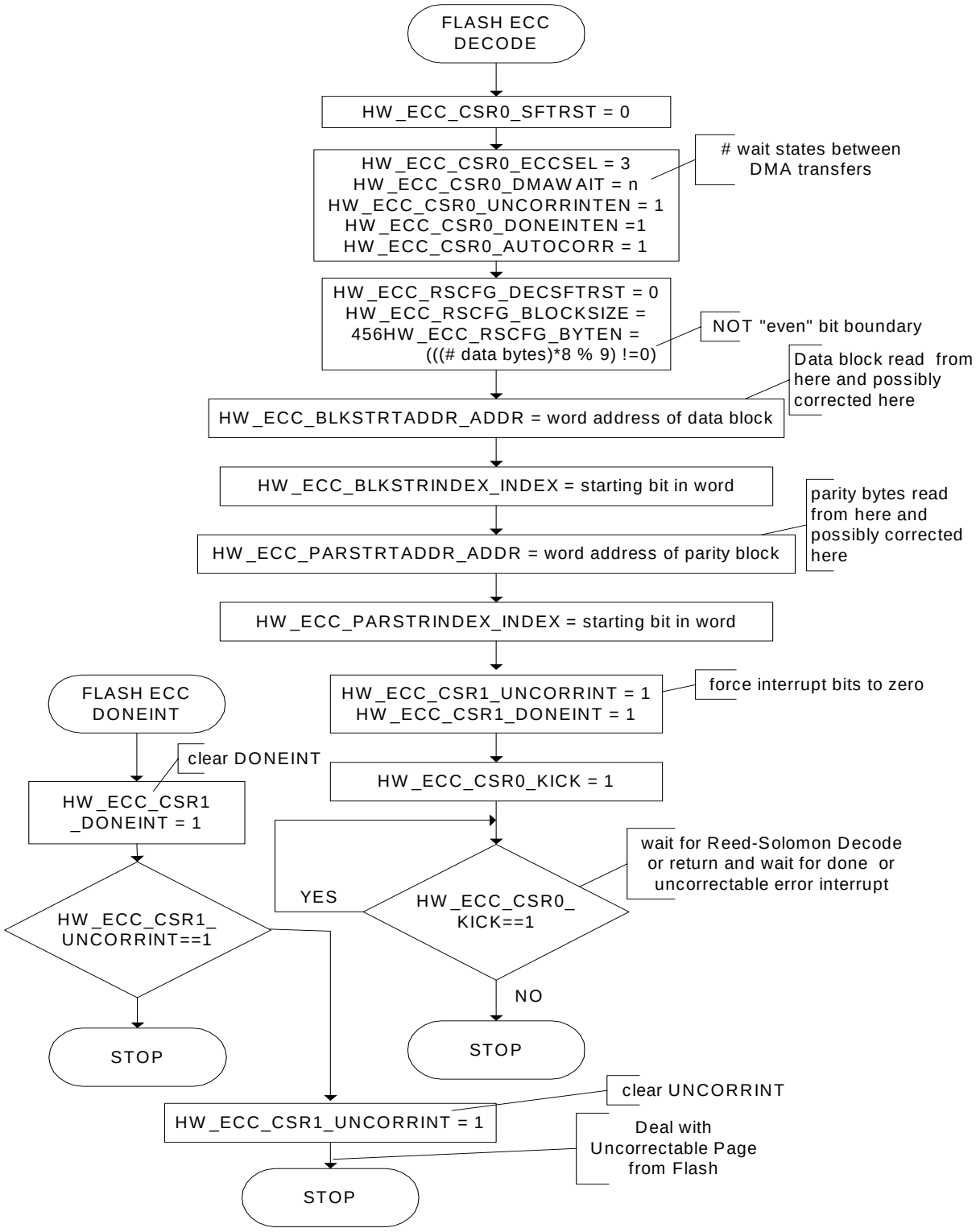


Figure 53. FLASH ECC Reed-Solomon Decode Flowchart

12.2. SSFDC Hamming ECC Accelerator

Consider the case where a 1Gbit large block NAND Flash device is attached to the STMP35xx. These devices have pages with 2048 bytes of data and 64 bytes of spare area. The Hamming code accelerator is designed to work on exactly 256 byte blocks. Therefore, one must treat the NAND Flash page as eight 256 byte blocks, computing eight separate three byte parity codes to cover the entire NAND Flash page as shown in Figure 54. This figure shows the computation of one 256 byte block parity out of the eight blocks needed.

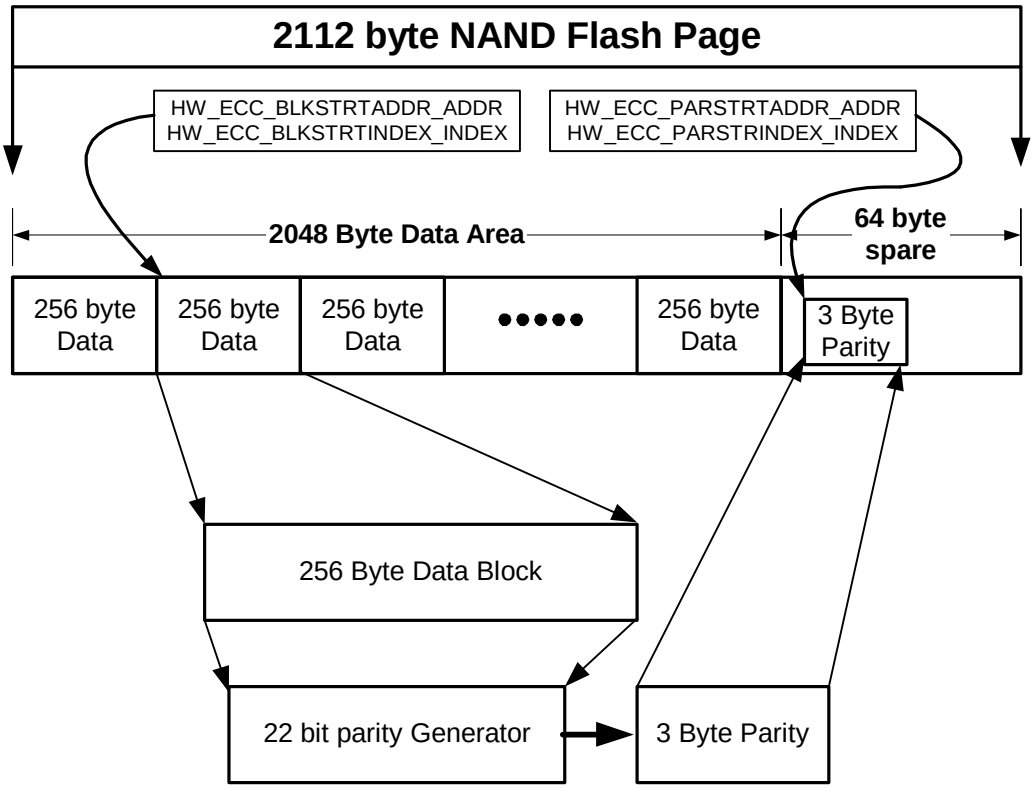


Figure 54. FLASH ECC SSFDC Encode Operation

As in the Reed-Solomon case, the **HW_ECC_BLKSTRTADDR** and **HW_ECC_BLKSTRTINDEX** registers are set to point to the beginning of the desired data block in on-chip RAM. The parity start address and index registers are set to point to the three byte buffer in on-chip RAM that will receive the generated parity bytes. When started, the DMA will fetch all 256 bytes from on-chip RAM and accumulate the appropriate 22 bit Hamming code for this fixed block size. The DMA will then write these bytes to the on-chip RAM.

Assume that the GPFLASH module will be used to write the results to the NAND Flash device in an overlapped fashion. While several options exists, the best compromise between buffer size and organization is to combine two 256 byte blocks as a unit and generate 6 bytes of parity. These can then be written to NAND Flash as a 512 byte data block followed immediately by a 16 byte spare area into which 6 bytes of parity are copied. With this scheme good overlap is established between the GPFLASH module and the ECC module while using only 2 X 518 byte buffers.

The encoder flow chart for the SSFDC looks similar to and behaves like the Reed-Solomon encoder. The differences are that the ECCSEL is set to zero for SSFDC encode and the SSFDC soft reset bit is turned off in the **HW_ECC_SSFDCCFG** register.

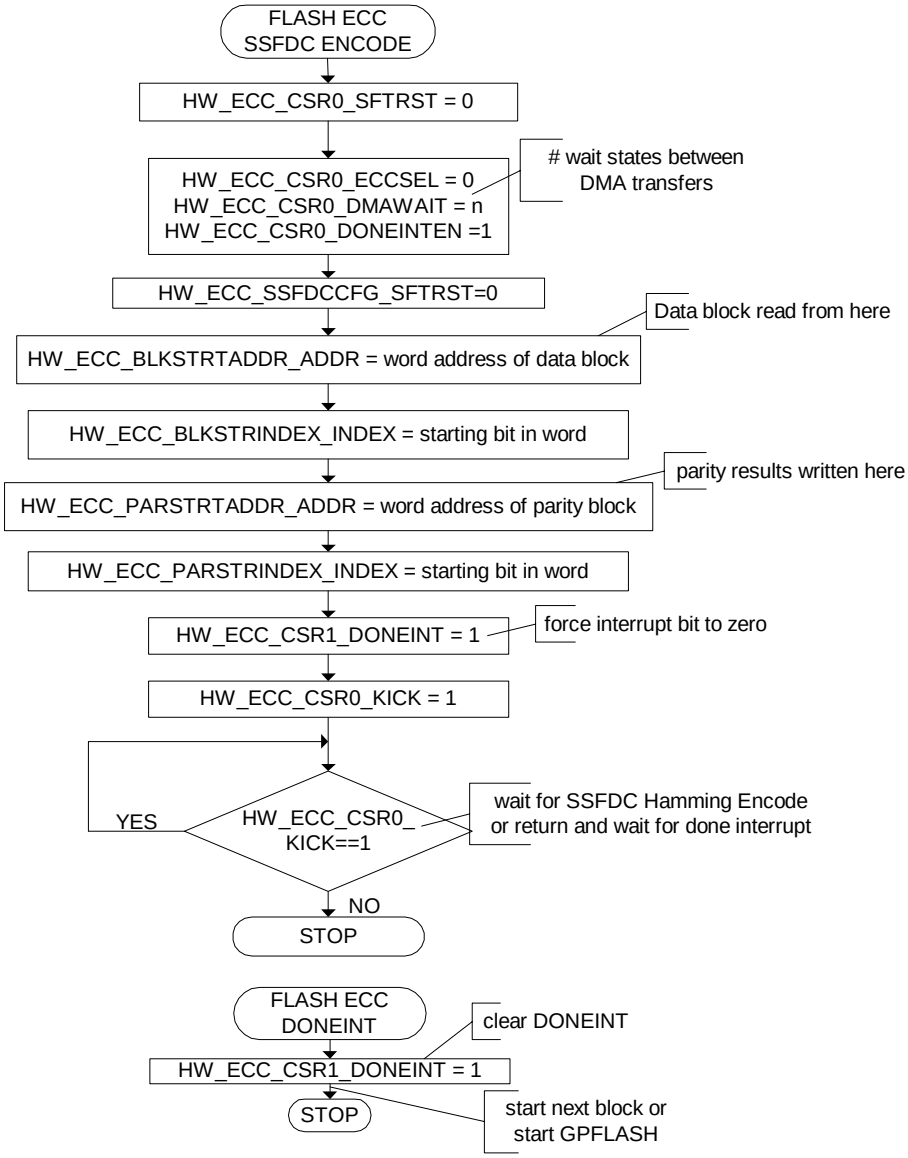


Figure 55. FLASH ECC SSFDC Encode Flowchart

The ECC module knows that it will write three bytes of parity instead of nine. The other difference is that software will set up the block start address and parity start address for 256 byte buffers instead of 512 byte as in the Reed-Solomon case. The ECC module takes 266 dclks to encode a 256 byte block.

When a page is read from NAND Flash, its SSFDC hamming code parity must also be read from the NAND Flash device. Each 256 byte block must be checked for errors and if a correctable error is found it must be corrected, either in the 256 byte

data portion or the 3 byte parity portion. This decoding process can also be fully overlapped with the NAND Flash transfers, as shown in Figure 56.

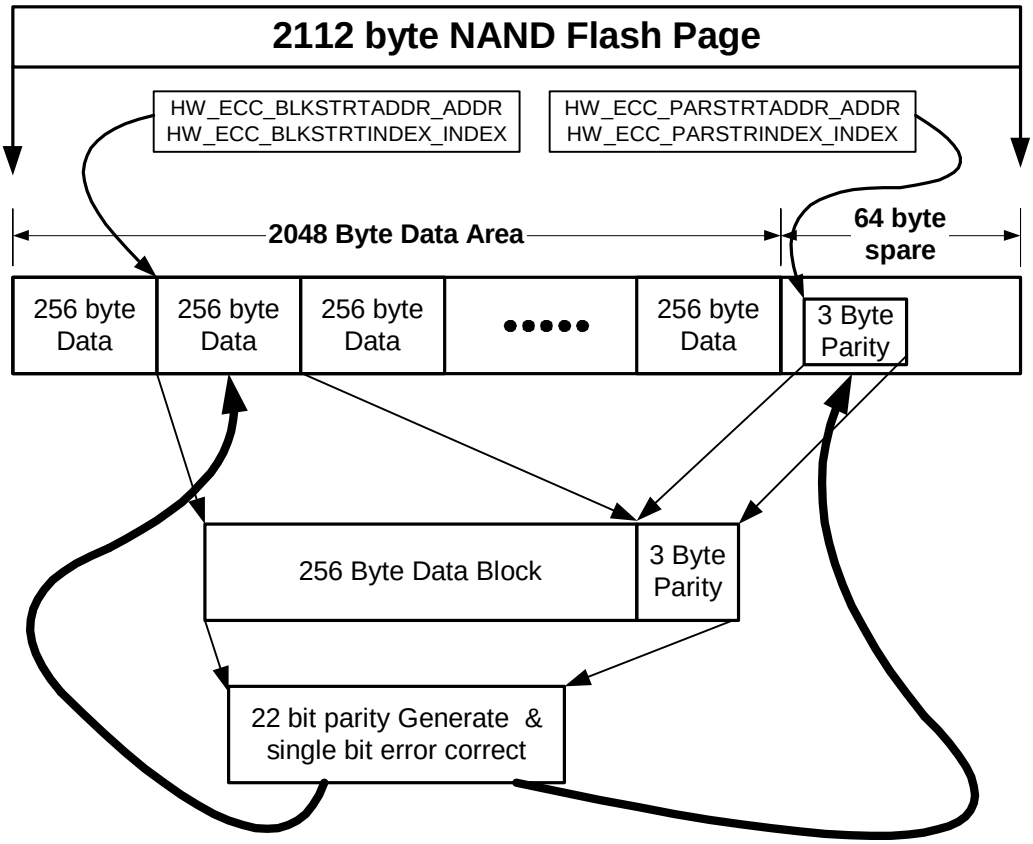


Figure 56. FLASH ECC SSFDC Decode Operation

Notice that in this case, the block start and the parity start address/index registers are set to point into the on-chip RAM buffers which received the bytes from the NAND Flash device. The decoder reads the data bytes and computes a new 22 bit parity value. It then reads the stored parity bytes and computes an error syndrome value by XORing the two parity values.

If all 22 bits of this error syndrome are zero then no errors are present. Otherwise the bits are examined to determine whether a single bit is in error. If not then an uncorrectable error interrupt is generated and further processing is terminated. For a single bit error, if the auto correct bit is set then the bit is corrected and a done interrupt is generated. The decoder takes approximately 266 DCLKs to complete, with or without error correction, provided no DMA WAIT cycles were added.

The SSFDC decoder flow chart of Figure 57 is very similar to the Reed-Solomon decoder flow chart. The major difference is setting the ECC SEL field to one for SSFDC decode. In addition, the SSFDC configuration register soft reset bit is reset.

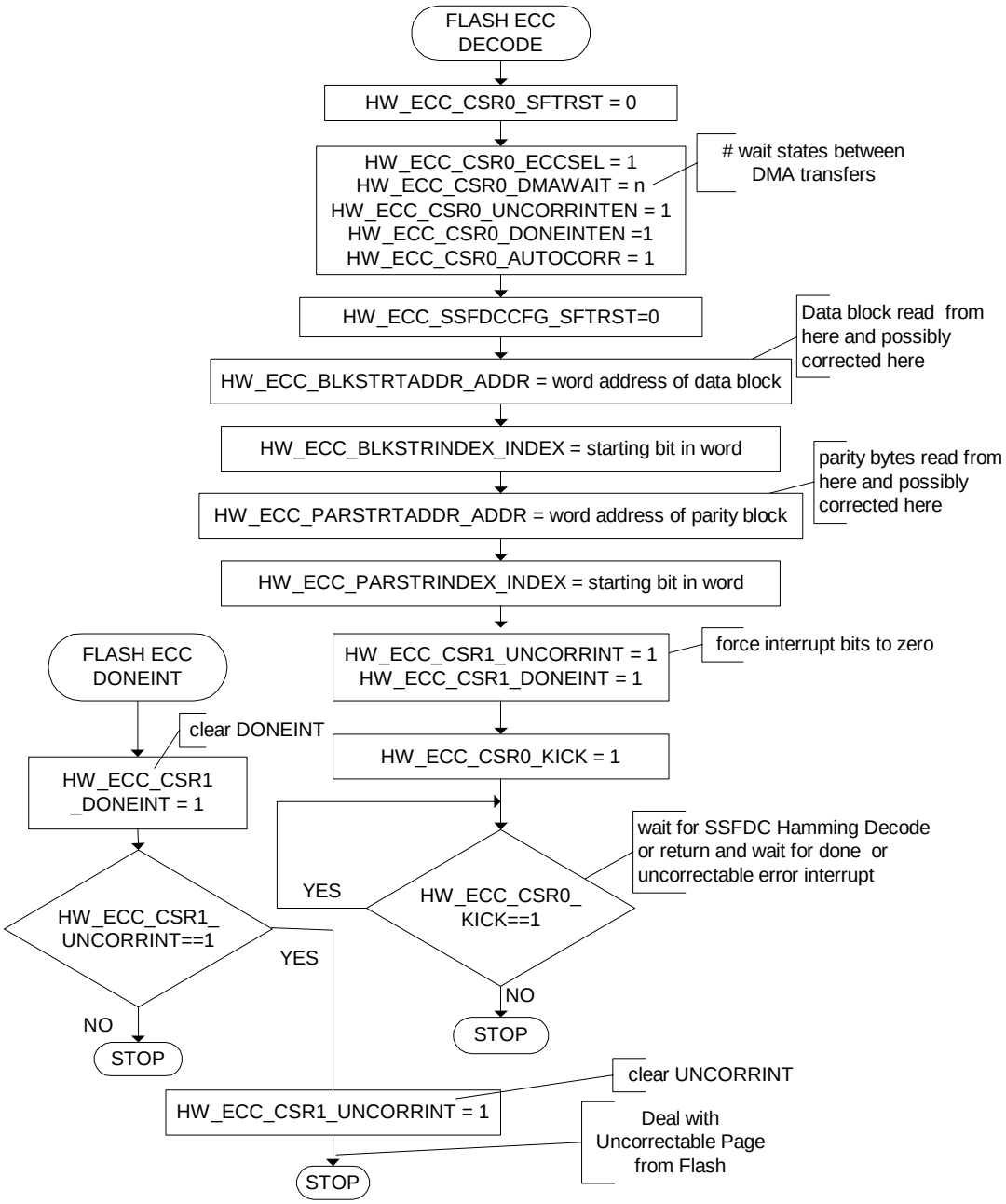


Figure 57. FLASH ECC SSFDC Decode Flowchart

12.3. FLASH ECC Programmable Registers

The following registers are available for DSP programmer access and control of the FLASH ECC accelerator.

12.3.1. Flash ECC Control/Status Register 0

This register provides overall control of the FLASH ECC accelerator for transaction and interrupts.

HW_ECC_CSR0 X:$F780

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SFTRST						ECCSEL		DMAWAIT							ERRINTEN	UNCORRINTEN	CORRINTEN	KESINTEN	DONEINTEN			AUTOCORR	ECCKICK

Table 143. HW_ECC_CSR0

BITS	LABEL	RW	RESET	DEFINITION
23	**SFTRST**	RW	1	Set to one for software reset and low-power mode. Set to zero for normal operation.
22:18	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
17:16	**ECCSEL**	RW	00	Select and enable desired ECC mode; 00 = HW_ECC_SSFDC_ENCODE 01 = HW_ECC_SSFDC_DECODE 10 = HW_ECC_RS_ENCODE 11 = HW_ECC_RS_DECODE
15:12	**DMAWAIT**	RW	0000	DMA Wait States - Specifies the number of cycles the ECC DMA will wait between consecutive memory cycle requests.
11:9	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
8	**ERRINTEN**	RW	0	Error Interrupt Enable - Set to one to enable the ECC to generate an interrupt when errors are detected in the current codeword.
7	**UNCORRINTEN**	RW	0	Set the uncorrectable interrupt enable bit to one to enable the ECC to generate an interrupt when an uncorrectable error situation is found and that further processing is not possible.
6	**CORRINTEN**	RW	0	Set the correctable interrupt enable bit to one to enable the generation of interrupts when a set of correctable errors is found. The interrupt (and stall if **AUTOCORR** is not set) is generated before initiating the corrections.
5	**KESINTEN**	RW	0	Set the key equation interrupt enable bit to one to enable the generation of an interrupt once KES processing is complete.
4	**DONEINTEN**	RW	0	Set the done interrupt enable bit to one to enable the generation of interrupts when processing is complete.
3:2	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
1	**AUTOCORR**	RW	0	Allow the ECC to automatically correct any correctable errors.
0	**ECCKICK**	RW	0	Set to one to start processing the currently configured block. This bit will automatically reset to zero when processing is complete.

Table 144. FLASH ECC Control/Status Register 0 Description

12.3.2. Flash ECC RS Configuration Register

This register provides configuration control for the various phases of the Reed-Solomon encoder/decoder.

HW_ECC_RSCFG X:$F782

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ENCSFTRST	DECSFTRST	RSCFGERR	BYTEEN								BLOCKSIZE									SYMBOLSIZE			

Table 145. HW_ECC_RSCFG

BITS	LABEL	RW	RESET	DEFINITION
23	**ENCSFTRST**	RW	1	Software reset and low-power mode for the RS-Encoder. Must be set to zero for normal operation of the encoder.
22	**DECSFTRST**	RW	1	Software reset and low-power mode for the RS-Decoder. Must be set to zero for normal operation of the decoder.
21	**RSCFGERR**	R	0	When the RS configuration error bit reads back a value of one, it indicates that an invalid combination of block size and symbol size has been written to the **BLOCKSIZE** field.
20	**BYTEEN**	RW	1	The RS encode/decode engine works on 9-bit symbols. The DMA behavior must be different if the last symbol perfectly aligns with the last word boundary. Let N equal the number of data bytes to transfer, if ((N*8)%9) is not zero then set **BYTEEN** to one.
19:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:4	**BLOCKSIZE**	RW	$1C8	The RS algorithm always implements a block code of 511 RS symbols, however, the programmer can choose to compute a 9 byte parity value for a source block smaller than 512 bytes. If 256 bytes are chosen, for example, then set this field to 228. The hardware will simply increase the number of zero pad bytes as appropriate.
3:0	**SYMBOLSIZE**	R	$9	This read only value indicates the symbol size for which the hardware was designed.

Table 146. FLASH RS Configuration Register Description

12.3.3. Flash ECC Control/Status Register 1

This register provides a continuation of control bits for the FLASH ECC module.

HW_ECC_CSR1 X:$F781

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
					ECCERROR			ECCEXCEPTION							ERRINT	UNCORRINT	CORRINT	KESINT	DONEINT				

Table 147. HW_ECC_CSR1

BITS	LABEL	RW	RESET	DEFINITION
23:19	**RSRVD**	R	0	**Reserved –** Must be written with 0.
18:16	**ECCERRORS**	R	$0	Number of errors detected/corrected. See **ECCEXCEPTION** field for details on uncorrectable errors.
15:12	**ECCEXCEPTION**	R	0000	Reason for termination of algorithm or uncorrectable error: 0: no exceptions 1: RS --> degree of “lambda” exceeds 4 2: RS --> “lambda” == all zeroes 4: RS --> degree of “lambda”!= number of roots of “lambda”, i.e. duplicate roots indicated. 8: SSFDC --> More than 1 Error.
11:9	**RSRVD**	R	0	**Reserved –** Must be written with 0.
8	**ERRINT**	RW	0	When set to one, the error interrupt status bit indicates that the current block contains errors and further processing is needed to determine correctability. This indicates that the syndrome computation is complete and the result is non-zero, i.e. indicating errors are present.
7	**UNCORRINT**	RW	0	When set to one, the uncorrectable error interrupt status bit indicates that uncorrectable errors have been detected in the most recently processed block. If this interrupt is enabled, processing will stop until this bit is cleared. Write a one to this bit to clear it.
6	**CORRINT**	RW	0	When set to one, the Correctable errors interrupt status bit indicates that a correctable error has been detected. Processing will stop when this bit is set until it is cleared. It will become valid each time the Chien Search module finds an error location. This can happen no more than 4 times per block. Write a one to this bit to clear it.
5	**KESINT**	RW	0	When set to one, the Key Equation Solver interrupt status bit indicates that the KES module has completed processing and a solution is available.
4	**DONEINT**	RW	0	When set to one, the done interrupt status bit indicates that the current block has been processed and all errors are identified and corrected.
3:0	**RSRVD**	R	0	**Reserved –** Must be written with 0.

Table 148. FLASH ECC Control/Status Register 1 Description

12.3.4. Flash ECC SSFDC ECC Configuration Register

This register provides configuration control for the SSFDC encoder and decoder.

HW_ECC_SSFDCCFG X:$F783

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SFTRST											BLOCKSIZE									SYM;BOLSIZE			

Table 149. HW_ECC_SSFDCCFG

BITS	LABEL	RW	RESET	DEFINITION
23	**SFTRST**	RW	1	Software reset and low-power mode for SSFDC. Common reset for both encode and decode functions.
22:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:4	**BLOCKSIZE**	R	$100	Only block size 256 ($100) is supported.
3:0	**SYMBOLSIZE**	R	$8	Only symbols size 8 is supported.

Table 150. FLASH ECC SSFDC Configuration Register Description

12.3.5. Flash ECC Block Start Address Register

This register provides a pointer into on-chip RAM to the first byte of the data block.

HW_ECC_BLKSTRTADDR X:$F784

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ADDR															

Table 151. HW_ECC_BLKSTRTADDR

•

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**ADDR**	RW	$0000	Start address for the data block.

Table 152. FLASH Block Start Address Register Description

12.3.6. Flash ECC Block Start Index Register

This register provides the starting bit offset into the first 24 bit word pointed to by **HW_ECC_BLKSTRTADDR**. This register also specifies the memory space that contains the data block.

HW_ECC_BLKSTRTINDEX X:$F785

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														MEMSPACE					INDEX				

Table 153. HW_ECC_BLKSTRTINDEX

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:8	**MEMSPACE**	RW	00	Memory space in which the data block is stored. 00 = X-Memory 01 = Y-Memory 10 = P-Memory
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**INDEX**	RW	00000	Bit index into the 24-bit memory word where the 1st symbol occurs in the data block.

Table 154. FLASH ECC Block Start Index Register Description

12.3.7. Flash ECC Parity Start Address Register

This register provides a pointer into on-chip RAM for the buffer containing the parity bytes.

HW_ECC_PARSTRTADDR X:$F786

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ADDR															

Table 155. HW_ECC_PARSTRTADDR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**ADDR**	RW	$000	Start address for the parity storage buffer.

Table 156. FLASH ECC Parity Start Address Register Description

12.3.8. Flash ECC Parity Start Index Register

This register provides the starting bit offset into the first 24 bit word pointed to by **HW_ECC_PARSTRTADDR**. This register also specifies the memory space that contains the parity storage block.

HW_ECC_PARSTRTINDEX X:$F787

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														MEMSPACE					INDEX				

Table 157. HW_ECC_PARSTRTINDEX

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:8	**MEMSPACE**	RW	00	Memory space in which the parity is stored. 00 = X-Memory 01 = Y-Memory 10 = P-Memory
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**INDEX**	RW	00000	Bit index into the 24-bit memory word where the 1st symbol occurs in the data block.

Table 158. FLASH ECC Parity Start Index Register Description

12.3.9. Flash ECC Error Location Address Register

This register provides a read only view of the location of an error.

HW_ECC_LOCADDR X:$F788

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ADDR															

Table 159. HW_ECC_LOCADDR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**ADDR**	R	$0000	Word address of the error. This field is only valid while the Corrint status bit is set.

Table 160. FLASH ECC Error Location Address Register Description

12.3.10. Flash ECC Error Location Index Register

This register provides a read only view of the memory space indicator and bit index for the error that is currently being reported.

HW_ECC_LOCINDEX X:$F789

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														MEMSPACE					INDEX				

Table 161. HW_ECC_LOCINDEX

BITS	LABEL	RW	RESET	DEFINITION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:8	**MEMSPACE**	R	00	Memory space for the error. This field is only valid while the Corrint status bit is set. 00 = X-Memory 01 = Y-Memory 10 = P-Memory
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**INDEX**	R	0000	Contains the bit index into the 24-bit memory word where the 1st bit of the erroneous symbol occurs. This can either be a 9-bit RS symbol or a 1-bit SSFDC symbol. This field is only valid while the Corrint status bit is set.

Table 162. FLASH ECC Error Location Index Register Description

12.3.11. Flash ECC Error Value Register

This register provides a read only view of the ECC error value to be used to correct a symbol in error.

HW_ECC_ERRVAL X:$F78A

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			BLOCKLOC												ERRORVALUE								

Table 163. HW_ECC_ERRVAL

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	0	**Reserved –** Must be written with 0.
20:12	**BLOCKLOC**	R	$000	Block location of an Error Value. Provides the location within the data block or parity block where the error is located. To be used in of Search/Replace function of Chien search module. This field is only valid when the Corrint field is set.
11:9	**RSRVD**	R	0	**Reserved –** Must be written with 0.
8:0	**ERRORVALUE**	R	$000	The Error Value is the 9-bit value to be XORed with the existing data at the Error Location. For SSFDC data, only the LSB should be used to correct the single-bit error. This field is only valid when the Corrint field is set.

Table 164. FLASH ECC Error Value Register Description

13. FILTER COPROCESSOR (FILCO)

Previous generations of the D-Major™ family used a significant amount of processor time to perform basic FIR filtering. The DAC output flow requires a 2:1 interpolating FIR filter on the stereo pairs to upsample the audio, see Section 24. “DAC” on page 269. The ADC flow requires an 8:1 decimating FIR filter on the stereo pairs to downsample the audio stream and to derive the appropriate quantization, see Section 25. “ADC” on page 279. In addition, a multi-band equalizer utilizes a number of bi-quadratic IIR filters to adjust the various bands.

All of these activities made extensive use of the DSP’s multiply accumulate features. The STMP35xx filter coprocessor (FILCO) is used to off load significant amount of very standard signal processing operations from the DSP. The reduction in DSP overhead yields more available MIPS for additional software. This reduction in DSP workload can also be used to lower the DCLK and in some cases reduce operating voltages to greatly extend battery life.

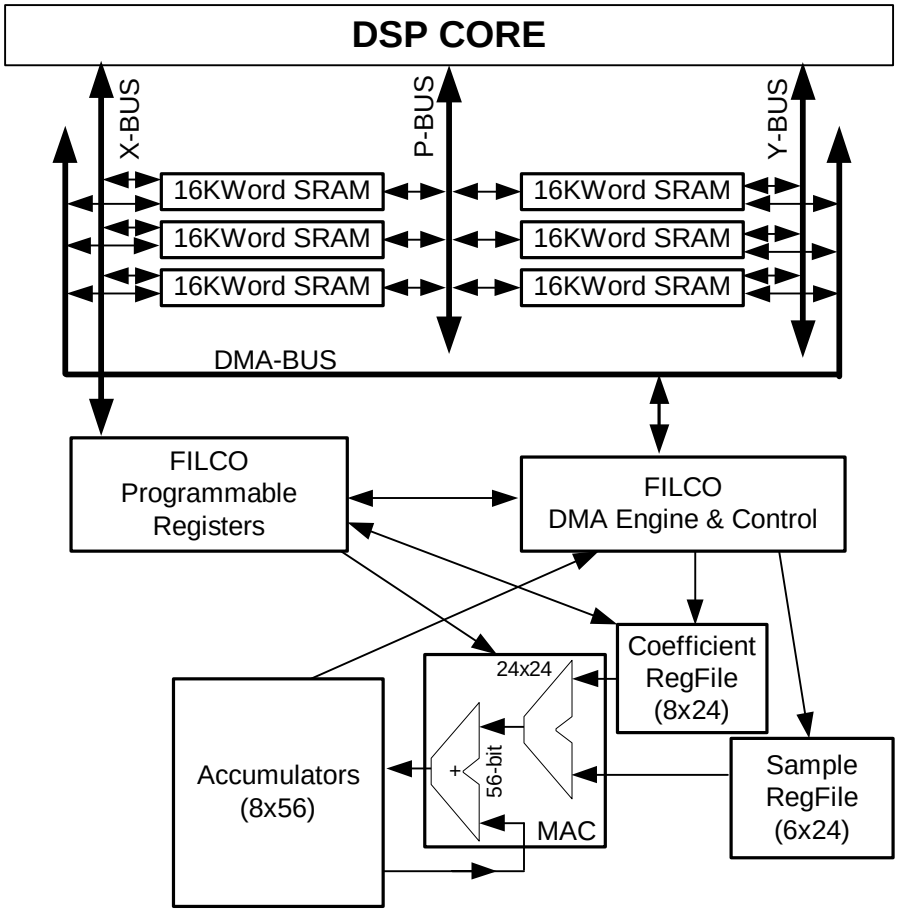


Figure 58. Filter Coprocessor (FILCO)

Figure 58 shows the high level block diagram for the FILCO. The FILCO is essentially a MAC engine that is directly fed from a DMA engine. To perform a simple 8-tap FIR, for example, the DMA is pointed at a 44.1KHz stereo sample buffer to be FIR low pass filtered. These samples are fetched via DMA cycles into the 24 bit wide Sample Register File. The eight coefficients are fetched into the coefficient reg-

ister file, as needed. The accumulator register file holds the intermediate summations for the multiply accumulates that are occurring for both the left channel output samples and the right channel output samples, as shown in Figure 59, below.

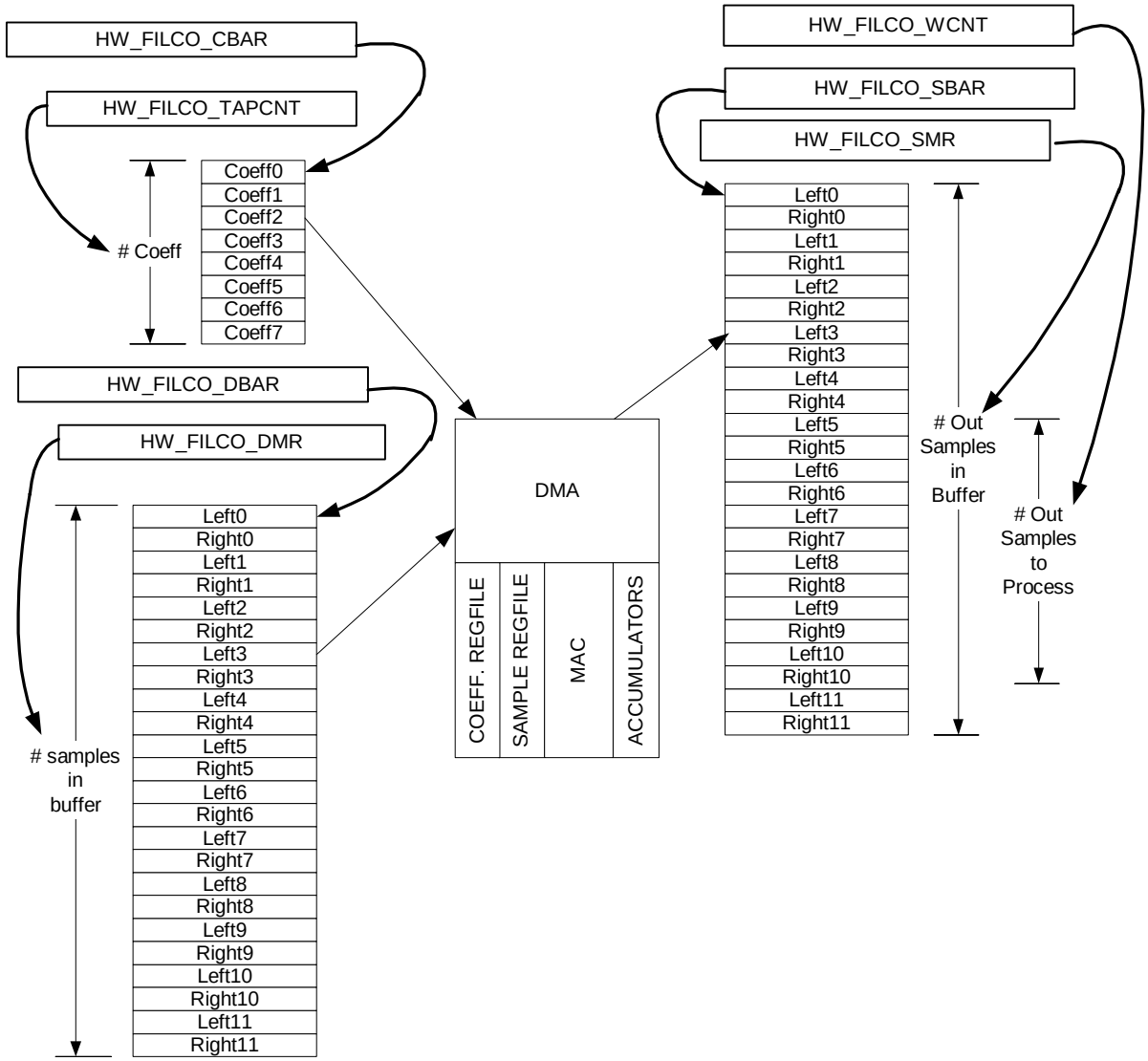


Figure 59. FILCO FIR EXAMPLE

In setting up this simple example, we set **HW_FILCO_DBAR** to point to the stereo input buffer and set the **HW_FILCO_DMR** modulo register to the number of words (not word pairs) in the input buffer. We set the **HW_FILCO_CBAR** base address register to point the buffer containing the eight coefficients to be used for this filter and we set **HW_FILCO_TAPCNT** tap count register to a value of eight. Next we set the **HW_FILCO_SBAR** output sample base address register to point to the circular buffer to receive the filtered output. We set the **HW_FILCO_SMR** sample modulo register to the number of words in the output buffer. Next we set the **HW_FILCO_WCNT** word count register to the number of output sample words to produce for the next DMA KICK. Notice that we can schedule much fewer words than the entire output circular buffer for each DMA KICK, (technically we could also

schedule more). Once these basics are completed along with other more detailed control register settings we can KICK the DMA and cause the FIR output samples to be computed without further DSP involvement. As the FILCO processes this operation, it reads a set of 8 left channel inputs and multiply accumulates them with their corresponding coefficients to produce the first left output sample. It then reads eight right channel samples and performs the coefficient MACs to produce the first right output sample. This process goes on until all the desired output samples have been generated at which time the KICK bit is reset and an interrupt is generated to the DSP. See the discussion below which describes the FILCOS use of all six sample registers, all eight coefficient registers and all eight accumulator registers to make this process much more parallel and efficient. In essence, the FILCO can sustain a rate of one MAC per DCLK indefinitely without saturating the DMA bus (it uses approximately 66% of the bus bandwidth at full speed). DSP applications rarely exceed 25% MAC utilizations over long time periods. Thus a significantly more powerful signal processing capability is available for a few specific algorithms, namely FIR and bi-quadratic IIR stereo filters.

13.1. FIR FILTER MODE

Typical DSPs are Harvard architectures with multiple memory banks, e.g. the 56K DSP has a P memory for instructions and X and Y memory for simultaneous access to coefficients and samples. For normal multiply accumulate instructions, the DSP simultaneously reads the next sample and coefficient values from X and Y memories. It then performs a 24 bit by 24 bit multiply and adds the result to a 56 bit accumulator. This allows the 56K DSP to execute a MAC every clock. For the specific algorithms supported by the FILCO, very simple but effective data reuse strategies exist. Consider the basic equation of an FIR filter as it would be executed for one FILCO KICK as shown here.

$$\forall j, (j \in \{0 \ldots (S-1)\}), \quad \text{Accumulator}_j = \sum_{i=0}^{T-1} a_{i+j} \times c_i \text{ (for left channel)}$$

$$\forall j, (j \in \{0 \ldots (S-1)\}), \quad \text{Accumulator}_j = \sum_{i=0}^{T-1} b_{i+j} \times c_i \text{ (for right channel)}$$

In these equations, T is the number of taps and S is the number of output samples to be computed for a single FILCO KICK. If one examines the partial computation in the middle of processing a pair of FIRs, one for each stereo channel, one sees the computations depicted in Figure 60. Thus the opportunity exists to compute four separate output sample results for the left channel and another four separate output sample results for the right channel. Each of these partial results is accumulated in one of the eight accumulators in the accumulator register file. From Figure 60, we see that each left input sample is fetched once and used four times, each right input sample is fetched once and used four times and each coefficient is used 8 times for each fetch, for a sustained utilization of $^1/_4 + ^1/_4 + ^1/_8$ or 63%. It takes 24 multiplies to compute the partial filter of figure 60 during this time 12 coefficient and samples are fetched so that the utilization ends up being 12/24 or approximately 50% of the DMA bus. This is a steady state number for long filters; for startup and rundown, the FILCO will use closer to 100% of the DMA bus utilization for short periods. The **HW_FILCO_CSR_DMADLY** value can be used to *throttle* the DMA utilization of the

FILCO. This field indicates the number of additional DCLKs to wait before asserting a DMA request to the on-chip RAM.

Because the FILCO is running eight separate FIR filter computations at once, there is a four deep pipeline running for each channel. As a result, three words of zero must be appended to the end of the coefficient array for FIR filters (plus three more for FIR decimation mode). These zero words pad out the run down of the four deep pipeline.,

$$\begin{aligned}
\text{Accumulator0} &+= a_n * c_m + a_{n+1} * c_{m+1} + a_{n+2} * c_{m+2} \\
\text{Accumulator1} &+= a_n * c_{m+1} + a_{n+1} * c_{m+2} + a_{n+2} * c_{m+3} \\
\text{Accumulator2} &+= a_n * c_{m+2} + a_{n+1} * c_{m+3} + a_{n+2} * c_{m+4} \\
\text{Accumulator3} &+= a_n * c_{m+3} + a_{n+1} * c_{m+4} + a_{n+2} * c_{m+5} \\
\\
\text{Accumulator4} &+= b_n * c_m + b_{n+1} * c_{m+1} + b_{n+2} * c_{m+2} \\
\text{Accumulator5} &+= b_n * c_{m+1} + b_{n+1} * c_{m+2} + b_{n+2} * c_{m+3} \\
\text{Accumulator6} &+= b_n * c_{m+2} + b_{n+1} * c_{m+3} + b_{n+2} * c_{m+4} \\
\text{Accumulator7} &+= b_n * c_{m+3} + b_{n+1} * c_{m+4} + b_{n+2} * c_{m+5}
\end{aligned}$$

Figure 60. FILCO FIR Mode DMA Fetch Re-use

The FILCO performs 24-bit by 24-bit, parallel, twos-complement fractional multiples. The “fixed point” for the fractional number occurs between the sign bit and the first data bit, i.e. between bit 23 and bit 22 with the sign bit located in bit 23. The result of the multiply is a 47-bit twos-complement fractional value to which a zero is appended on the right to make a 48-bit number before accumulation. The most negative number that can be represented in 24 bit fractional representation is -1.0 while the most positive is approximately +0.999998 which is ((2^23)-1)/(2^23), i.e. just less than +1.0.

Multiplying two numbers whose value is -1.0 should yield a value of +1.0 but this value is not representable in 24-bit twos-complement fractional numbers. An overflow occurs in this case and the result is -1.0. This overflow is neither detected nor corrected in the FILCO. Users should avoid FIR filters whose coefficient sets contain the number -1.0. If this coefficient is used in an FIR filter, then the user should insure that no corresponding sample with a value of -1.0 will be seen.

When the output sample is extracted from the accumulator, the DMA pulls bits 47 down through bit 24 to write to on-chip RAM as the 24 bit FIR result. For a fractional number representation, this has the effect of discarding precision without changing the magnitude. For example, a 48 bit twos-complement fractional number with a value of +0.5 becomes a 24-bit twos-complement fractional number with a value of +0.5. The accumulator value is saturated so that the 24 bit result has the most positive number substituted for any value that is larger or the most negative number is substituted for any value that is smaller. Otherwise convergent rounding is used on the accumulator value to produce the 24-bit result.

The **HW_FILCO_SAT_SAT** bit-field is an 8-bit field with a sticky bit for each of the eight accumulators. Anytime a saturation correction is applied to a specific accumulator value to store it to on-chip RAM during an FIR operation, then its correspond-

ing SAT bit is set. Once software has examined a saturation status bit, it can be cleared by writing a one to it.

The FILCO module includes a completion interrupt to notify software that a kicked off operation is complete. To facilitate context switching, a currently running filter can be “unkicked” so that it will quiesce to a safe state even before its word count reaches zero.

Consider a simple FIR with 10 taps whose data reference patterns can be seen in Figure 61, below. For this FIR, all ten coefficients are accessed from the coefficient register file, sequentially one after the other. For the “m^{th}” output sample, whose basis is shown starting at left sample Left0, all of the left samples from Left0 through Left 9, inclusive, are read and multiplied by their corresponding coefficient.

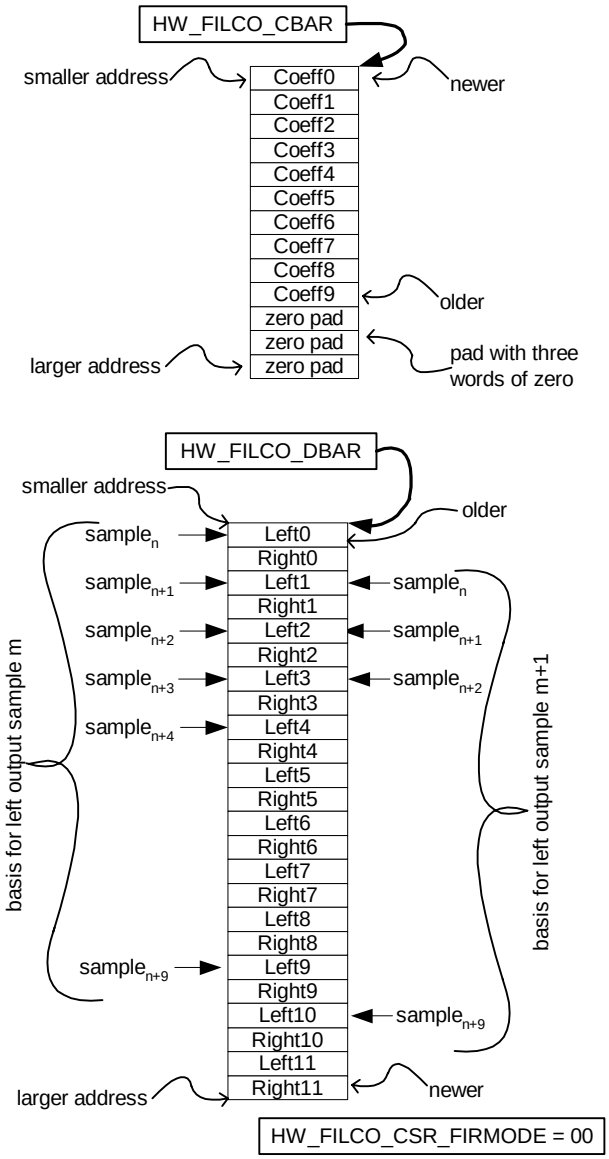


Figure 61. FILCO Normal FIR Mode

For output sample m+1, the basis begins at the next left sample after Left0 or Left1. So the coefficient vector is multiplied by a basis vector that slides along the sample stream, moving one sample down the input sample stream for each filter output sample. This sample reference pattern results when the HW_FILCO_CSR_FIRMODE bit field is set to a value of 00, for the 1:1 or normal FIR mode

13.1.1. FIR Decimation Filter

When the **HW_FILCO_CSR_FIRMODE** bit field is set to 10 then it enables the 2:1 decimation mode. In this mode, FIR filter calculations for each output sample are made, as in the normal case, except that the filter basis slides down the input sample buffer differently, see Figure 62. “FILCO FIR Decimation Mode” on page 156. In this mode, after sample m has been computed with a basis that began at sample Left0, the sample m+1 basis is fetched beginning at sample Left2. Notice that sample Left1 was “skipped” and never used as the first sample of any basis. Thus the 2:1 decimation mode produces one output sample for every two input samples.

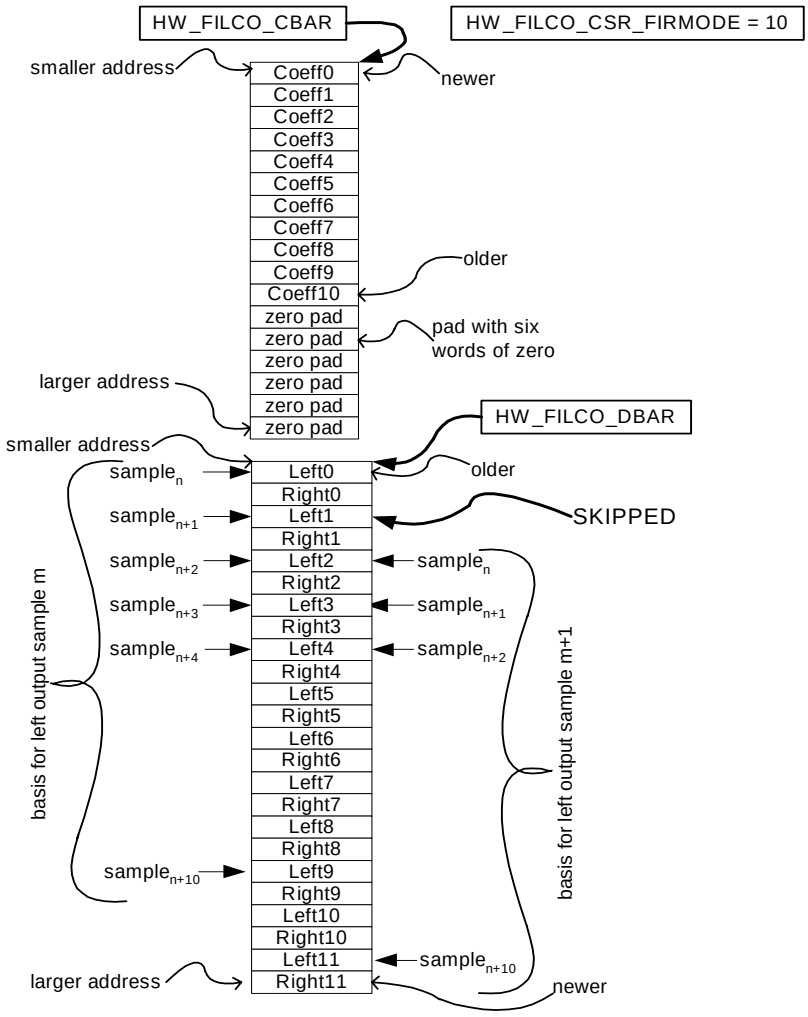


Figure 62. FILCO FIR Decimation Mode

13.1.2. FIR Interpolation Filter

When the **HW_FILCO_CSR_FIRMODE** bit field is set to 01 then it enables the 1:2 interpolation mode. In this mode, FIR filter calculations for each output sample are made, as in the normal case, except that the filter basis slides down the input sample buffer differently, see Figure 63. “FILCO FIR Interpolation Mode” on page 157.

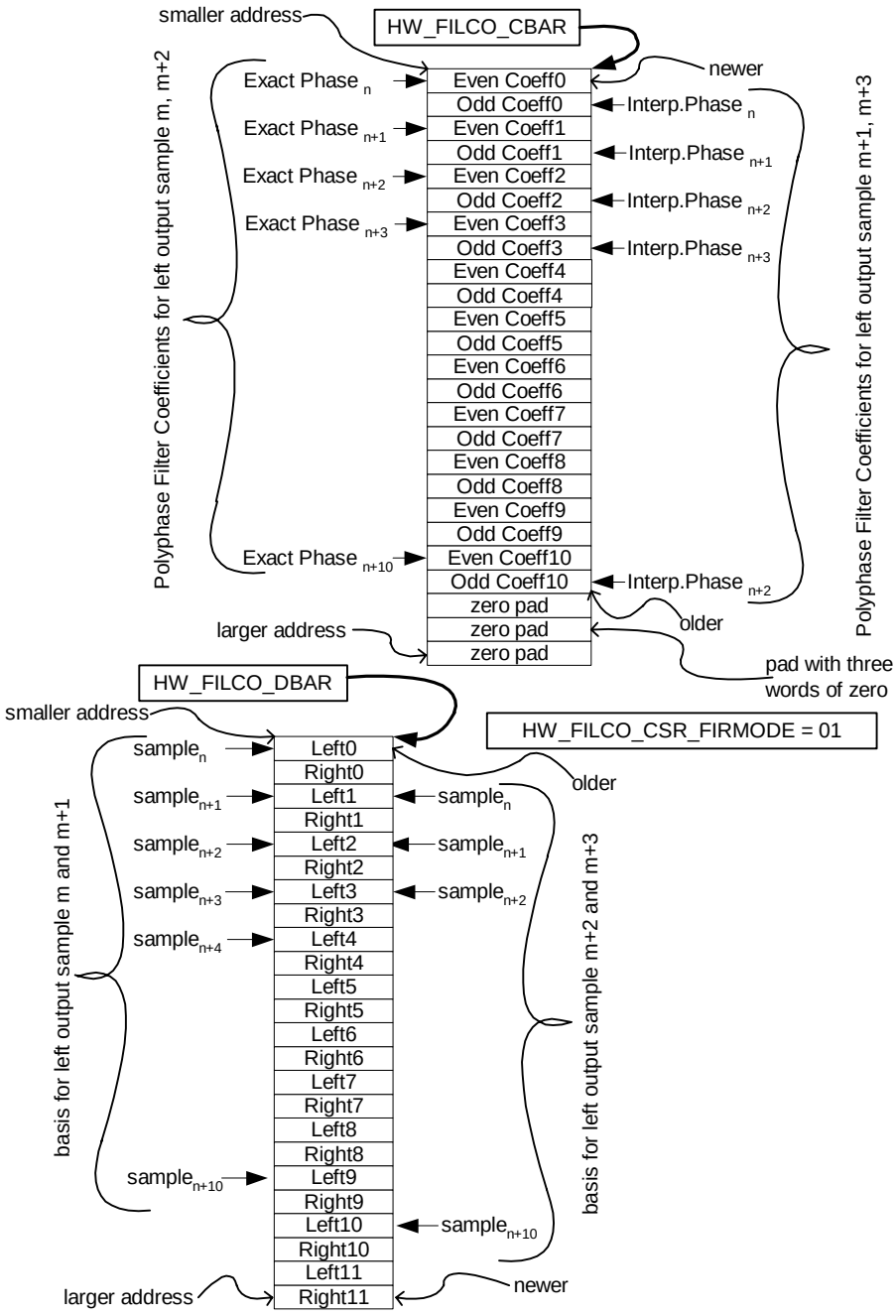


Figure 63. FILCO FIR Interpolation Mode

This setting enables a special polyphase filter mode in which the same basis is read twice and convolved with two separate sets of coefficients, an even set and an odd

set. For the case shown in Figure 63, output sample m is computed using the input sample basis starting at sample Left 0. EvenCoeff 0, 1, 2, etc. is used to compute output sample m. To compute output sample m+1, the same input sample basis starting at sample Left0 is read and convolved with the Odd coefficients, starting with OddCoeff0, 1, 2, etc. To compute output sample m+2, the basis is stepped so that it starts at input sample Left1 which is then convolved with the Even coefficients. Finally the same input sample basis, beginning with input sample Left1 is used to generate output sample m+3. Pay careful attention to the four output samples shown in Figure 63, noting which coefficient sets and which input sample basis are used for each one.

13.2. Bi-Quadratic IIR Filter Cascade Mode

FILCO supports Bi-Quadratic IIR filter modes for both additive and subtractive graphic equalizer implementations. The basic bi-quad filter is shown in Figure 64. The bi-quad mode is selected by setting **HW_FILCO_CSR_FILCOSEL** to one. For this example, we set **HW_FILCO_CSR_IIRMODE** to zero for cascade mode.

This filter uses exactly five user supplied coefficients, a_1, a_2, b_0, b_1 and b_2. These are pointed to by the **HW_FILCO_CBAR** coefficient base address register. Since exactly five coefficients are used every time, FILCO ignores any value that may be in the **HW_FILCO_TAPCNT** register at the time a bi-quad filter is *"KICKed"* off. The bi-quad inner loop scale factor is controlled by **HW_FILCO_CSR_IIRSCALE** and can be set to multiply by either 1.0 or 2.0, recall that neither of these values could have been represented in a 24-bit twos complement fractional number.

The bi-quad filter processes interleaved stereo sample streams so there are two bi-quad filters running at the same time with both filters using the same coefficient sets. Figure 64 and the following discussion focuses only on the left channel but right channel filtering is identical.

HW_FILCO_DBAR points to the input sample data buffer and **HW_FILCO_DMR** specifies its modulo size in words as it did for FIR filters. For every left sample taken from the input buffer, one sample is produced and written to the output buffer. The output buffer is pointed to by **HW_FILCO_SBAR** and its modulo size is given in **HW_FILCO_SMR** as it was for FIR filters. The **HW_FILCO_WCNT** register specifies the number of samples to compute for each kick, as id did for the FIR case.

Finally notice that each bi-quad output value has a final gain multiplier applied to it before it is written to the output sample buffer. Various zero crossing decisions and gain value substitutions can be automatically handled by FILCO, see Section 13.4. "IIR Mode Zero Crossing Detector and Gain Substitution" on page 162.

As with the FIR filter, 56 bit accumulator values are saturated and/or rounded before storing to the output buffer. In addition, feed back values derived from the accumulator are also saturated and rounded before being used in subsequent multiplications. Also as in the FIR case, multiplication of -1.0 by -1.0 is neither detected nor saturated to +1.0 and will supply -1.0 to the accumulation.

FILCO is used to processes sub-buffers of long continuous streams of samples. When the available samples have been processed then FILCO must have its context switched to a new filter segment. When switching to or from an IIR filter then the accumulated state of the filter must saved and restored. Figure 65 shows the seven registers that must be saved and restored at an IIR context switch. When an IIR filter is first started, these registered should be loaded with zeroes.

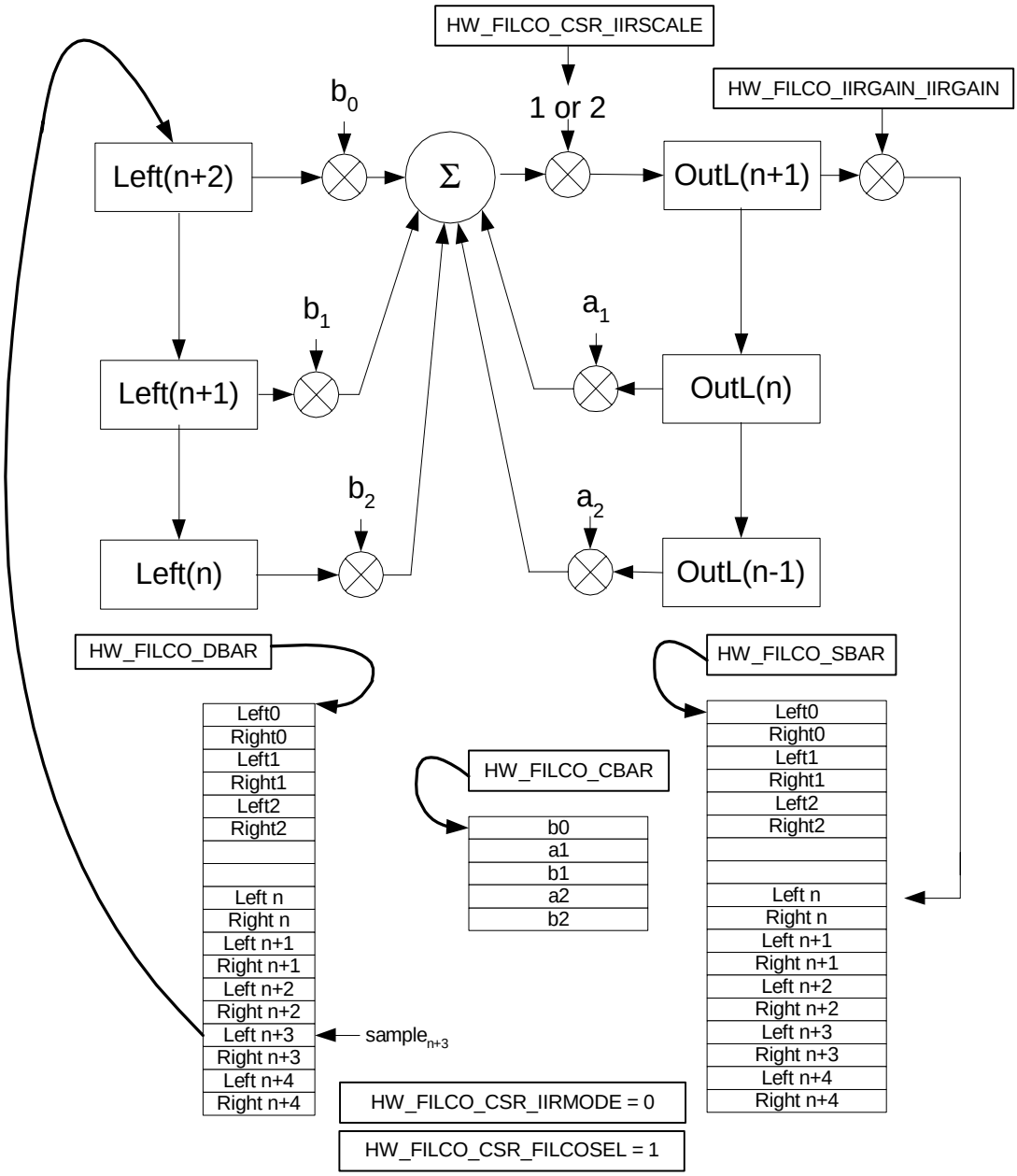


Figure 64. FILCO BIQUAD IIR Cascade Filter

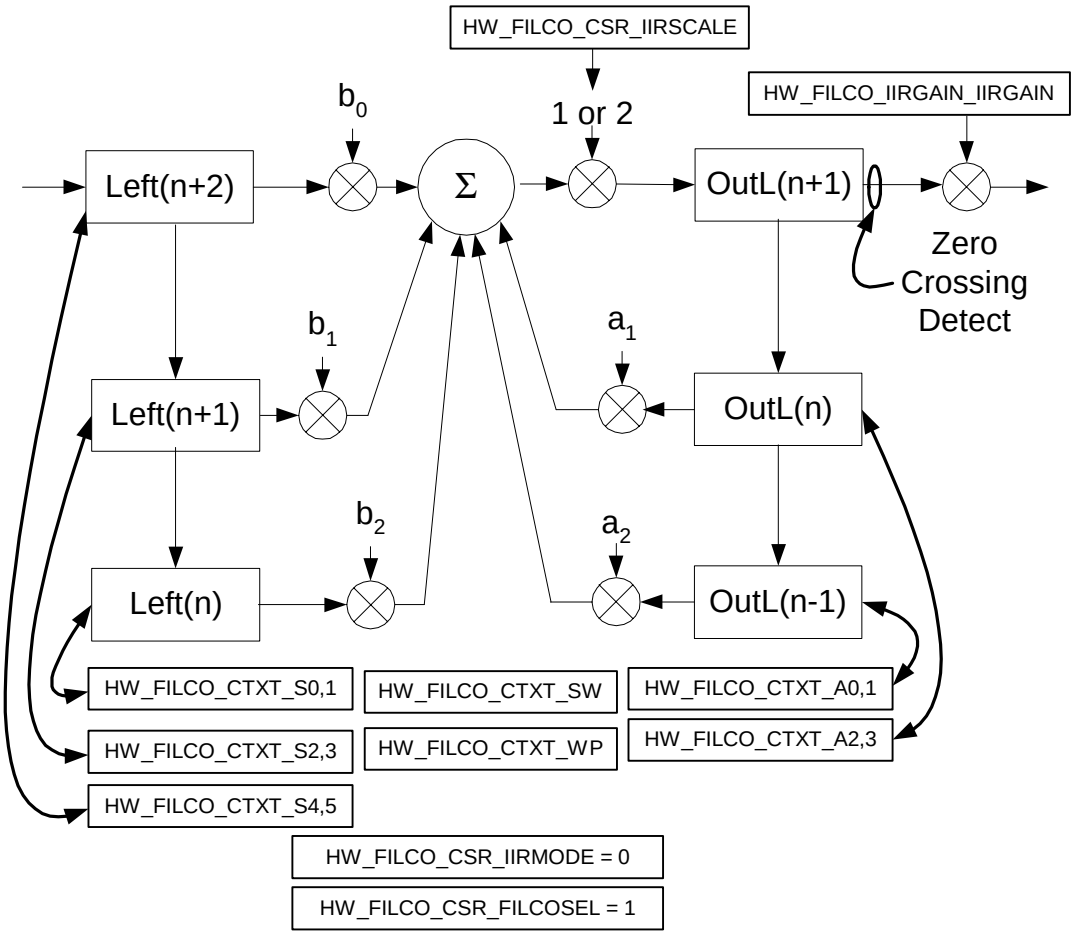


Figure 65. Context Switching an IIR Filter

Figure 66 shows the construction of a K band subtractive mode graphics equalizer. With a subtractive mode equalizer, a single buffer is used for multiple passes of each stage/band of the equalizer. Each time the bi-quad filter is run to process a band, both **HW_FILCO_DBAR** and **HW_FILCO_SBAR** are set to point the same circular buffer. **HW_FILCO_DMR** and **HW_FILCO_SMR** are both set to the size of the buffer. The filter coefficients are set for the first band and the filter is kicked off. The filter coefficients for this case are set for notch filtering, i.e. potentially reducing the amplitude of any frequency components within its passband. Out-of-band frequency components are essentially unmodified. Thus, frequency components within the band are effectively “subtracted” from the composite signal. This is counter intuitive to what is displayed on the user interface of a graphic equalizer but an effective implementation technique. For the final filter/channel bank run for the equalizer, one

can choose to point the results back in to the same buffer for in-place computation or one can use this filter operation to also copy the result to another circular buffer.

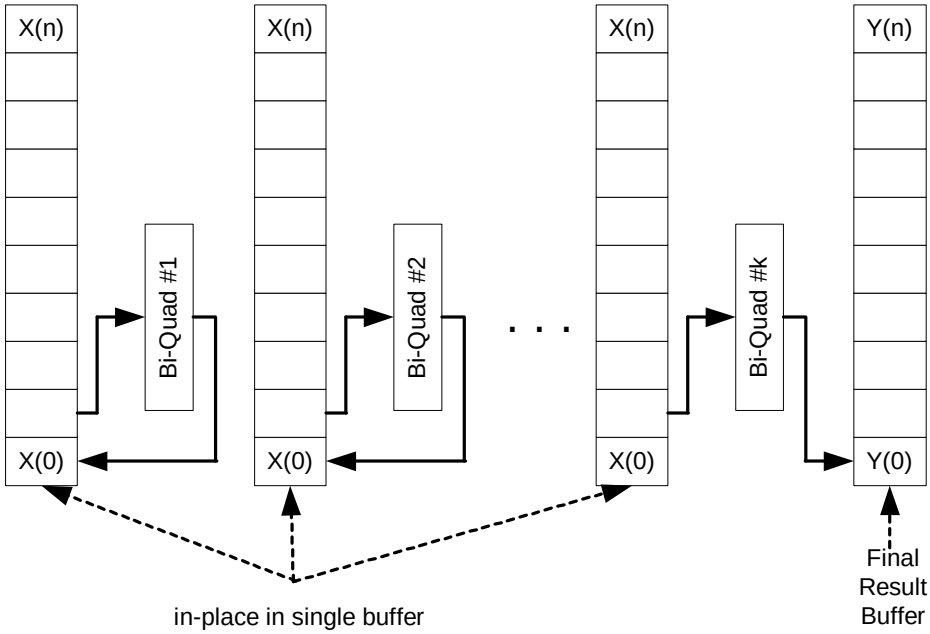


Figure 66. FILCO BiQuad Cascade EQ Application

13.3. Bi-Quadratic IIR Filter Parallel Mode

The K-band additive graphic equalizer model shown in Figure 67 uses a different and more intuitively pleasing strategy. In this case, each filter is set as a bandpass filter so that the frequency components outside of its passband are strongly suppressed while the components lying inside its passband are copied to the output buffer where the components from all banks are superpositioned with the summation operator.

This form of the equalizer is used when all bands are computed in parallel and results are simultaneously available from all filter banks. Thanks to the associativity of the addition operation, the STMP35xx implements a final addition stage, see Figure 68. “FILCO BIQUAD IIR Parallel Filter” on page 163. In this parallel mode, each filter bank is run as a separate kick. For a five bank equalizer there are five separate filters to run and each runs to completion before the next one is started. In parallel mode, just as a filter output sample is computed, it is added to the value in the output sample array. Thus the superposition of all filter banks is formed in the output buffer where all filter results for a given sample are added together.

To simplify the number of operations, set the HW_FILCO_CSR_IIRMODE to cascade (zero) for the first filter bank and then set it to parallel (one) for all subsequent filters.

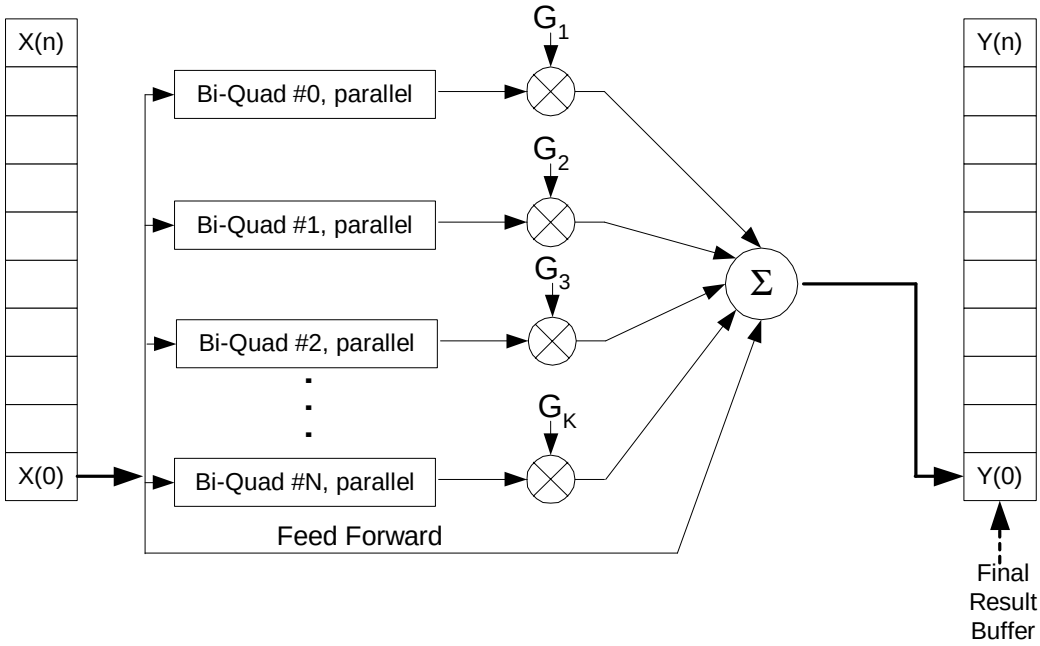


Figure 67. FILCO BiQuad Parallel EQ Application

Setting up for the parallel mode bi-quad filter of Figure 68 is essentially the same as setting up the cascade case, except for setting **HW_FILCO_CSR_IIRMODE** to one. Of course different final gain values are indicated for **HW_FILCO_IIRGAIN_IIRGAIN**. The coefficients are computed for a very different filter shape, namely bandpass filters instead of notch filters.

Note that no additional zero padding words are required for the IIR filter modes as opposed to the FIR modes.

The addition of the current output buffer value to the freshly computed IIR output is performed with overflow detection, saturation and convergent rounding.

13.4. IIR Mode Zero Crossing Detector and Gain Substitution

FILCO contains two zero crossing detect status bits, one for the left channel and one for the right channel. These bits are cleared at the time an IIR filter operation is kicked off. These bits can be seen in:

HW_FILCO_ZC_STATUS_ZERO_CROSS_DETECT_STATUS.

When a bit is set it indicates that a zero crossing detect was enabled for the corresponding channel, and the zero crossing has been detected within that filter run. Once the status bit is set then the 24-bit zero crossing gain from register HW_FILCO_ZC_GAIN is substituted for the value in HW_FILCO_IIRGAIN in the bi-quad filter loop.

Thus one can implement a mute function by setting a very low gain value in the zero crossing gain register. When the zero crossing is detected, then the filter output is essentially muted while the signal is in the vicinity of its zero crossing. Zero crossing are detected at the output of the Bi-Quad IIR before the gain is multiplied. Thus one

can use the zero crossing detector to go from play to mute and also from mute to play.

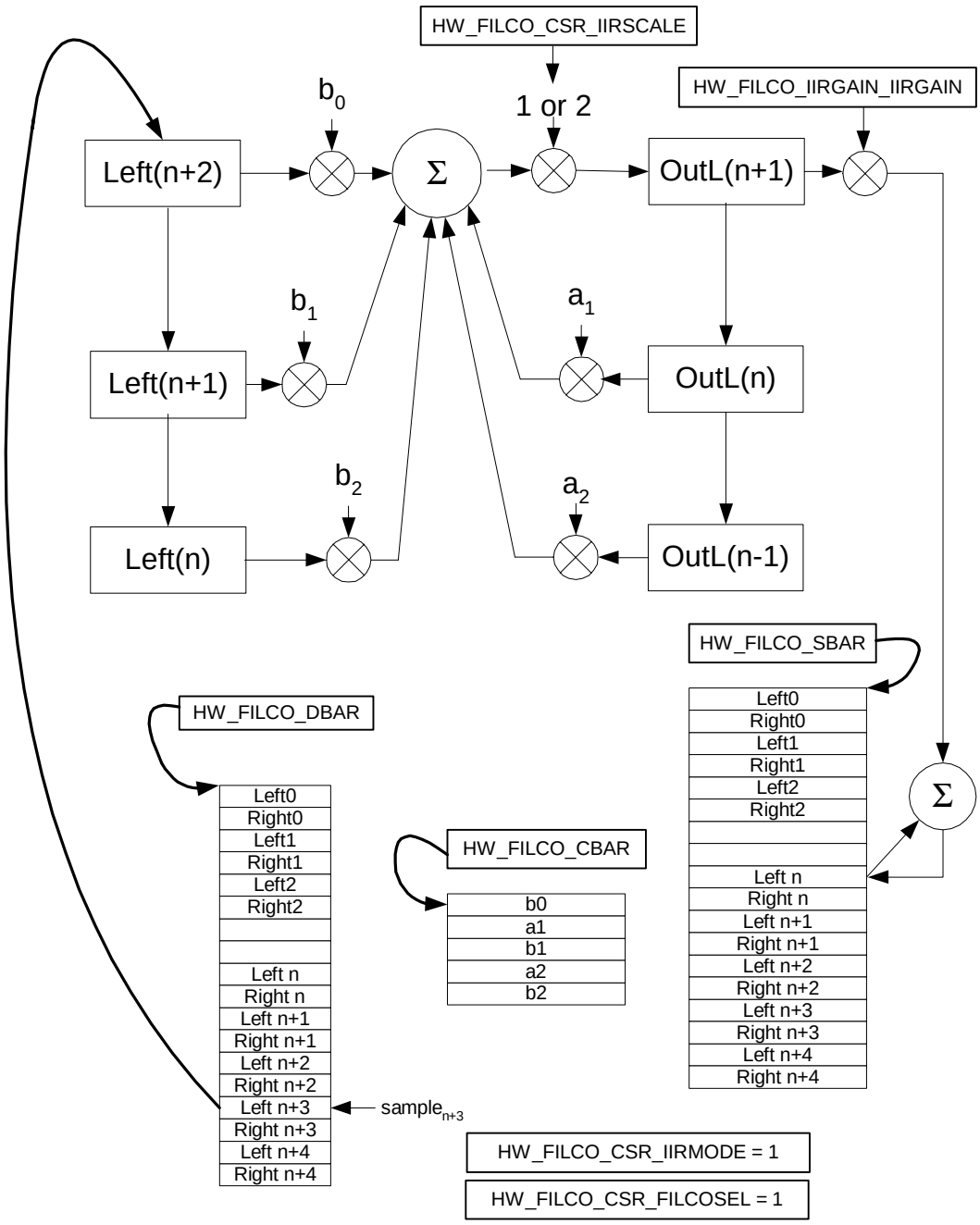


Figure 68. FILCO BIQUAD IIR Parallel Filter

13.5. FILCO Programmable Registers

The following registers are available for DSP programmer access and control of the FILCO Filter Coprocessor.

13.5.1. FILCO Control/Status Register

This register provides overall control of the filter coprocessor for transactions and interrupts.

HW_FILCO_CSR X:$FC00

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CLKGT				DMADLY				FIRMODE		IIRSCALE	IIRMODE	FILCOSEL				DMACOEF		DMASAMP		DMAOUT			

Table 165. HW_FILCO_CSR

BITS	LABEL	RW	RESET	DEFINITION
23	**CLKGT**	RW	1	Clock Gate for the Filco functions
22:20	**RSRVD**	R	000	**Reserved –** Must be written with 0.
19:16	**DMADLY**	RW	$0	Delay each DMA request by a number of clocks $0 = 1 clock delay . . . $F = 16 clock dlelays.
15:14	**FIRMODE**	RW	00	FIR Mode filter type 00 = FIR 1:1 normal mode 01 = FIR 1:2 interpolation mode 10 = FIR 2:1 decimation mode
13	**IIRSCALE**	RW	0	IIR final scale mode 0 = Multiply IIR filter result by 1x 1 = Multiply IIR filter result by 2x
12	**IIRMODE**	RW	0	IIR Mode 0 = Cascade 1 = Parallel (add to output buffer)
11	**FILCOSEL**	RW	0	Filter Type: 0 = FIR 1 = IIR
10:8	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
7:6	**DMACOEF**	RW	00	00 - DMA input coefficients from on-chip XRAM 01 - DMA input coefficients from on-chip YRAM 10 - DMA input coefficients from on-chip PRAM 11- Reserved
5:4	**DMASAMP**	RW	00	00 - DMA input samples from on-chip XRAM 01 - DMA input samples from on-chip YRAM 10 - DMA input samples from on-chip PRAM 11- Reserved

Table 166. FILCO Control/Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
3:2	**DMAOUT**	RW	00	00 - DMA into on-chip XRAM 01 - DMA Into on-chip YRAM 10 - DMA into on-chip PRAM 11- Reserved
1:0	**RSRVD**	R	$0	**Reserved –** Must be written with 0.

Table 166. FILCO Control/Status Register Description (Continued)

13.5.2. FILCO SAT Register

This register provides a debug view of the saturation state of each of the eight accumulators.

HW_FILCO_SAT X:$FC01

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																SAT							

Table 167. HW_FILCO_SAT

BITS	LABEL	RW	RESET	DEFINITION
23:8	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
7:0	**SAT**	RW	$00	Overflow status for each accumulator.These bits will be set if an accumulator store referenced an accumulator with an overflow (out range condition). Once these sticky bits are set they will remain set unto cleared. Set to one to clear these bits.

Table 168. FILCO Accumulator Status Register Description

13.5.3. FILCO Coefficient Base Address Register

This register provides a pointer to the array of coefficients to be used for the current filter.

HW_FILCO_CBAR X:$FC02

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								CBAR															

Table 169. HW_FILCO_CBAR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
15:0	**CBAR**	RW	$0000	Address Location of first coefficient.These bits work in conjunction with **HW_FILCO_CSR_DMACOEF** bit field which determines the source on-chip RAM.

Table 170. FILCO Coefficient Base Address Register Description

13.5.4. FILCO TAP COUNT Register

This register contains the number of coefficients in an FIR filter. Remember that either three or six words of zero padding must be appended to the end of the coefficient array, however, they should be included in the tap count value stored here. In addition, this register contains the number of even or odd phase taps for a 1:2 interpolating FIR filter, not the sum of the two. Finally, all bi-quad IIR filters have exactly five coefficients and therefore ignore any value written to this register.

HW_FILCO_TAPCNT X:$FC03

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																CMR							

Table 171. HW_FILCO_TAPCNT

BITS	LABEL	RW	RESET	DEFINITION
23:8	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
7:0	**CMR**	RW	$00	Size of filter, i.e. number of coefficients (256 taps). This field is ignored for IIR registers which always have exactly five coefficients.

Table 172. FILCO Coefficient Modulo Register Description

13.5.5. FILCO FIR Coefficient Current Position Register

This register provides a diagnostic view of the offset to add to the CBAR to find the address from which the current coefficient will be fetched.

HW_FILCO_CCPR X:$FC04

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											CCPR												

Table 173. HW_FILCO_CCPR

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
12:0	**CCPR**	R	$0000	Current offset from coefficient base address.

Table 174. FILCO Coefficient Current Position Register Description

13.5.6. FILCO Spare Register

This register provides bits that can read or written from DSP software for use in generating metal mask patches to the silicon.

HW_FILCO_SPARE X:$FC05

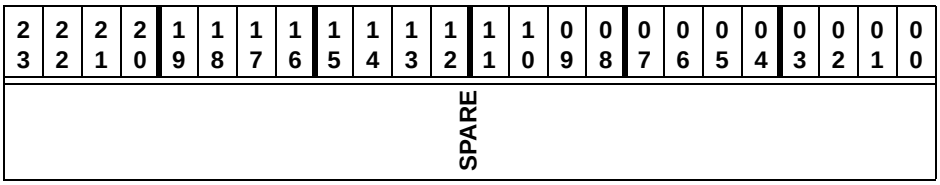


Table 175. HW_FILCO_SPARE

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SPARE**	RW	$000000	Spare bits for patch gates

Table 176. FILCO SPARE Register Description

13.5.7. FILCO Interrupt Configuration Register

This register holds the completion interrupt status bit and the completion interrupt enable for the FILCO.

HW_FILCO_INTR X:$FC0B

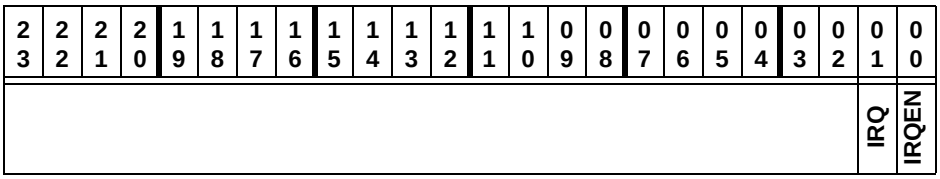


Table 177. HW_FILCO_INTR

BITS	LABEL	RW	RESET	DEFINITION
23:2	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
1	**IRQ**	RW	0	FILCO interrupt is a status bit indicating a completion interrupt has been generated. IRQ occurs when the filter has finished processing a buffer, i.e. WORDCOUNT = 0.
0	**IRQEN**	RW	0	FILCO interrupt enable. Set to one to enable interrupt generation whenever **IRQ** is a one.

Table 178. FILCO Interrupt Configuration Register Description

13.5.8. FILCO Kick Register

This register provides access to the bits that control kicking off a transaction as well as stopping that transaction.

HW_FILCO_KICK X:$FC0C

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PAUSE																						UNKICK	KICK

Table 179. HW_FILCO_KICK

BITS	LABEL	RW	RESET	DEFINITION
23:21	**PAUSE**	RW	000	101- Pause the FILCO at any point of the operation. All other values allow normal operation.
20:2	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
1	**UNKICK**	RW	0	Write a one to this bit to temporarily freeze states to allow for processing new samples. When read, a one indicates that the FILCO is busy; a zero indicates that the FILCO is idle.
0	**KICK**	RW	0	Set the KICK bit to one to start the filter process in the FILCO. Kick is automatically cleared when the word count reaches zero.

Table 180. FILCO Kick Register Description

13.5.9. FILCO Input Sample (Data) Base Address Register

This register contains the base address pointing to the input *Data* sample buffer.

HW_FILCO_DBAR X:$FC0D

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								DBAR															

Table 181. HW_FILCO_DBAR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
15:0	**DBAR**	RW	$0000	Address of the first input sample *data* word. These bits work in conjunction with **HW_FILCO_CSR_DMADATA** bit field which determines the source on-chip RAM.

Table 182. FILCO Input Sample Data Base Address Register Description

13.5.10. FILCO Sample Data Buffer Modulo Register

This register provides the modulo address for the input sample *data* circular buffer. This register holds the count of the maximum number words in the buffer. All data references from the sample data buffer fall within the range **HW_FILCO_DBAR** to **HW_FILCO_DBAR** plus **HW_FILCO_DMR**.

HW_FILCO_DMR X:$FC0E

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											DMR												

Table 183. HW_FILCO_DMR

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
12:0	**DMR**	RW	$000	Size of data input buffer (up to 8192 words)

Table 184. FILCO Sample Data Buffer Modulo Register Description

13.5.11. FILCO Sample Count Register

FILCO uses base plus index addressing to access samples in the input sample buffer. The 13 bit index is constrained to wrap from the modulo register value back to zero as successive input samples are fetched. There are two sample index register in the FILCO. The first one, called the start index is used to keep track of where the next filter basis will begin. The second one, called the running index, is used to step backward in time, fetching each sample from the filter's basis.

Samples are fetched from locations determined by adding the value in HW_FILCO_DBAR to the running index register.

The writes to this register are directed to the start index register. Reads come from the running index register. When FILCO is not in the "KICKED" they should be equal so that saving the read value as part of a context switch will provide the correct value to write to HW_FILCO_DCPR when the context is switched back in.

Values written to the HW_FILCO_DCPR will not be visible when read back. Values written to the start index register are copied to the running index register after a kick start.

HW_FILCO_DCPR X:$FC0F

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											DCPR												

Table 185. HW_FILCO_DCPR

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
12:0	**DCPR**	RW	$0001	Current input sample counter. Reads return the value of the running index register. Reads indicate the first sample to be read on the next set of output samples. Reads should only occur while the FILCO is not in the KICKED state or while it is paused. Writes are directed to the start index register. Only odd values can be written to this start index register.

Table 186. FILCO Sample Count Register Description

13.5.12. FILCO Output Sample Base Address Register

This register holds the base address pointing to the on-chip RAM buffer which will receive the filtered output *samples*.

HW_FILCO_SBAR X:$FC10

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
															SBAR								

Table 187. HW_FILCO_SBAR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
15:0	**SBAR**	RW	$0000	Address of first filtered output sample word. These bits work in conjunction with **HW_FILCO_CSR_DMAOUT** bit field which determines the destination on-chip RAM.

Table 188. FILCO Output Sample Base Address Register Description

13.5.13. FILCO Output Sample Modulo Register

This register provides the modulo size for the output *sample* circular buffer.

HW_FILCO_SMR X:$FC11

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											SMR												

Table 189. HW_FILCO_SMR

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
12:0	**SMR**	RW	$0000	Size of output sample data buffer.

Table 190. FILCO Output Sample Modulo Register Description

13.5.14. FILCO Output Sample Current Position Register

This register provides a diagnostic and context switch view of the offset to add to the SBAR to find the address to which the next output sample word will be written.

HW_FILCO_SCPR X:$FC12

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											SCPR												

Table 191. HW_FILCO_SCPR

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
12:0	**SCPR**	RW	$0000	Current output sample pointer.

Table 192. FILCO Output Sample Current Position Register Description

13.5.15. FILCO Tail Pointer Register

This register provides a read only view of the address of the last input data sample fetched for an FIR filter.

HW_FILCO_TPTR X:$FC13

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
															TAIL_PNTR								

Table 193. HW_FILCO_TPTR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
15:0	**TAIL_PNTR**	R	$0000	DMA address (pointer) of last FIR filter sample fetched for current input data sample basis.

Table 194. FILCO Tail Pointer Register Description

13.5.16. FILCO Unity Gain Register

This register provides a unity gain feed forward path for one or both channels.

HW_FILCO_UGAIN X:$FC17

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																						PER_CHANNEL	UNITY_GAIN

Table 195. HW_FILCO_UGAIN

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
1	**PER_CHANNEL**	RW	0	Use each channel zero cross detect bit to enable unity gain on a channel by channel basis.
0	**UNITY_GAIN**	RW	0	Force unity gain in IIR mode. Since +1.0 is not representable in a 24-bit twos complement fractional number but is a highly desirable value for the final IIR gain multiplier, this bit provides a way to specify an exact multiply by +1.0.

Table 196. FILCO Unity Gain Register Description

13.5.17. FILCO IIR Gain Register

This register provides an overall gain control for the output of an IIR filter step. This is a convenient place to set the attenuation for each equalizer band in a parallel IIR implementation of an equalizer. If the **HW_FILCO_UGAIN_UNITY_GAIN** bit is set then a value of 1.0 is used in place of this value. If the zero cross detector is in use and a zero crossing has been detected then the HW_FILCO_ZC_GAIN value will be used in place of this registers value to set the final gain of an IIR filter.

HW_FILCO_IIRGAIN X:$FC18

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
IIRGAIN																							

Table 197. HW_FILCO_IIRGAIN

BITS	LABEL	RW	RESET	DEFINITION
23:0	**IIRGAIN**	RW	$000000	IIR mode volume register, same register as coefficient register 5.

Table 198. FILCO IIR Volume Register Description

13.5.18. FILCO Word Count Register

This register holds the count of the number of output samples to process (count of individual output sample words, not word pairs).

HW_FILCO_WORDCOUNT X:$FC1B

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											WCNT												

Table 199. HW_FILCO_WORDCOUNT

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
12:0	**WCNT**	RW	$0000	Number of single samples to be processed.

Table 200. FILCO IIR Volume Register Description

13.5.19. FILCO Zero Cross Status Register (IIRMODE ONLY)

This register provides controls for the IIR zero crossing detector mode.

HW_FILCO_ZC_STATUSX:$FC1C

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																				ZERO_CROSS_DETECT_STATUS		ZERO_CROSS_DETECT_EN	

Table 201. HW_FILCO_ZC_STATUS

BITS	LABEL	RW	RESET	DEFINITION
23:4	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
3:2	**ZERO_CROSS_DETECT_STATUS**	RW	00	Zero Cross Detect Status 00 - Zero cross not detected. 01 - even channel detected. 10 - odd channel detected. 11 - both channels detected.
1:0	**ZERO_CROSS_DETECT_EN**	RW	00	Zero Cross Detect Enable 00 - Zero cross detect disabled. 01 - right channel, i.e. odd addresses enabled. 10 - left channel, i.e. even addresses enabled. 11 - zero crossing detector enabled for both channels.

Table 202. FILCO Zero Cross Status Register Description

13.5.20. FILCO Zero Cross Gain Setting Register (IIRMODE ONLY)

This register provides the gain to be substituted when a channel goes through a zero crossing.

HW_FILCO_ZC_GAIN X:$FC1D

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ZCGAIN																							

Table 203. HW_FILCO_ZC_GAIN

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ZCGAIN**	RW	$000000	Gain setting to be used once a zero cross is detected.

Table 204. FILCO Zero Cross Gain Setting Register Description

13.5.21. FILCO Context Acc0 Register (IIRMODE ONLY)

This register provides access to accumulator zero for context switching between IIR sub-buffers.

HW_FILCO_CTXT_A0R X:$FC20

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ACCUM																							

Table 205. HW_FILCO_CTXT_A0R

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ACCUM**	RW	$000000	Accumulator 0, save and restore this register between IIR sub buffers. Initialize to zero for first sub-buffer.

Table 206. FILCO Context Acc0 Register Description

13.5.22. FILCO Context Acc1 Register (IIRMODE ONLY)

This register provides access to accumulator one for context switching between IIR sub-buffers.

HW_FILCO_CTXT_A1R X:$FC21

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ACCUM																							

Table 207. HW_FILCO_CTXT_A1R

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ACCUM**	RW	$000000	Accumulator 1, save and restore this register between IIR sub buffers. Initialize to zero for first sub-buffer.

Table 208. FILCO Context Acc1 Register Description

13.5.23. FILCO Context Acc2 Register (IIRMODE ONLY)

This register provides access to accumulator two for context switching between IIR sub-buffers.

HW_FILCO_CTXT_A2R X:$FC22

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ACCUM																							

Table 209. HW_FILCO_CTXT_A2R

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ACCUM**	RW	$000000	Accumulator 2, save and restore this register between IIR sub buffers.

Table 210. FILCO Context Acc0 Register Description

13.5.24. FILCO Context Acc3 Register (IIRMODE ONLY)

This register provides access to accumulator three for context switching between IIR sub-buffers.

HW_FILCO_CTXT_A3R X:$FC23

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ACCUM																							

Table 211. HW_FILCO_CTXT_A3R

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ACCUM**	RW	$000000	Accumulator 3, save and restore this register between IIR sub buffers.

Table 212. FILCO Context Acc3 Register Description

13.5.25. FILCO Switch Register (IIRMODE ONLY)

This register provides access to the ping pong switch bit for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_SWX:$FC24

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
XX																							BIT

Table 213. HW_FILCO_CTXT_SW

BITS	LABEL	RW	RESET	DEFINITION
23:1	**RSRVD**	R	$000000	**Reserved –** Must be written with 0.
0	**BIT**	RW	0	Ping Pong Switch bit. Save and restore this bit at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 214. FILCO Switch Register Description

13.5.26. FILCO Sample 0 Register (IIRMODE ONLY)

This register provides access to sample 0 of the sample register file for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_S0X:$FC25

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SAMPLE																							

Table 215. HW_FILCO_CTXT_S0

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SAMPLE**	RW	$000000	Sample 0 of the sample register file. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 216. FILCO Sample 0 Register Description

13.5.27. FILCO Sample 1 Register (IIRMODE ONLY)

This register provides access to sample 1 of the sample register file for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_S1X:$FC26

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SAMPLE																							

Table 217. HW_FILCO_CTXT_S1

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SAMPLE**	RW	$000000	Sample 1 of the sample register file. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 218. FILCO Sample Register Description

13.5.28. FILCO Sample 2 Register (IIRMODE ONLY)

This register provides access to sample 2 of the sample register file for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_S2X:$FC27

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SAMPLE																							

Table 219. HW_FILCO_CTXT_S2

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SAMPLE**	RW	$000000	Sample 2 of the sample register file. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 220. FILCO Sample 2 Register Description

13.5.29. FILCO Sample 3 Register (IIRMODE ONLY)

This register provides access to sample 3 of the sample register file for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_S3X:$FC28

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SAMPLE																							

Table 221. HW_FILCO_CTXT_S3

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SAMPLE**	RW	$000000	Sample 3 of the sample register file. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 222. FILCO Sample 3 Register Description

13.5.30. FILCO Sample 4 Register (IIRMODE ONLY)

This register provides access to sample 4 of the sample register file for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_S4X:$FC29

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SAMPLE																							

Table 223. HW_FILCO_CTXT_S4

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SAMPLE**	RW	$000000	Sample 4 of the sample register file. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 224. FILCO Sample 4 Register Description

13.5.31. FILCO Sample 5 Register (IIRMODE ONLY)

This register provides access to sample 5 of the sample register file for context switching between IIR sub-buffers.

HW_FILCO_ CTXT_S5X:$FC2A

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SAMPLE																							

Table 225. HW_FILCO_CTXT_S5

BITS	LABEL	RW	RESET	DEFINITION
23:0	**SAMPLE**	RW	$000000	Sample 5 of the sample register file. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 226. FILCO Sample 5 Register Description

13.5.32. FILCO Write Pointer Register (IIRMODE ONLY)

This register provides to the write pointer bits for context switching between IIR sub-buffers.

HW_FILCO_CTXT_WPX:$FC2B

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																						WPTR	

Table 227. HW_FILCO_CTXT_WP

BITS	LABEL	RW	RESET	DEFINITION
23:2	**RSRVD**	R	$000000	**Reserved –** Must be written with 0.
1:0	**WPTR**	RW	$000000	Holds the write pointer value at the end of a sub-buffer. Save and restore this value at IIR context switches. Initialize it to zero at the beginning of a filter run.

Table 228. FILCO Write Pointer Register Description

14. PULSE WIDTH MODULATOR (PWM) CONTROLLER

The STMP35xx contains four PWM output controllers that can be used in place of GPIO pins. Applications include LED brightness control and high voltage generators for electroluminescent lamp (E.L.) display back lights. Independent output control of each phase allows zero, one or hi-Z to be independently selected for the active and inactive phases. Individual outputs can be run in lock step with guaranteed non-overlapping portions for differential drive applications.

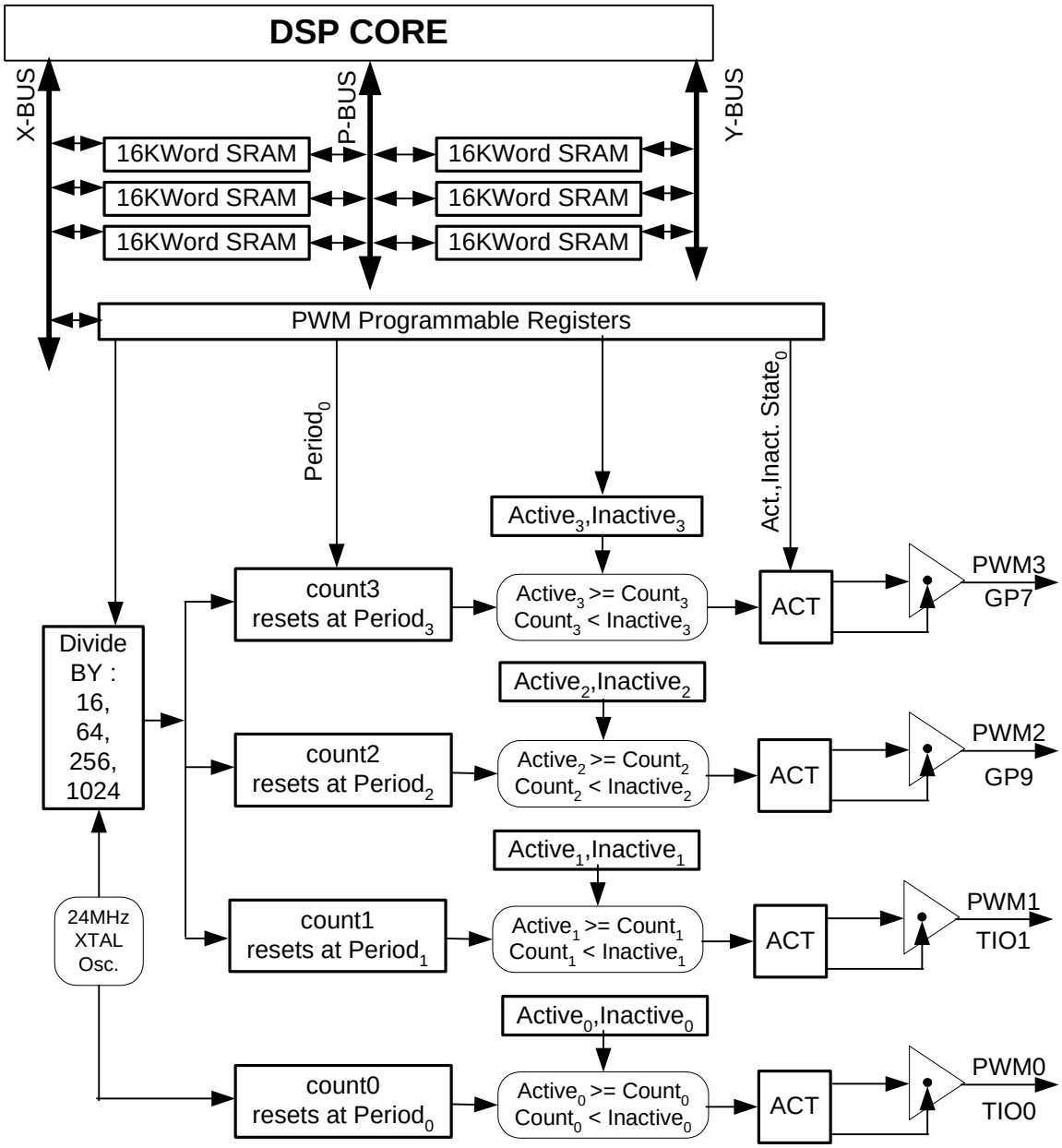


Figure 69. Pulse Width Modulation Controller (PWM) Block Diagram

Figure 69 shows the block diagram of the PWM controller. The controller does not use the DMA. Initial values of Period, Active, and Inactive widths are set for each desired channel. The outputs are selected by phase and then the desired PWM

channels are simultaneously enabled. This effectively launches the PWM outputs to autonomously drive their loads without further DSP intervention. In backlit high voltage applications, a feed forward control can be periodically used to change the count parameters based on LRADC evaluation of the battery state. Feedback control can be provided by assigning one LRADC channel to monitor the integrating capacitor voltage. Care must be taken to protect the LRADC from catastrophic over voltage in this case. For most EL back light applications, open loop control with precision PWM timers based on a stable crystal oscillator is sufficient.

Each PWM channel has a dedicated internal 12 bit counter which increments once for each divided clock period presented from the clock divider. The internal counter resets when it reaches the value stored in the channel control registers, e.g. **HW_PWM_CH0BR_PERIOD**. The Active flip flop is set to one when the internal counter reaches the value stored in **HW_PWM_CH0AR_ACTIVE**. It remains high until the internal counter exceeds the value stored in **HW_PWM_CH0AR_INACTIVE**. These two value define the starting and ending points for the logically *"active"* portion of the waveform. As shown in Figure 70, the actual state on the output for each phase, e.g. active or inactive, is completely controlled by the active and inactive state values in the channel control registers.

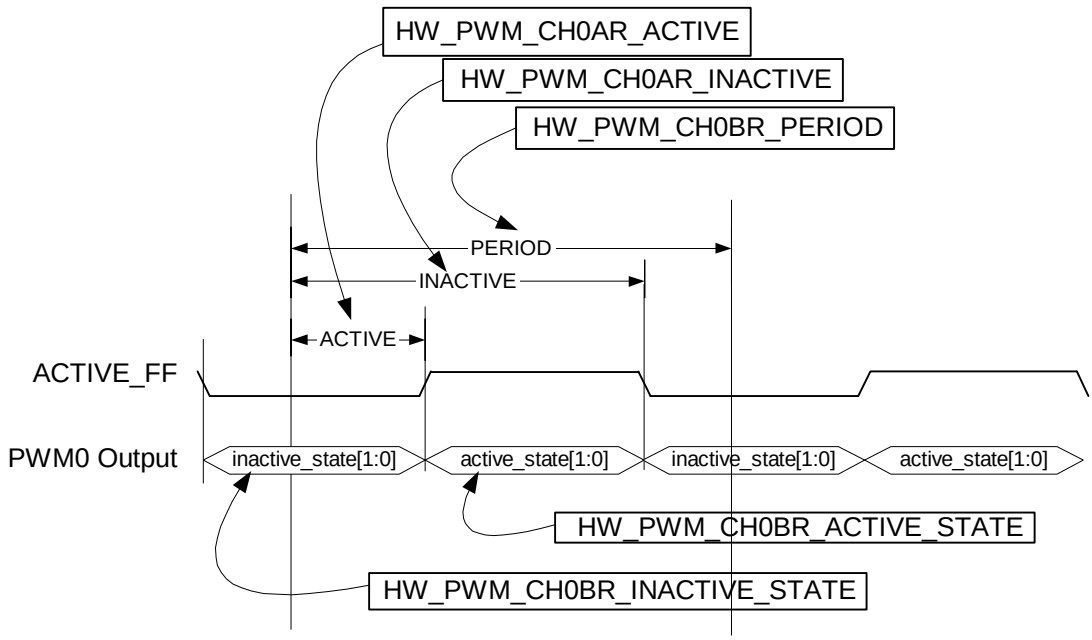
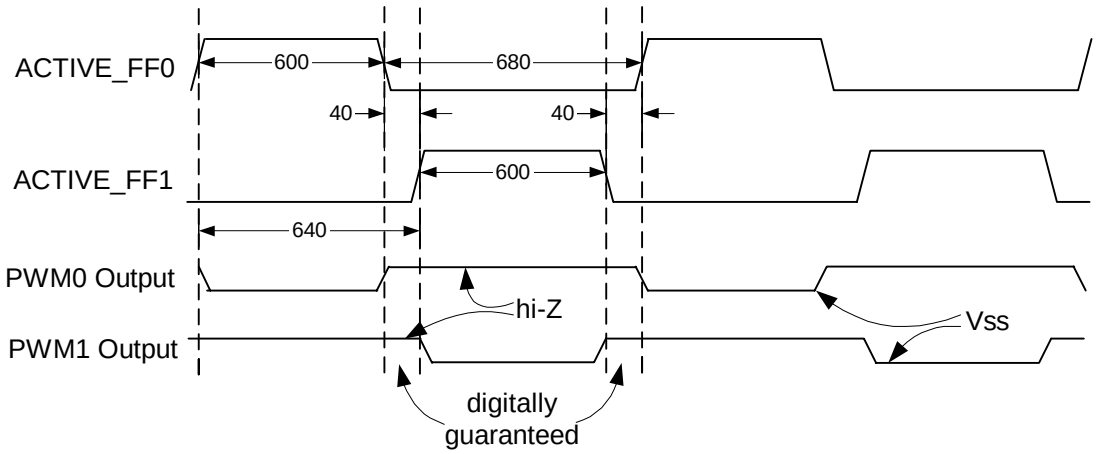


Figure 70. PWM Output Example

The actual values obtainable on the output are shown below, see Figure 72. "PWM Output Driver" on page 184. Notice that one possible state is to turn off the output driver to provide a hi-Z output. This is useful for external circuits that drive E.L. backlights and for direct drive of LEDs.

By setting up two channels in lock step and by setting their low and high states to opposite values, one can generate a differential signal pair that alternates between pulling to Vss and floating to Hi-Z. By creating an appropriate offset in the settings of the two channels with the same period and the same enable instant one can generate differential drive pulses with digitally guaranteed non-overlapping intervals suitable for controlling high voltage switches,



HW_PWMCH0BR_PERIOD = 1280	HW_PWMCH1BR_PERIOD = 1280
HW_PWMCH0AR_ACTIVE = 0	HW_PWMCH1AR_ACTIVE = 640
HW_PWMCH0AR_INACTIVE = 600	HW_PWMCH1AR_INACTIVE = 1240
HW_PWMCH0BR_ACTIVE_STATE = 10	HW_PWMCH1BR_ACTIVE_STATE = 10
HW_PWMCH0BR_INACTIVE_STATE = 00	HW_PWMCH1BR_INACTIVE_STATE = 00

Figure 71. PWM Differential Output Pair Example

In the above example, a differential pair is established using channel zero and channel one. The period is set for 1280 divided clocks for both channels. All active phases are set for 600 divided clocks. There is a 40 divided clock guaranteed off-time between each active phase. Since this is based on a crystal oscillator it is a very stable non-overlapping period. The total period is also a very stable crystal oscillator based time interval. In this example, the active phases are pulled to Vss (ground) while the inactive phases are allowed to float to a hi-Z state.

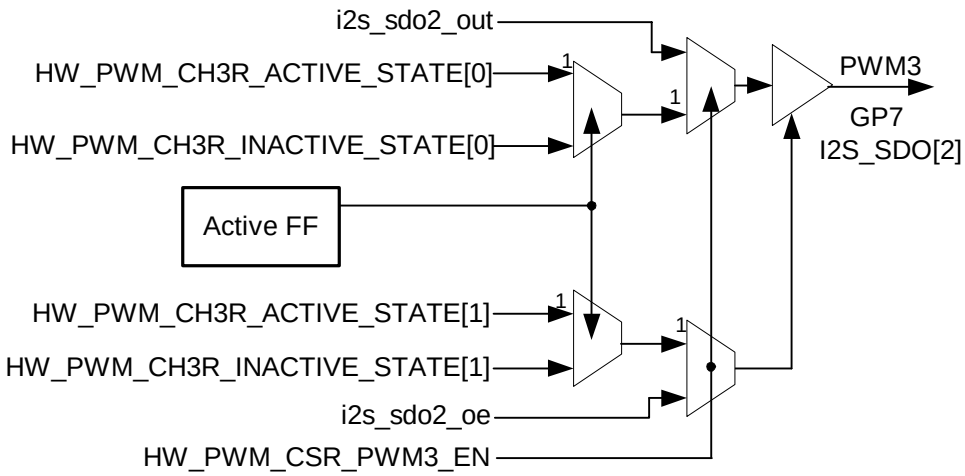


Figure 72. PWM Output Driver

Figure 72 shows the generation of the PWM Channel three output. This channel controls the GP7 output pin when **HW_PWM_CSR_PWM3_EN** is set to one. The output pin can be set to a “ZERO”, a “ONE”, or left to float in the high impedance state. These choices can be independently made for either the active or inactive phase of the output, see Figure 72. “PWM Output Driver” on page 184.

14.1. PWM Programmable Registers

The following registers are available for DSP programmer access and control of the PWM controller.

14.1.1. PWM Configuration Register

This register provides overall control of the four PWM channels.

HW_PWM_CSR X:$FA31

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MSTR_EN														CDIV						PWM3_EN	PWM2_EN	PWM1_EN	PWM0_EN

Table 229. HW_PWM_CSR

BITS	LABEL	RW	RESET	DEFINITION
23	**MSTR_EN**	RW	0	Master Enable is set to one to start the clock divider and to enable any PWM channel operation.
22:10	**RSRVD**	R	$000	**Reserved –** Must be written with 0.
9:8	**CDIV**	RW	00	Clock divider ratio to apply to crystal clock frequency. 00 = divide by 16 01 = divide by 64 10 = divide by 256 11 = divide by 1024 **NOTE**: Revision TA2 of the 3500 uses the crystal clock for PWM 0 instead of the output of this clock divider. **NOTE**: Be sure to set **HW_CCR_XTLEN** to use the PWM.
7:4	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
3	**PWM3_EN**	Rw	0	Enables PWM Channel 3 to begin cycling when set to one. Setting this bit to one also enables PWM3 onto the output pin in place of the other uses for that pin, see Section 31. “PIN DESCRIPTION” on page 368.
2	**PWM2_EN**	RW	0	Enables PWM Channel 2 to begin cycling when set to one.Setting this bit to one also enables PWM2 onto the output pin in place of the other uses for that pin, see Section 31. “PIN DESCRIPTION” on page 368.

Table 230. PWM Configuration and Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
1	**PWM1_EN**	RW	0	Enables PWM Channel 1 to begin cycling when set to one.Setting this bit to one also enables PWM0 onto the output pin in place of the other uses for that pin, see Section 31. “PIN DESCRIPTION” on page 368.
0	**PWM0_EN**	RW	0	Enables PWM Channel 0 to begin cycling when set to one.Setting this bit to one also enables PWM0 onto the output pin in place of the other uses for that pin, see Section 31. “PIN DESCRIPTION” on page 368.

Table 230. PWM Configuration and Status Register Description (Continued)

14.1.2. PWM Channel 0 A Register

This register controls PWM channel 0.

HW_PWM_CH0AR X:$FA32

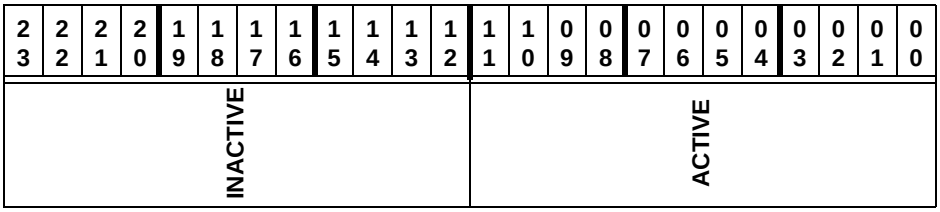


Table 231. HW_PWM_CH0AR

BITS	LABEL	RW	RESET	DEFINITION
23:12	**INACTIVE**	RW	$000	Number of divided xtal clocks to count before resetting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than this value to reset the ACTIVE FF.
11:0	**ACTIVE**	RW	$000	Number of divided xtal clocks to count before setting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than or equal to this value. If the internal count is greater than or equal then the ACTIVE FF is set to one.

Table 232. PWM Channel 0 A Register Description

14.1.3. PWM Channel 0 B Register

This register controls PWM channel 0.

HW_PWM_CH0BR X:$FA33

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ACTIVE_STATE		INACTIVE_STATE		PERIOD											

Table 233. HW_PWM_CH0BR

BITS	LABEL	RW	RESET	DEFINITION
15:14	**ACTIVE_STATE**	RW	00	The logical active state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
13:12	**IN_ACTIVE_STATE**	RW	00	The logical inactive state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
11:0	**PERIOD**	RW	$000	Number of divided xtal clocks in entire period of the PWM waveform minus one, i.e. to obtain six clocks in the actual period then set this field to five.

Table 234. PWM Channel 0 B Register Description

14.1.4. PWM Channel 1 A Register

This register controls PWM channel 1.

HW_PWM_CH1AR X:$FA34

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
INACTIVE												ACTIVE											

Table 235. HW_PWM_CH1AR

BITS	LABEL	RW	RESET	DEFINITION
23:12	**INACTIVE**	RW	$000	Number of divided xtal clocks to count before resetting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than this value to reset the ACTIVE FF.
11:0	**ACTIVE**	RW	$000	Number of divided xtal clocks to count before setting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than or equal to this value. If the internal count is greater than or equal then the ACTIVE FF is set to one.

Table 236. PWM Channel 1 A Register Description

14.1.5. PWM Channel 1 B Register

This register controls PWM channel 1.

HW_PWM_CH1BR X:$FA35

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ACTIVE_STATE		INACTIVE_STATE		PERIOD											

Table 237. HW_PWM_CH1BR

BITS	LABEL	RW	RESET	DEFINITION
15:14	**ACTIVE_STATE**	RW	00	The logical active state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
13:12	**IN_ACTIVE_STATE**	RW	00	The logical inactive state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
11:0	**PERIOD**	RW	$000	Number of divided xtal clocks in entire period of the PWM waveform.

Table 238. PWM Channel 1 B Register Description

14.1.6. PWM Channel 2 A Register

This register controls PWM channel 2.

HW_PWM_CH2AR X:$FA36

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
INACTIVE												ACTIVE											

Table 239. HW_PWM_CH2AR

BITS	LABEL	RW	RESET	DEFINITION
23:12	**INACTIVE**	RW	$000	Number of divided xtal clocks to count before resetting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than this value to reset the ACTIVE FF.
11:0	**ACTIVE**	RW	$000	Number of divided xtal clocks to count before setting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than or equal to this value. If the internal count is greater than or equal then the ACTIVE FF is set to one.

Table 240. PWM Channel 2 A Register Description

14.1.7. PWM Channel 2 B Register

This register controls PWM channel 2.

HW_PWM_CH2BR X:$FA37

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ACTIVE_STATE		INACTIVE_STATE		PERIOD											

Table 241. HW_PWM_CH2BR

BITS	LABEL	RW	RESET	DEFINITION
15:14	**ACTIVE_STATE**	RW	00	The logical active state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
13:12	**IN_ACTIVE_STATE**	RW	00	The logical inactive state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
11:0	**PERIOD**	RW	$000	Number of divided xtal clocks in entire period of the PWM waveform.

Table 242. PWM Channel 2 B Register Description

14.1.8. PWM Channel 3 A Register

This register controls PWM channel 3.

HW_PWM_CH3AR X:$FA38

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
INACTIVE												ACTIVE											

Table 243. HW_PWM_CH3AR

BITS	LABEL	RW	RESET	DEFINITION
23:12	**INACTIVE**	RW	$000	Number of divided xtal clocks to count before resetting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than this value to reset the ACTIVE FF.
11:0	**ACTIVE**	RW	$000	Number of divided xtal clocks to count before setting the ACTIVE flip flop for this channel. The internal count of the channel is compared for greater than or equal to this value. If the internal count is greater than or equal then the ACTIVE FF is set to one.

Table 244. PWM Channel 3 A Register Description

14.1.9. PWM Channel 3 B Register

This register controls PWM channel 3.

HW_PWM_CH3BR X:$FA39

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ACTIVE_STATE		INACTIVE_STATE		PERIOD											

Table 245. HW_PWM_C30BR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	$0	**Reserved –** Must be written with 0.
15:14	**ACTIVE_STATE**	RW	00	The logical active state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1

Table 246. PWM Channel 3 B Register Description

BITS	LABEL	RW	RESET	DEFINITION
13:12	**IN_ACTIVE_STATE**	RW	00	The logical inactive state is mapped to a physical state at the output pins, as follows: 00,01 = hi-Z 10 = 0 11 = 1
11:0	**PERIOD**	RW	$000	Number of divided xtal clocks in entire period of the PWM waveform.

Table 246. PWM Channel 3 B Register Description (Continued)

15. I²C INTERFACE

The I²C is a standard 2 wire serial interface used to connect the chip with peripherals or host controllers. This interface provides a standard speed (up to 100kbps) and high speed (up to 400kbps) I²C connection to multiple devices with the chip acting in either I²C master or I²C slave mode. Typical applications for the I²C bus include: EEPROM, LED/LCD, FM Tuner, Real Time Clock, etc.

The I²C port supports multi-master configurations where peripheral devices can send messages without being queried.

15.1. I²C-Specific Implementation

- The I²C block can be configured as either a master or slave device. In master mode it generates the clock (I2C_SCL) and initiates transactions on the data line (I2C_SDA).
- The I²C block can be configured to pack data into 8, 16 or 24 bit words. Data on the I²C bus is always byte oriented.
- The I²C block has a programmable device address for master transactions. It has a fixed seven bit address of $43 = 7'b1000011 for slave transactions. As seen in the eight bit device address byte, this address corresponds to $86 where the least significant bit is the R/W bit.
- When the interface is enabled, it immediately goes into slave mode and searches for a start event. It then looks for a match on its hardwired device address. If an STMP35xx master transaction is started, then the slave is forced to remain in its idle state until the master transaction completes.

15.1.1. I²C Interface External Pins

I2C_SDA – I²C Serial Data. This pin carries all address and data bits.

I2C_SCL – I²C Serial Clock. This pin carries the clock used to time the address & data.

Pull-up resistors are required on both of the I²C lines as all of the I²C drivers are open drain (pull-down only). Typically, external 2k-Ohm resistors are used to pull the signals up to VddIO.

Note: See Table 495 for I²C pin placement.

15.1.2. I^2C Interface Control/Status Register

HW_I2CCSR X:$FFE7

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		ONEBYTE	TUFLCL	ROFLCL	SUBA	LWORD	BCNT		ACKF	TUFL	ROFL	TREQ	WL		RWN	TDE	RDR	MODE	TIE	ARBLOST	BUSY	RIE	I2C_EN

Table 247. HW_I2CCSR

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**ONEBYTE**	RW	1	**Master One Byte Transfer Mode –** When set to one, this bit enables a special one payload byte transfer mode in which the sub address bytes are not generated. This master only mode allows the transmission of a single I2C address byte from HW_I2CDAT[23:16] and one data byte taken from HW_I2CDAT[15:8].
20	**TUFLCL**	RW	0	**Transmitter Underflow Clear bit –** Setting this bit clears both the TDE and TUFL bits. When an underflow condition occurs the correct procedure is to write data to the HW_I2CDATR register and then send a high pulse on the TUFLCL to clear the TUFL bit. The TUFLCL bit must be manually reset by software after setting it to clear the TUFL or TDE bit.
19	**ROFLCL**	RW	0	**Receive Overflow Clear Flag –** Setting this bit clears both the RDR and ROFL bits. The correct procedure is to read the HW_I2CDATR register followed by a high pulse on the ROFLCL bit. The ROFLCL bit must be manually reset by software after setting it to clear the ROFL or RDR bit.
18	**SUBA**	R	0	**Sub Address –** This bit is set when the Master is transmitting two bytes of sub address following the slave address byte.
17	**LWORD**	RW	0	**Last Word –** This bit is set in master mode when the processor has written its last word to the I2C transmit register or upon receiving the second last word. The WL field of this register decides the size of the last block of data received by the master. This bit is cleared when the stop is generated. If a receive overflow condition occurs in the master, at least two bytes must be received before a stop is generated i.e. if the WL field is 0 then the last word bit must not be set on this receive.
16:15	**BCNT**	R	00	**Byte Count –** These bits hold the number of bytes received by the device in either master or slave mode. **BCNT Bytes received** 01 1 byte 10 2 bytes 00 3 bytes
14	**ACKF**	RW	0	**Acknowledge Failure –** When this bit is set, no acknowledge has been returned.
13	**TUFL**	R	0	**Transmitter Underflow –** An underflow occurs when TDE = 1 and another transfer is triggered copying the contents of the data register into the shift register.
12	**ROFL**	R	0	**Receiver Overflow –** An overflow occurs when RDR = 1 and the shift register performs another parallel load of the data register. The shift register is 8 bits wide for the master receiver, 24 bits wide for the slave receiver. Cleared by a read of the HW_I2CDATR register followed by a high pulse on the ROFLCL bit.

Table 248. I^2C Interface Control/Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
11	**TREQ**	RW	0	**Master Transaction Request –** This bit is used to request a transfer with the I²C in master mode. It is pulsed high (bset,bclr) to request the transfer and the mode bit must already be driven for either standard or fast mode. Upon receiving the pulse the I²C master section will generate a start condition and transmit the slave address. This bit is also used to request a repeated start to the master device. A pulse must be sent, when the processor has written its last word to the I²C transmit register or upon receiving the last word, to generate a repeated start at the end of the subsequent receive or transmit. To request a repeated start immediately after the slave address and two bytes of sub address have been transmitted it is necessary to send a pulse first to start the transfer of the slave address and then another pulse a maximum of ns later to request the repeated start.
10:9	**WL**	RW	00	**Word Length –** Defines the word length being used by the interface for either I²C master or slave modes. **WL** **Word Length** 00 8 bit 10 16 bit 01 24 bit These bits are also used with the LWORD bit so that the device can decide whether the last block of data the master receives is 8, 16 or 24 bits, i.e. wl0, wl1 are set before the transfer and then the LWORD bit is set on the receive of the second last block of data. These bits should not be changed while a transfer is in progress.
8	**RWN**	R	0	**Read/Not Write –** If equal to zero the Master is writing to the Slave, when equal to one the Master is reading from the slave.
7	**TDE**	R	0	**Transmitter Data Empty –** This bit is set when the contents of the transmitter data register are loaded into the shift register. When this bit is set, the processor can write to the Transmit Data register. If transmit interrupts are enabled, when this bit becomes set, an interrupt will be asserted to the processor.
6	**RDR**	R	0	**Receiver Data Ready –** This bit is set when the shift register performs a broadside load to the Receive Data Register. It is cleared by hardware when the processor reads the Receive Data Register. If receive interrupts are enabled, when this bit becomes set, an interrupt will be asserted to the processor.
5	**MODE**	RW	0	**Operating Mode Bit –** When set to one, the I²C is in Fast mode (400Kbits/sec.) and when cleared, the I²C is in Standard mode (100 Kbits/sec). There is noise suppression for both modes. If the device is to be a master, the mode must be set or cleared before TREQ (bit 11) becomes active.
4	**TIE**	RW	0	**Transmitter Interrupt Enable –** When set, the transmit interrupts are enabled to the processor.
3	**ARBLOST**	RW	0	**Arbitration lost –** After every transfer using the master I²C device this bit should be checked to ensure arbitration has not been lost to another master. This bit is cleared by a write to the control/status register.
2	**BUSY**	R	0	**I²C Bus Busy –** When this bit is set the I²C bus is busy. It is set after a START condition is detected and remains set until a STOP condition is detected. If there is arbitration on the I2C_SDA line and our device loses, then the busy bit will remain set until our device detects a stop condition from the winning device.
1	**RIE**	RW	0	**RIE Receiver Interrupt –** Enable When set, the receive interrupts (Receiver Data Ready and Receiver Overflow) are enabled. Both interrupts are processed by reading data from the I²C data register (HW_I2CDATR). If these interrupts are not processed they will continuously reoccur.
0	**I2C_EN**	RW	0	**Peripheral Enable –** When set, the I²C receive and transmit channel are enabled.

Table 248. I²C Interface Control/Status Register Description (Continued)

15.1.3. I²C Data Registers

There are two 24-bit Data Registers located at the same address, Receive and Transmit Data Registers. The Receive Data Register contents are read over the data bus if the processor performs a read of the address X:$FFE6. The Transmit Data Register is written to the external data bus if the processor writes to this same address. The Transmit Data Register should be filled before setting the TREQ bit to initiate a transfer.

HW_I2CDAT X:$FFE6

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DATA																							

Table 249. HW_I2CDAT

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DATA**	RW	0	**Address/DATA register**

Table 250. I²C Data Register Description

On the first write to the transmit data register, the I2C master transmits HW_I2CDAT[23:16] first. Therefore this byte is always the slave address targeted for a master read or write transaction. When the HW_I2CCSR_ONEBYTE bit is set to one in master mode, then HW_I2CDAT_DATA[23:16] is transmitted followed by HW_I2CDAT_DATA[15:8]. Again, the upper byte holds the slave address. For master read transactions, the target slave address is written to HW_I2CDAT_DATA[23:16] before the read transaction is started with the TREQ bit.

15.1.4. I²C Clock Divider Register

The I²C Clock Divider Register controls the division ratio between the system clock (dclk) and the I²C serial clock (I2C_SCL). The value in the FACT field is used to divide the system clock to generate the I²C clock. This clock is further divided by 4 if the MODE bit (bit 5) in the I²C Interface Control/Status register is clear. The I²C clock frequency can be calculated as follows:

Fast mode (HW_I2CCSR:MODE = 1): I²C clock = DCLK / (FACT*2 + 3)

Slow mode (HW_I2CCSR:MODE = 0):I²C clock = DCLK / (FACT*8 + 22)

If the system clock is set to the crystal oscillator with a 24.0 MHz crystal, then the reset value for this register will divide this clock by 550 to give an I²C clock of ~43.6 kHz. During I²C boot modes, the bootrom sets the HW_I2CCSR:MODE bit to switch the I²C block into fast mode — the clock speed in this case will become ~178 kHz.

Note that the fastest master on the bus will determine the high period of the clock while the slowest master OR slave determines the low period of the clock. The HW_I2CDIV counter determines the minimum high period as controlled by the STMP35xx when it is in master mode. This counter controls the minimum low period when the STMP35xx is enabled in master mode. In slave mode, the STMP35xx follows the clock timing established by the controlling master.

HW_I2CDIV X:$FFE5

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
															FACT								

Table 251. HW_I2CDIV

BITS	LABEL	RW	RESET	DEFINITION
23:9	**RSRVD**	R	0	**Reserved –** Must be written with 0.
8:1	**FACT**	**W**	01000010	**Clock divider value bits, NOTE This is a write only register.**
0	**RSRVD**	R	0	**Reserved –** Must be written with 0.

Table 252. I²C Clock Divider Register Description

15.1.5. I²C Interrupt Sources

The I²C port can be used in either interrupt driven or polled modes. If interrupts are enabled, a level-sensitive interrupt will be signaled to the processor upon one of the following events.

ADDRESS	BIT	LABEL	INTERRUPT SOURCE
P:$0030	6	RDR	I²C Receiver Data Ready
P:$0032	12	ROFL	I²C Receiver Overflow
P:$0034	7	TDE	I²C Transmitter Data Empty
P:$0036	13	TUFL	I²C Transmitter Underflow

Table 253. I²C Interrupt Address Map

The interrupt lines are tied directly to the status bits of the Control/Status Register. Clearing these bits through software will remove the interrupt request. The Receiver Data Ready and Transmitter Data Empty requests are automatically removed via hardware when the Receive Data Register is read or the Transmit Data Register is written respectively. The overflow and underflow error signals need to be cleared through software by writing directly to the status bits of the Control/Status Register.

15.2. I²C Bus Protocol

With reference to the clocking scheme shown in the figure below, the I²C interface operates in the following manner:

A START condition is defined as a HIGH to LOW transition on the data line while the I2C_SCL line is held HIGH. After this has been transmitted by the Master, the bus is considered busy. The next byte of data transmitted after the start condition contains the address of the slave in the first 7 bits and the eighth bit tells whether the Master is receiving data from the slave or transmitting data to the slave. When an address is sent, each device in the system compares the first seven bits after a start condition with its address. If they match, the device considers itself addressed by the Master. In slave mode, the I²C write address is 86h, its read address is 87h.

Data transfer with acknowledge is obligatory. The transmitter must release the I2C_SDA line during the acknowledge pulse. The receiver must then pull the data

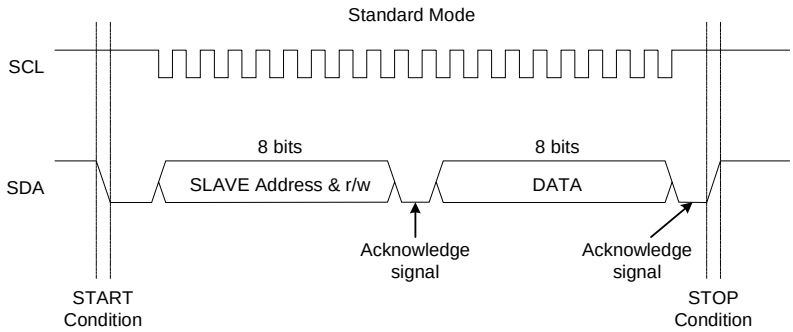


Figure 73. I²C data and clock timing

line LOW so that it remains stable low during the HIGH period of the acknowledge clock pulse. A receiver which has been addressed is obliged to generate an acknowledge after each byte of data has been received.

15.2.1. Slave Mode Protocol

The flow chart for the finite state machine is shown in Figure 74. At device start-up all the registers are reset so that the state is known from that time onward. Once the I²C is enabled the slave will wait to detect a start condition on the I2C_SCL and I2C_SDA lines. Once this is detected the slave will read in 8 bits and check against its own device address which is $86 == 7'b1000011 to see if a master device is trying to start a transfer with our device. If it is our address an acknowledge is sent, otherwise our slave will not acknowledge and will return to state IDLE.

Next the RW is checked. If it is a write operation then the two sub address bytes must be received and acknowledged and then the data is received with a check of receive overflow before data is overwritten into the DSP_rx register.

If the master is requesting a read operation then the slave must start sending data on the I2C_SDA bus immediately after acknowledging the Slave address and RW bit. After each byte the acknowledge from the master must be checked. When the master has received the last byte it will not send an acknowledge and the slave will return to state IDLE. As long as the slave continues transmitting it must first check whether the DSP_tx register has been fully transmitted and if so request more data. The underflow condition is also checked.

Data is now transmitted in byte format. Each data transfer has to contain 8 bits. The number of bytes transferred per transfer is unlimited. Data is transferred with the Most Significant Bit (MSB) first. If a receiver can't receive another complete byte of data until it has performed some other function, it can hold the clock line, I2C_SCL LOW to force the transmitter into a wait state. Data transfer only continues when the receiver is ready for another byte and releases the clock line.

If a slave receiver doesn't acknowledge the slave address (e.g. it is unable to receive because it is performing some real time function) the data line must be left HIGH by the slave. The Master can then abort the transfer.

A LOW to HIGH transition on the I2C_SDA line while the I2C_SCL line is HIGH is defined as a STOP condition. Each data transfer must be terminated by the generation of a STOP condition.

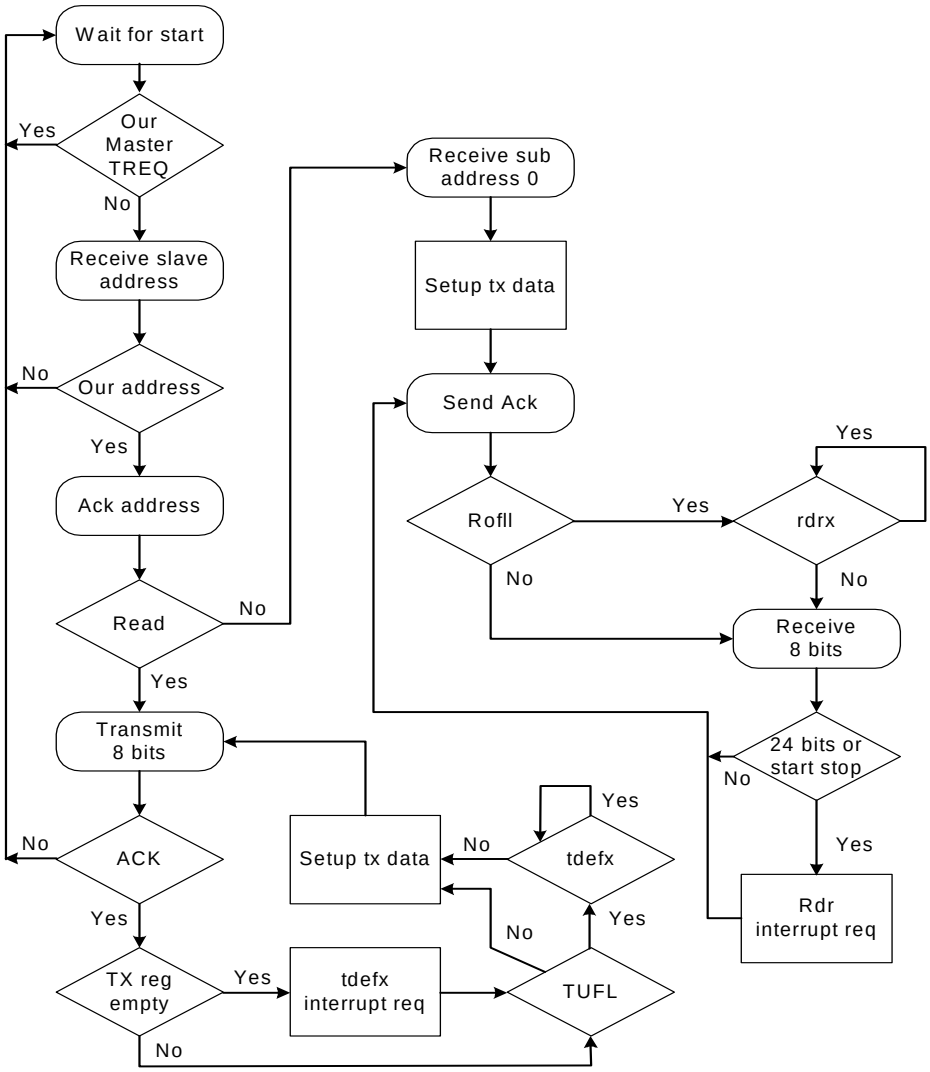


Figure 74. I²C Slave Mode Flow Chart

Tables 254 - 257 show the first 2 bytes of data as a sub-address for purposes of illustration. The sub-address is used to address the memory space inside the device. Table 258 defines each sub-address shown.

ST	SAD+W	SAK	SUB	SAK	SUB	SAK	DATA	SAK	SP

Table 254. I²C transfer when Master is writing one byte of data to a slave

ST	SAD+W	SAK	SUB	SAK	SUB	SAK	DATA	SAK	DATA	SAK	SP

Table 255. I²C transfer when Master is writing multiple bytes to a slave

ST	SAD+W	SAK	SUB	SAK	SUB	SR	SAD+R	SAK	DATA	NMAK	SP

Table 256. I²C transfer when Master is receiving one byte of data from a slave

ST	SAD+W	SAK	SUB	SAK	SUB	SR	SAD+R	SAK	DATA	MAK	DATA	MAK	DATA	NMAK	SP

Table 257. I²C transfer when Master is receiving multiple bytes of data from a slave

BIT	DESCRIPTION
ST	START condition
SR	Repeated START condition
SAD	Slave address
SAK	Slave acknowledge
SUB	Sub address
DATA	Data
SP	STOP condition
MAK	Master Acknowledge
NMAK	No Master Acknowledge

Table 258. I²C Slave and Master mode address definitions

15.2.2. Master Mode Protocol

In Master Mode the I²C interface generates the clock and initiates all transfers.

15.2.2.1. Clock Generation

The I²C clock is generated from the system clock as described above in the register description.

If another device pulls the clock low before the I²C block has counted the high period, then the I²C block immediately pulls the clock low as well and starts counting its the low period. Once the low period has been counted the I²C block releases the clock line high but must then check to see if another device stills holds the line low in which case it enters a high wait-state.

In this way the I2C_SCL clock is generated with its low period determined by the device with the longest clock low period and its high period determined by the one with the shortest clock high period.

15.2.2.2. Master Mode Operation

The finite state machine for master mode operation is shown in Figure 75 and Figure 76. Figure 75 shows the start condition and transmission of the slave address and two sub address bytes. Figure 76 shows the read and write states.

Tables 259 - 262 show examples of Master Mode I²C transactions. Table 258 defines each sub-address shown. The following read after write transactions are performed using the restart technique described in the description of HW_I2CCSR_TREG.

ST	SAD+W	SAK	SUB	SAK	SUB	SAK	DATA	SAK	SP

Table 259. I²C transfer when the interface as master is transmitting one byte of data

ST	SAD+R	SAK	DATA	MAK	DATA	MAK	DATA	NMAK	SP

Table 260. I²C transfer when the interface as master is >1 byte of data from slave

ST	SAD+W	SAK	SUB	SAK	SUB	SAK	SR	SAD+R	SAK	DATA	NMAK	SP

Table 261. I²C transfer when Master is receiving one byte of data from slave internal sub-address

ST	SAD+W	SAK	SUB	SAK	SUB	SAK	SR	SAD+R	SAK	DATA	MAK	DATA	MAK	DATA	NMAK	SP

Table 262. I²C transfer when Master is receiving >1 byte of data from slave internal sub-address

To receive data bytes from non-EEPROM devices that do not require the two sub-address bytes, perform an isolated read without using the read after write restart technique. Make sure that the HW_I2CCSR_ONEBYTE bit is a zero and the following read transactions occur::

ST	SAD+R	SAK	DATA	MAK	SP

Table 263. I²C transfer “FM Tuner” read of one byte.

ST	SAD+R	SAK	DATA	MAK	DATA	MAK	DATA	NMAK	SP

Table 264. I²C transfer “FM Tuner” read of three bytes.

15.2.2.3. Special One Byte Master Mode Transmission

There is a special “one byte mode” which is enabled when the HW_I2CCSR_ONEBYTE bit equals one. In this mode, a start condition is transmitted, followed by the device address byte followed by a single byte of write data. This sequence always ends with a stop condition.

ST	SAD+W	SAK	DATA	SAK	SP

Table 265. I²C transfer when the interface as master is transmitting special ONEBYTE mode

The following C code is used to send a one byte transmission:

```
// Power up the I2C bus pins
HW_GP0PWR.B.B16 = 1;
HW_GP0PWR.B.B17 = 1;

// Load the data register with 1 byte of address[23:16] = 0xC0,
// 1 byte data [15:8] = 0x55 and 1 byte of don’t care [7:0]
HW_I2CDAT.U = 0xC05500;

// Enable I2C
HW_I2CCSR.B.I2C_EN = 1;
```

```
// Set word length to 16 bits (1 addr byte & 1 data byte)
HW_I2CCSR.B.WL = 2;

// Set the last word bit
HW_I2CCSR.B.LWORD = 1;

// Enable special one byte functionality
HW_I2CCSR.B.ONEBYTE = 1;

// Start the transaction
HW_I2CCSR.B.TREQ = 1;
HW_I2CCSR.B.TREQ = 0;

// Wait for busy to clear
while(HW_I2CCSR.B.BUSY == 1);

// a frame with one byte of address and one byte of data was just sent
```

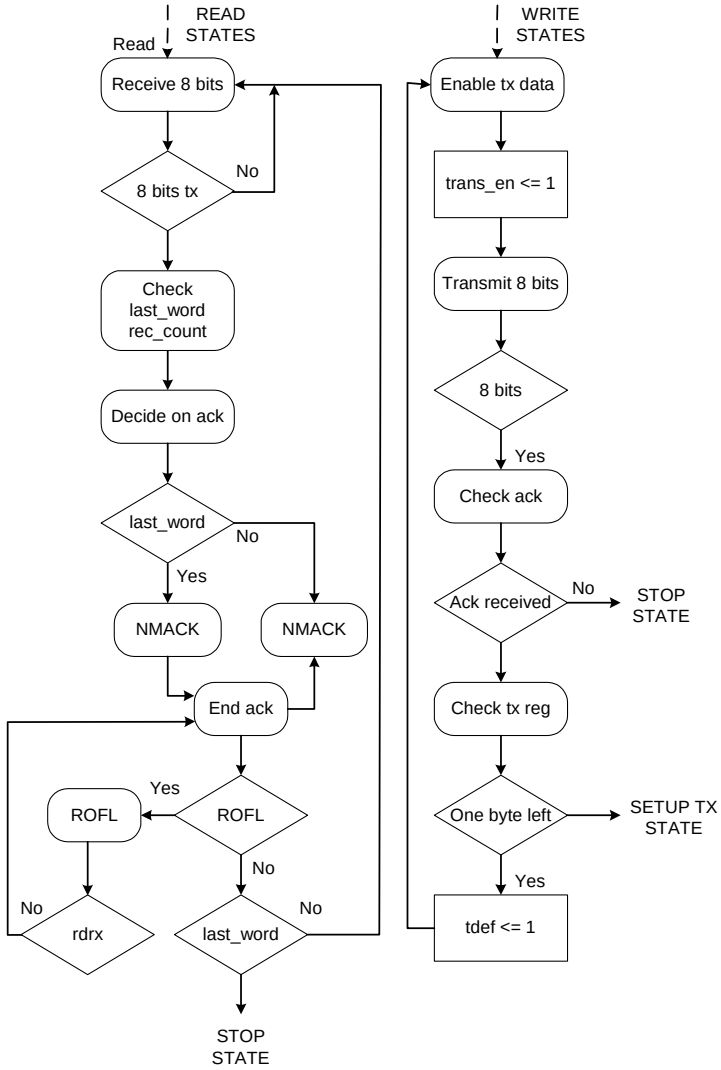


Figure 76. I²C Master Mode Flow Chart – Read and Write States

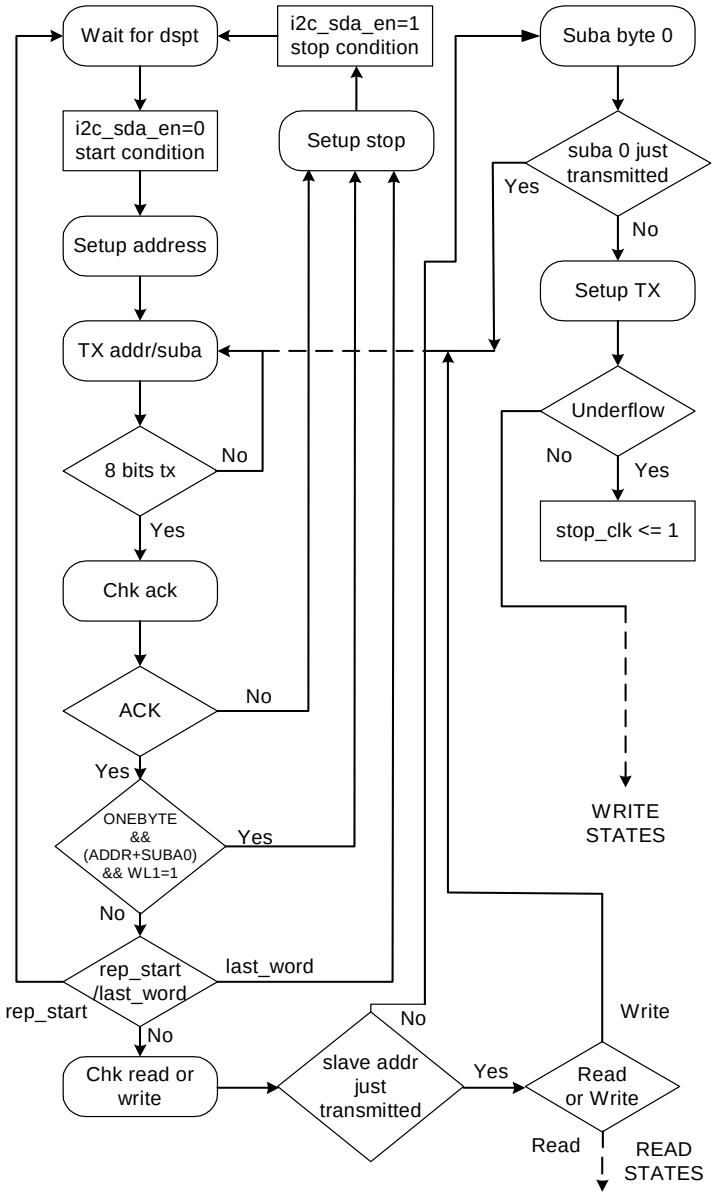


Figure 75. I²C Master Mode Flow Chart

Start condition and transmission of slave address and two sub address bytes or special device address plus one byte mode (HW_I2CCSR_ONEBYTE =1).

16. ENHANCED SPI INTERFACE

The SPI is a standard 4-pin Serial Peripheral Interface for inter-IC control and communication. It interfaces on one side to the SPI bus and on the other has a standard register data and interrupt interface. Although an SPI system can be configured as a master or a slave device, the STMP35xx only supports a master interface via the eSPI.

During an SPI transfer, data is shifted out and shifted in (transmitted and received) simultaneously. The SPI_SCK line synchronizes the shifting and sampling of the information. It is an output when the SPI is configured as a master and an input when the SPI is configured as a slave. Selection of an individual slave SPI device is performed on the slave select line and slave devices that are not selected do not interfere with the SPI buses.

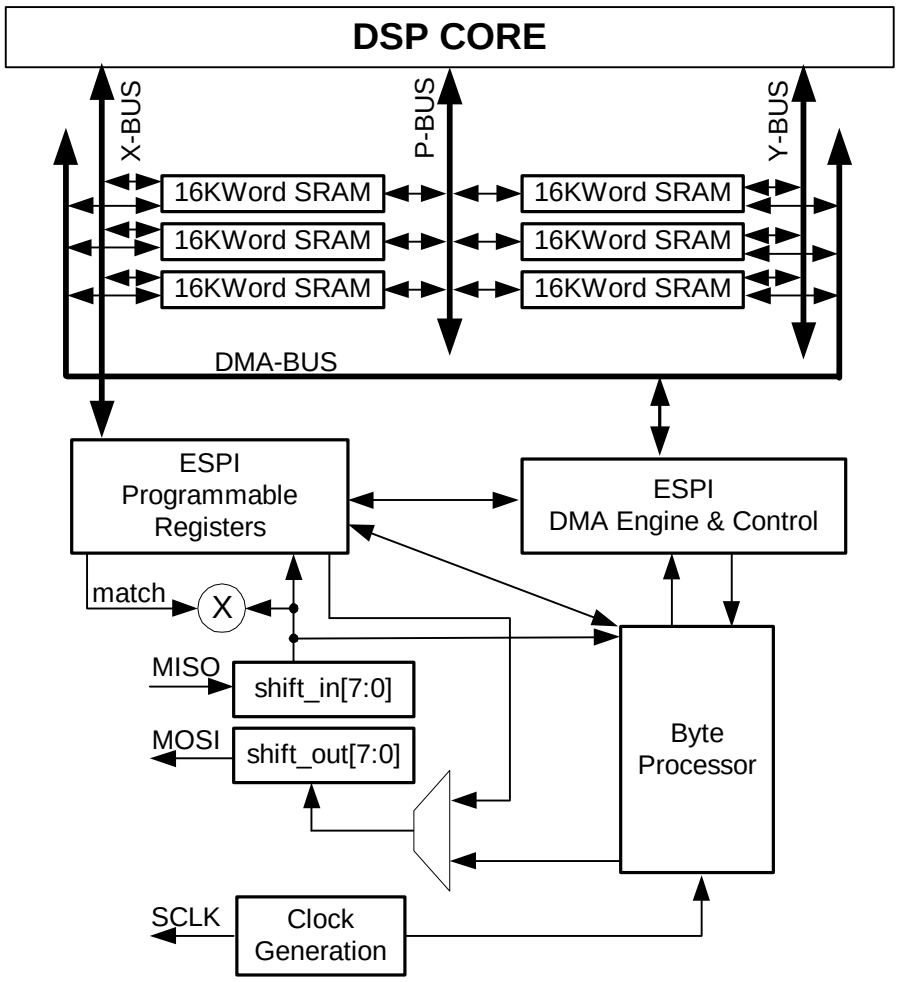


Figure 77. Enhanced SPI Block Diagram

16.1. SPI Pins

SPI_MISO – Master In/Slave Out. Serial input in master mode and output in slave mode.

SPI_MOSI – Master Out/Slave In. Serial output in master mode and input in slave mode.

SPI_SCK – Serial Clock. Bit clock output in master mode and input in slave mode.

SPI_SSn – Slave Select. Selects SPI CS0 in master mode, chip select input in slave mode.

In master mode the SPI_SSn pin is controlled via software through the GPIO module. In slave mode the pin is in SPI mode and connects directly to the SPI module.

Note: See Table 499 for SPI pin placements.

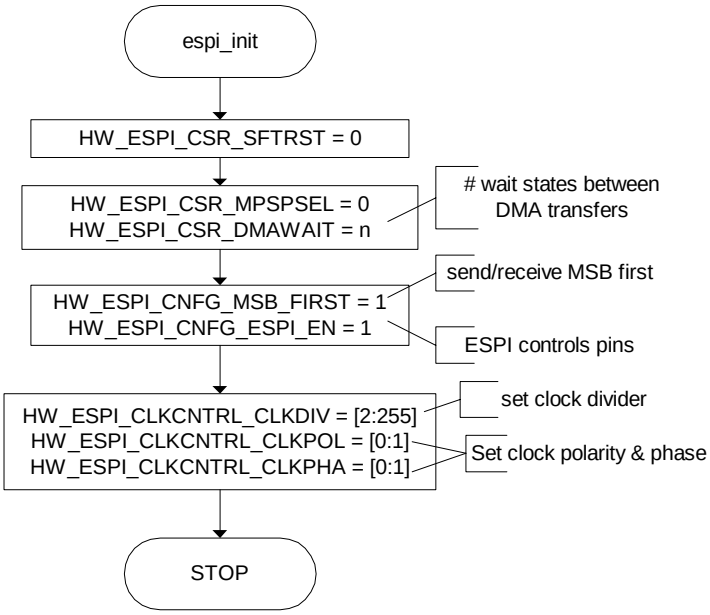


Figure 78. Enhanced SPI Common Initialization Sequence

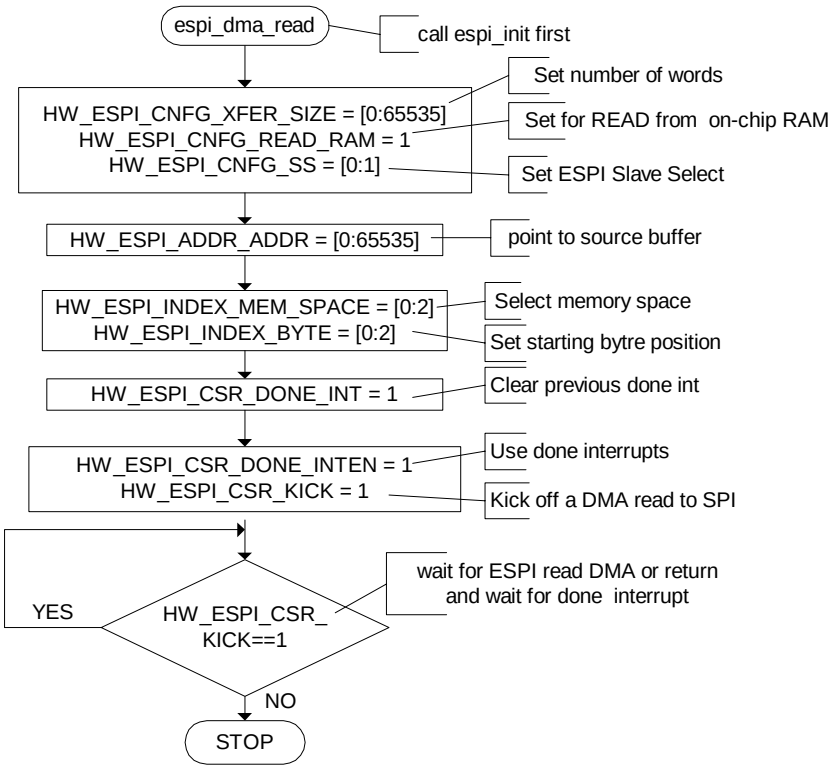


Figure 79. Enhanced SPI DMA Read Flowchart

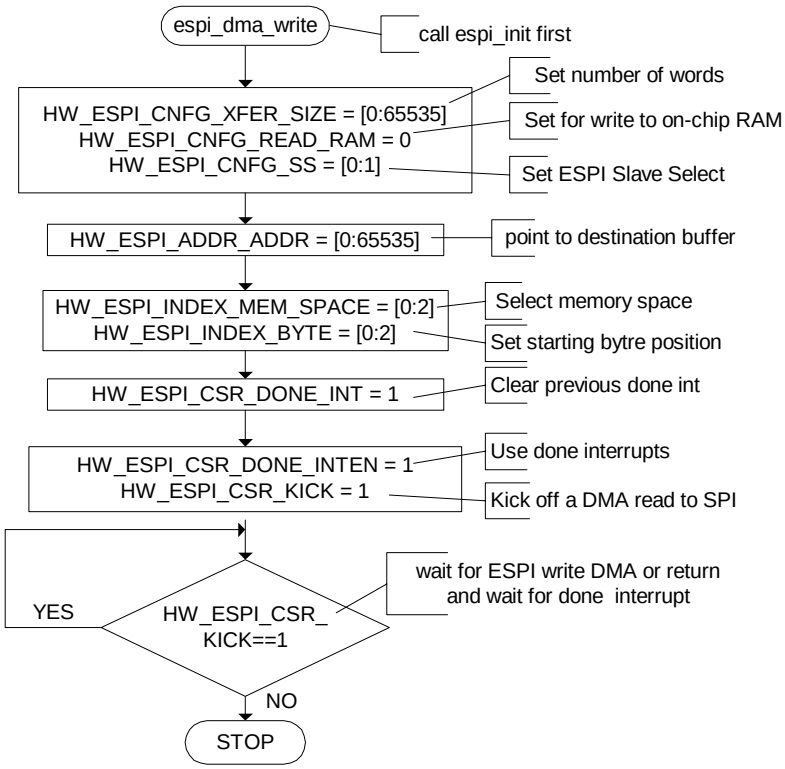


Figure 80. Enhanced SPI DMA Read Flowchart

16.2. Enhanced SPI Programmable Registers

The following registers are available for DSP programmer access and control of the enhanced SPI controller.

16.2.1. ESPI Configuration and Status Register

This register provides overall control and status information for the ESPI, including interrupt control and DMA overhead control.

HW_ESPI_CSR X:$FF00

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SFTRST		MPSPSEL						DMAWAIT						PIOMATCH_INTEN	DONE_INTEN			PIOMATCH_INT	DONE_INT				KICK

Table 266. HW_ESPI_CSR

BITS	LABEL	RW	RESET	DEFINITION
23	**SFTRST**	RW	1	Software Reset
22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21:20	**MPSPSEL**	RW	00	Mode Select bits. In the current design these bits have no effect.
19:16	**RSRVD**	R	0000	**Reserved –** Must be written with 0.
15:12	**DMAWAIT**	RW	0000	Number dclk wait states to insert between DMA requests.
11:10	**RSRVD**	R	00	**Reserved –** Must be written with 0.
9	**PIOMATCH_INTEN**	RW	0	PIO Match Interrupt Enable.
8	**DONE_INTEN**	RW	0	Done Interrupt Enable.
7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5	**PIOMATCH_INT**	RW	0	When SPI_MODE == MATCH and PIO Read Register matches received data then this interrupt status bit will be set to one. Writing a one to this bit will clear it.
4	**DONE_INT**	RW	0	When the SPI interface has completed transfer that was kicked off then this bit is set to one. Writing a one to this sticky bit will clear it.
3:1	**RSRVD**	R	000	**Reserved –** Must be written with 0.
0	**KICK**	RW	0	Set to one to process the command that has been established in the various HW_ESPI registers.

Table 267. ESPI Configuration and Status Register Description

16.2.2. ESPI Configuration Register 1

This register provides configuration and control specific to an ESPI transfer.

HW_ESPI_CNFG X:$FF01

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SPI_MODE				XFER_SIZE																MSB_FIRST	SS	READ_RAM	ESPI_EN

Table 268. HW_ESPI_CNFG

BITS	LABEL	RW	RESET	DEFINITION
23:20	**SPI_MODE**	RW	$0	00 = DMA Mode, ESPI will transmit entire buffer. 01 = PIO Mode, ESPI will transmit each byte written to PIO DATA register. 10 = PIO Match Mode, ESPI continuously transmits byte in PIO register until received data matches PIO_MATCH register. 011,1XX = reserved
19:4	**XFER_SIZE**	RW	$0200	Number of bytes to transfer when the KICK bit is set in DMA or PIO MATCH mode.
3	**MSB_FIRST**	RW	1	Transmit serial data from the Most Significant Bit (MSB) first when this bit is set to one. Transfer LSB first when this bit is set to zero.

Table 269. ESPI Configuration Register Description

BITS	LABEL	RW	RESET	DEFINITION
2	**SS**	RW	0	Slave Select pin is controlled by HW_ESPI_SPICNFG_SS when HW_ESPI_SPICNFG_ESPI_EN is a one.
1	**READ_RAM**	RW	0	Set to one to “read” from on-chip RAM and write to the slave SPI device. Set to zero to transfer from the external SPI device TO the on-chip RAM buffer. Same directions relative to SPI device apply for the PIO case.
0	**ESPI_EN**	RW	0	Set to one to enable ESPI Control of chip pins instead of older SPI device interface.

Table 269. ESPI Configuration Register Description (Continued)

16.2.3. ESPI Clock Control Register

This register controls the SPI clock signal frequency, phase and idle polarity.

HW_ESPI_CLKCNTRL X:$FF02

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
												CLKDIV										CLKPOL	CLKPHA

Table 270. HW_ESPI_CLKCNTRL

BITS	LABEL	RW	RESET	DEFINITION
23:12	**RSRVD**	R	00	**Reserved –** Must be written with 0.
11:4	**CLKDIV**	RW	$02	DCLK divider for SPI-clock. Maximum frequency is dclk/2, minimum frequency is dclk/256. Set to 0 or 1 to disable.
3:2	**RSRVD**	R	00	**Reserved –** Must be written with 0.
1	**CLKPOL**	RW	0	Defines the idle state of the clock. 1: clock idle at logic high voltage 0: clock idle at logic low voltage
0	**CLKPHA**	RW	0	In conjunction with **CLKPOL**, determines the edge on which data is transmitted.

Table 271. SPI Clock Control Register Description

16.2.4. ESPI Programmed I/O Data Register

This register provides PIO read and write access to the serial port.

HW_ESPI_PIODATA X:$FF03

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
OUT								IN								MATCH							

Table 272. HW_ESPI_PIODATA

BITS	LABEL	RW	RESET	DEFINITION
23:16	**OUT**	RW	$00	Data to output on the SPI interface during a PIO transfer or during a PIO_MATCH mode transfer.
15:8	**IN**	R	$00	Received data from the SPI interface can be read here. Valid only while in PIO-Mode and DONE_INT is set to one.
7:0	**MATCH**	RW	$00	Data used to compare against input from slave to generate interrupts.

Table 273. ESPI Programmed I/O Data Register Description

16.2.5. ESPI DMA Address Register

This register holds the DMA starting address.

HW_ESPI_ADDR X:$FF04

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ADDR															

Table 274. HW_ESPI_ADDR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	00	**Reserved –** Must be written with 0.
15:0	**ADDR**	RW	$0000	Address location of the first data word to be transferred by the DMA.

Table 275. ESPI DMA Address Register Description

16.2.6. ESPI DMA Index Register

This register holds the memory space selector and starting byte number for DMA transfers.

HW_ESPI_INDEX X:$FF05

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																		MEM_SPACE				OFFSET	

Table 276. HW_ESPI_INDEX

BITS	LABEL	RW	RESET	DEFINITION
23:6	**RSRVD**	R	$00000	**Reserved –** Must be written with 0.
5:4	**MEM_SPACE**	RW	00	Memory space in which the data-block is located 00 = X-Memory 01 = Y-Memory 10 = P-Memory 11 = reserved
3:2	**RSRVD**	R	00	**Reserved –** Must be written with 0.
1:0	**OFFSET**	RW	00	Byte index into 24-bit on-chip RAM word where the first SPI byte will be written or read. This bitfield provides byte level addressability to DMA transfers targeted at the 24 bit wide on-chip RAM. It specifies the starting byte within the first word addressed by the DMA.

Table 277. ESPI DMA Index Register Description

17. SPI INTERFACE

The SPI is a standard 4-pin Serial Peripheral Interface for inter-IC control and communication. It interfaces on one side to the SPI bus and on the other has a standard register data and interrupt interface. The SPI system can be configured as a master or a slave device.

During an SPI transfer, data is shifted out and shifted in (transmitted and received) simultaneously. The SPI_SCK line synchronizes the shifting and sampling of the information. It is an output when the SPI is configured as a master and an input when the SPI is configured as a slave. Selection of an individual slave SPI device is performed on the slave select line and slave devices that are not selected do not interfere with the SPI buses.

17.1. SPI Pins

SPI_MISO – Master In/Slave Out. Serial input in master mode and output in slave mode.

SPI_MOSI – Master Out/Slave In. Serial output in master mode and input in slave mode.

SPI_SCK – Serial Clock. Bit clock output in master mode and input in slave mode.

SPI_SSn – Slave Select. Selects SPI CS0 in master mode, chip select input in slave mode.

In master mode the SPI_SSn pin is controlled via software through the GPIO module. In slave mode the pin is in SPI mode and connects directly to the SPI module.

Note: See Table 499 for SPI pin placements.

17.2. SPI Registers

17.2.1. SPI Control Register

HW_SPCSR X:$FFF9

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:8	**DIV**	RW	00001010	**Divide factor bits –** These bits are used to control the frequency of the SPI serial clock with relation to the device clock. The number must be an even number. These bits must be set before the SPE or MSTR bits i.e. before the SPI is configured into master mode. **Note:** When the DIV field is written to, if the value written is less than the current value in the internal counter register, then the counter will continue to increment to its maximum value, thereby causing a single very long clock cycle. The new DIV value will then not take effect until the next SPI clock.
7	**MODF**	RW	0	**Mode fault flag –** Mode fault occurs if the SPI system is configured as a master and the SPI_SSn input line goes to active low. This happens if a second SPI device becomes a master and selects this device as if it were a slave. Reading the HW_SPCSR with MODF set automatically clears the MODF flag. **Note:** This condition should never exist since the SPI_SSn line is not used as an input in master mode.
6	**WCOL**	RW	0	**Write Collision Error Flag –** The WCOL Flag is set if the HW_SPDR is written before the end of transfer is signaled. WCOL is cleared by reading the HW_SPCSR with WCOL set, followed by an access of the HW_SPDR.

Table 278. SPI Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
5	**SPIF**	RW	0	**SPI Transfer Complete Flag –** The SPIF Flag is set to a one at the end of an SPI transfer. SPIF is cleared by an access of the HW_SPDR or by a write to the HW_SPDR register. The definition of an end of transfer depends on who is the master and who is the slave.
4	**CPHA**	RW	0	**Clock Phase Select –** **Note:** There must be a delay of 1/2 the clock period of SCLK, after a toggle of CPHA or CPOL, before there is a write to the HW_SPDR. This is to prevent glitches on the SCLK.
3	**CPOL**	RW	0	**Clock Polarity Select –** 0 Active high clocks selected; SCLK idles low. 1 Active low clocks selected; SCLK idles high. **Note:** The SPI_SSn line must be de-asserted and reasserted for a change in the CPOL bit.
2	**SPIE**	RW	0	**SPI interrupts enabled –** When the SPIE bit is set SPI Interrupts are enabled and triggered when the SPIF bit is set.
1	**MSTR**	RW	0	**Master/Slave Select –** When the MSTR bit is set the SPI is configured as a master and when 0 a slave.
0	**SPE**	RW	0	**SPI System Enable –** When SPE is set the SPI system is enabled and, when cleared, disabled.

Table 278. SPI Control Register Description (Continued)

17.2.2. SPI Data Register

The SPI interface runs in PIO mode only. Its 8-bit data register is mapped to the X data address space. Writes to the lower byte of this register are shifted out SPI_MOSI in Master Mode and SPI_MISO in slave mode. This register is read on the transaction completion to retrieve the incoming data from SPI_MISO in master mode and SPI_MOSI in slave mode.

HW_SPDR X:$FFFA

BITS	LABEL	RW	RESET	DEFINITION
23:8	**RSRVD**	R	0	**Reserved –** Must be written with 0.
7:0	**SPIDATA**	RW	00000000	

Table 279. SPI Data Register Description

17.3. Transferring Data over SPI

17.3.1. Master Mode

In master mode the SPI block must generate the SPI_SCK and send appropriate data/commands to the slave device(s).

17.3.1.1. Clock Generation

The clock is the main chip digital clock divided by the divide factor (see above). Out of reset the SPI_SCK is 24.0/10 ~= 2.4 MHz.

The CPOL (clock polarity) and CPHA (clock phase) bits of the HW_SPCSR are used to select any of the four combinations of serial clock. These bits must be the same for both the master and slave SPI devices. The clock polarity bit, selects either an active high or active low clock but does not affect transfer format. The clock phase bit selects the transfer format. See Figure 81.

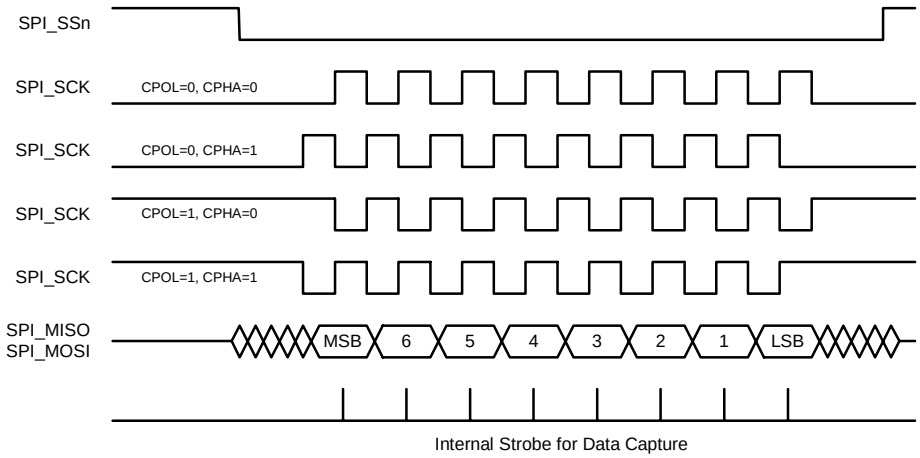


Figure 81. SPI signal timing, including the various SCLK phases

17.3.1.2. Slave Selects in Master Mode

Multiple peripheral slave devices can be accessed by the chip, using GPIO pins as chip selects. The chip select pins must be configured as GPIO outputs by the on-chip DSP. The system code should contain a setup routine that sets each chip select to non-asserted (high). Before any peripheral is accessed its chip select must be asserted (low). The chip select should be de-asserted again before another peripheral is accessed. The SPI_SSn pin should not be in a low state while the SPI is used or a mode fault will occur.

17.3.2. Slave Mode

In slave mode the master device (another chip) generates the clock and slave select for the SPI block. The SPI block will only receive and transmit data on the SPI_MISO and SPI_MOSI lines when the SPI_SSn input is asserted low. In this mode the SPI_SSn line is controlled by the SPI peripheral block.

To initialize the SPI into slave mode, the SPI interrupts are enabled (SPIE=1) and the Master Mode is left off (MSTR=0). The clock polarity is set and finally the system is enabled (SPE=1). The master device should start sending a slave select, clock pulses data. After each byte of data is transferred the SPI will interrupt the processor to receive the incoming data and send a new transmit byte.

18. TIMERS

Four timers are implemented, TIO0-TIO3. Two are connected to off-chip pins (TIO0, TIO1) and the others are used for on-chip functions only. The timers are independently configured through PIO mode registers mapped to the X memory space at base address X:$F100, X:$F140, X:$F180, and X:$F1C0. All timers are identical (except for external connections) and provide the function shown in Figure.

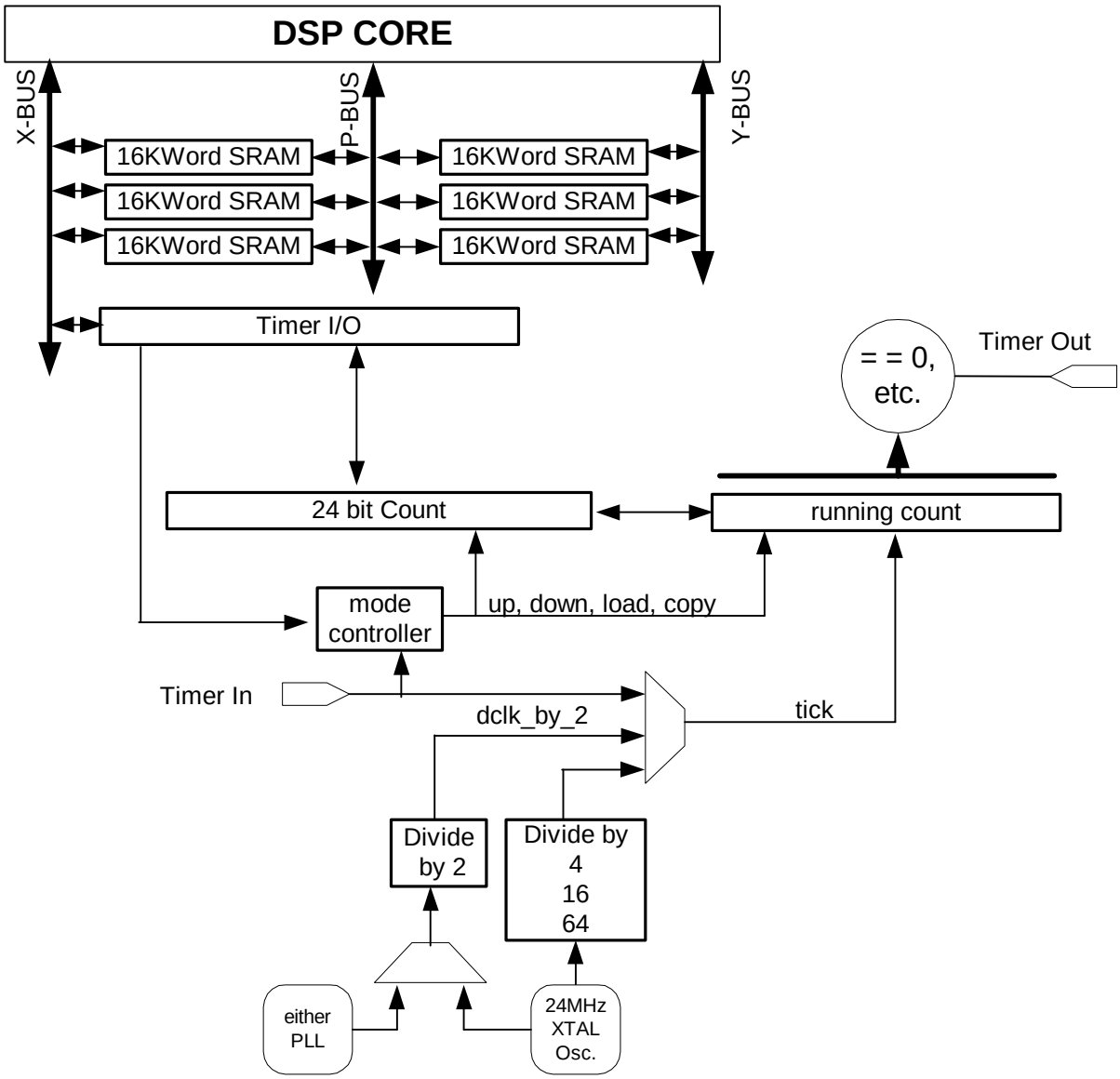


Figure 82. One of Four Timers

18.1. Using the Timer Modules

18.1.1. General Programming Guidelines

If you are re-starting a timer from scratch, all relevant fields of the Timer Count and Timer Control registers should be re-programmed.

The Timer Interrupt Service Routine resets the TimerStatus bit in the Control Register below. The timer will wait until it has expired to assert the interrupt. Otherwise, the Timer Interrupt Service Routine would run continuously.

Throughout the time when an interrupt occurs, the internal, software-invisible running counter within the module, continues to run (otherwise the timer events will not occur at periodic intervals). Thus, if an interrupt is not serviced in a timely fashion, it can become lost - i.e., it merges with another interrupt event before the handler can get to it.

In the default timer clock mode, the timer counts time in terms of DSP clocks. Since the DSP clock frequency can change based on system activity, this means that the delays in the timer will adjust accordingly. To allow the timers to operate independently of the DSP clock speed, use one of the crystal clock modes controlled by the TimerMode field of the Timer Control register. Note, that when using one of the crystal clock modes, the DSP clock must still be running at a minimum of twice the clock source rate for correct operation. This means that the DSP clock must be running at a speed of at least crystal clock divided by 2 (or 12 MHz when using a 24.0 MHz crystal) when in crystal clock/4 mode. Similarly DSP clock must run at a speed of at least crystal clock/8 (3.0 MHz) when in crystal clock/16 mode, and must run at a speed of at least crystal clock/32 (750kHz) when in crystal clock/64 mode.

18.1.2. Software-Visible Programmable I/O (PIO) Register

All register reserved fields should be written with zeroes.

18.1.2.1. Timer Control Register

This is the module's control and status register. Note that the timer can be enabled (TimerEnable bit) and configured in the same PIO write to this register

HW_TMR0CSR X:$F100
HW_TMR1CSR X:$F140
HW_TMR2CSR X:$F180
HW_TMR3CSR X:$F1C0

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CLKGT														TIMER_MODE		TIMER_STATUS		TIMER_CONTROL			INVERT	TIMER_INT_ENABLE	TIMER_ENABLE

Table 280. HW_TMR*CSR

BITS	LABEL	RW	RESET	DEFINITION
23	**CLKGT**	RW	0	**Clock gate –** Used to disable the clocks to the timer module to conserve power when the timer is not in use. Must be set to 0 before writing to any other timer registers. 0 Clocks not gated, i.e. normal operation 1 Clocks are gated, i.e. clocks disabled.
22:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:8	**TIMER_MODE**	RW	00	**Timer clock mode** 00 - Clock source = DSP clock/2 01 - Clock source = crystal clock/4 (requires a DSP clock of at least crystal clock /2) 10 - Clock source = crystal clock/16 (requires a DSP clock of at least crystal clock /8) 11 - Clock source = crystal clock/64 (requires a DSP clock of at least crystal clock /32) **NOTE**: Be sure to set **HW_CCR_XTLEN** to use the timers in crystal clock modes.
7	**TIMER_STATUS**	R	0	**Timer status notification** The TimerStatus bit is set to one in different ways depending on the value in **TIMER_CONTROL** field. See the sections below which describe the conditions that set the **TIMER_STATUS** bit to one. The **TIMER_STATUS** bit is reset to zero as a side effect of reading this register.
6	**RSRVD**	R	0	**Reserved –** Must be written with 0.

Table 281. Timer Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
5:3	**TIMER_CONTROL**	RW	000	**Timer Control** (mode) 000 internal clock, downcount, no output 001 internal clock, downcount, output pulse 010 internal clock, downcount, output toggle 011 (reserved) 100 internal clock, external pulse width measurement 101 internal clock, external period measurement 110 external clock, upcount 111 external clock, downcount
2	**Invert**	RW	0	**Edge/level inversion selection**
1	**TIMER_INT_EN**	RW	0	**Timer interrupt enable** **Note:** 1. The Interrupt must also be enabled in the Interrupt Collector. 2. **HW_CCR XTLEN** must be set to one for some timer modes. 3. **Set HW_TMR*CSR_TIMER_CONTROL** and **HW_TMR*CSR_MODE** before enable interrupts.
0	**TIMER_ENABLE**	RW	0	**Timer enable**

Table 281. Timer Control Register Description (Continued)

18.1.2.2. Timer Count Register

This register is used both to program initial count values and to monitor ongoing counts, depending on the timer mode.

HW_TMR0CNTR X:$F101
HW_TMR1CNTR X:$F141
HW_TMR2CNTR X:$F181
HW_TMR3CNTR X:$F1C1

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
COUNT																							

Table 282. HW_TMR*CNTR

BITS	LABEL	RW	RESET	DEFINITION
23:0	**COUNT**	RW	not reset	Count value expressed as the number of periods specified by the TimerMode field of the **HW_TMR0CSR**/**HW_TMR1CSR**/**HW_TMR2CSR**/**HW_TMR3CSR** registers. **WARNING**: these register are not reset to a known value.

Table 283. Timer Count Register Description

18.1.3. Timer Modes

18.1.3.1. Internal Clock Decrement, No Output Clock

(TimerControl = 000)

With the timer enabled (TimerEnable = 1), the running counter is loaded with the value contained in the Count register. The running counter is decremented every two system clock cycles (dclk/2). During the dclk cycle following the cycle when the run-

ning counter decrements to zero, the TimerStatus bit is set and the module asserts an interrupt (tio_interrupt output) if the TimerIntEnable field is one. Upon reaching zero, the running counter is automatically reloaded with the value from the Count register and the process is repeated until the timer is disabled (TimerEnable = 0).

18.1.3.2. Internal Clock Decrement, Output Pulse

(TimerControl = 001)

This mode is a super-set of the Internal Clock, No Output mode (TimerControl = 000). In addition to its functionality, this mode causes the TIO pin to be pulsed by the timer. The duration of the pulse is two dclk cycles. The pulse begins in the dclk cycle after the running counter has reached zero. The Invert bit determines the polarity of the output pulse. If Invert is zero, the TIO pulses high for two cycles; if Invert is one, TIO pulses low for two cycles.

18.1.3.3. Internal Clock, Output Toggle

(TimerControl = 010)

This mode is a super-set of the Internal Clock, No Output mode (TimerControl = 000). In addition to its functionality, this mode causes the TIO pin to be toggled by the timer. When the chip is reset or when the timer is disabled (TimerEnable zero), the state of the TIO output is identical to the Invert programming field. A toggle of the TIO output occurs off the dclk rising edge after the running counter has reached zero.

18.1.3.4. External Pulse Width Measurement

(TimerControl = 100)

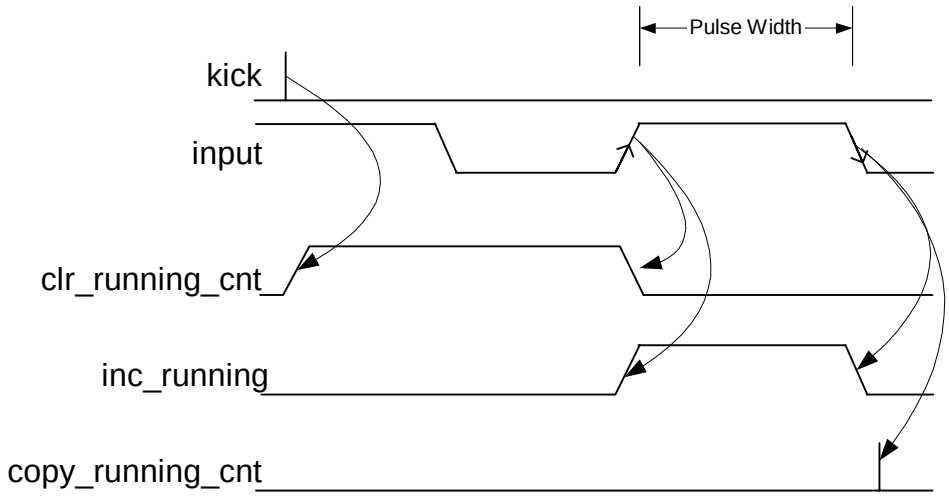


Figure 83. Pulse Width Measurement Mode

In this mode the timer samples an external event/clock on the TIO pin. The internal sampling clock runs at half the internal clock frequency, or dclk/2. This mode mea-

sures the duration of high or low pulses on TIO, in terms of dclk/2 periods. If the Invert bit in the TimerControl register is zero, a TIO-high pulse is measured. If the Invert bit is set, TIO's low phase is measured. When the timer is enabled, the running counter is held at zero until a leading timing edge is detected on TIO. Thus if the timer is enabled in the during TIO's active phase, no counting begins (this eliminates start-up run counts). At the count-terminating edge of TIO, the running count value is transferred to the Count PIO register, the TimerStatus bit is set, the running count register internal to the timer is cleared and an interrupt is optionally asserted (if the interrupt enable is set). The running count register re-starts on the next leading edge of TIO.

Note that this and the following modes should only be used on the externally accessible timers, TIO0 and TIO1.

18.1.3.5. External Period Measurement

(TimerControl = 101)

In this mode the timer samples an external event/clock on the TIO pin. The internal sampling clock runs at half the internal clock frequency, or dclk/2. This mode measures the period TIO, in dclk/2 periods. The user programs an initial count value in the Count PIO register. When the timer is enabled, this count value is transferred to the timer's running count register. The running count register runs continuously, as long as the timer is enabled, overflowing to zero without event. At each significant edge of TIO (rising edge if the Invert bit is zero; falling edge otherwise), the running count is written to the count register, the TimerStatus bit in the Control register is set and the timer will optionally interrupt (if the TimerIntEnable bit is set). The user program can read successive values in the Count register to get a running count of TIO's clock period, in dclk/2 clock cycles. A user program must discard the first count value, since, unlike the pulse width mode, the timer is enabled at an arbitrary point in the TIO clock cycle.

18.1.3.6. External Clock Increment

(TimerControl = 110)

In this mode, the timer counts external clock cycles. When the timer is enabled (TimerEnable = 1), the one's complement of the value in the Count register is transferred to the running count register. The running counter is will increment by transitions on the relevant TIO0 or TIO1 pin. If the Invert bit is zero, the running count register increments on 0-to-1 transitions of the TIO0 or TIO1 pin; otherwise it increments on 1-to-0 transitions. At each update of the running count register, the software-visible Count register is also updated. The running counter counts continuously through its overflow condition (all ones to all zeroes). At the transition after the running counter has reached zero, the TimerStatus bit is set; and if interrupts are enabled, an interrupt is asserted.

18.1.3.7. External Clock Decrement

(TimerControl = 111)

In this mode, the timer counts external clock cycles. When the timer is enabled (TimerEnable = 1), the value in the Count register is transferred to the running count register. The running counter is decremented by transitions on the relevant TIO0 or TIO1 pin. If the Invert bit is zero, the running count register decrements on 0-to-1 transitions of the relevant TIO0 or TIO1 pin; otherwise it decrements on 1-to-0 transitions. On the transition after the running count register has reached zero, the Tim-

erStatus bit is set, the tio_interrupt is asserted (if the TimerIntEnable bit is set) and the running count register is re-loaded with the value from the Count register.

18.1.4. AC Timing Considerations

When a timer module counts off an external clock, its running counter increments every other dclk cycle. Design reuse methodology forces an external clock (TIO) sampling scheme whereby the external clock is sampled into the dclk domain and its edges are then detected. This combination of Nyquist and dclk/2 counting means that the external clock must run at most one fourth as fast as the system clock, or at <=dclk/4 frequency. In practice, the external clock frequency needs to be even lower than this because of the timer interrupt service or polling overhead. For instance if the timer is placed in a mode to measure the pulse width of an external clock or event, the period of such events must be long enough to give the interrupt handler enough time to read the timer's Count register and record the sampled count.

19. SDRAM INTERFACE

The SDRAM interface is a DSP configurable interface to external SDRAM. The interface will connect to various combinations of SDRAMs. The addressing scheme provided by the interface will allow connections to 64 Mbit, 128 Mbit, 256 Mbit, or 512Mbit SDRAMs. The data connection allows only 8-bit transfers. The rate of data transfers and all SDRAM activities is programmable from a divide by one to fifteen of the system clock. The SDRAM timing specifications are also programmable based on this rate. Read or write accesses to SDRAM data can be done through linear or modulo addressing of the system addresses and linear addressing of the SDRAM addresses.

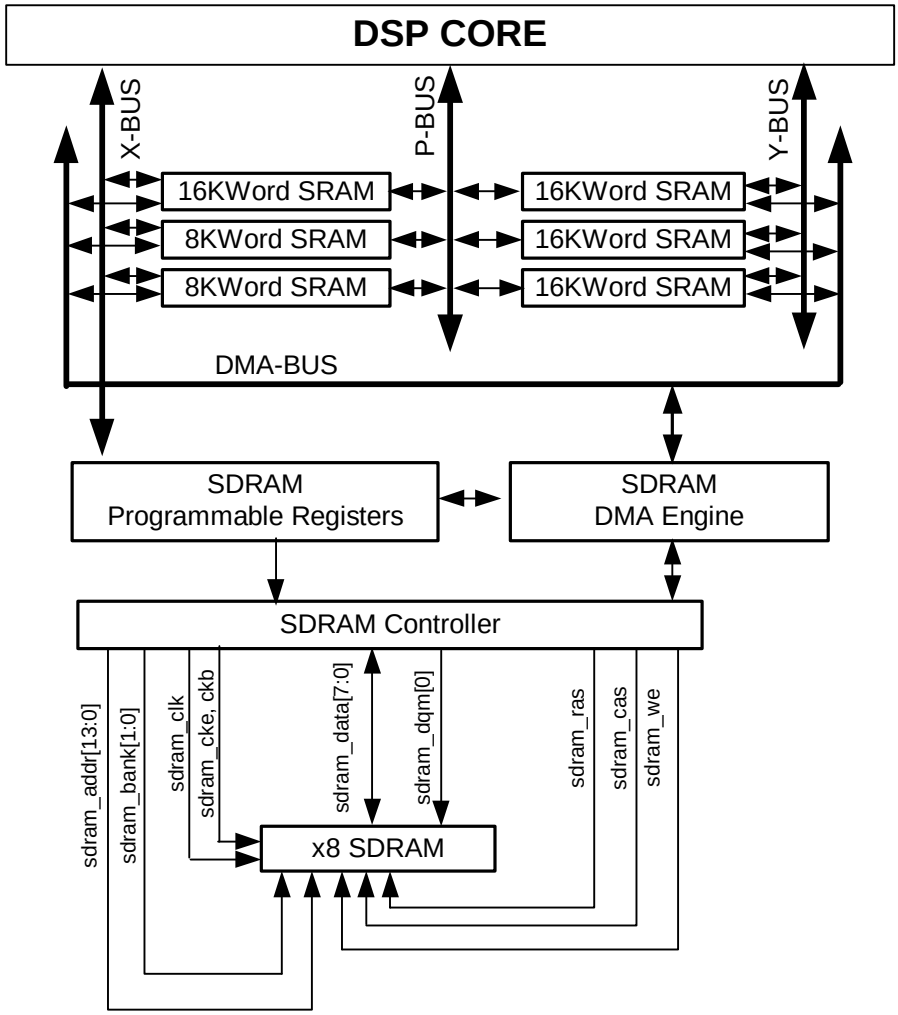


Figure 84. SDRAM Interface

Internally the SDRAM interface will transfer data to/from the system X/Y/P memory (aka system memory) only via DMA and is accessible in a linear or modulo address mode. Data transfers can be programmable to various combinations of data packing and unpacking to/from system memory.

In 8-bit transfers of SDRAM data to/from system memory (24-bit bus), the interface is programmable to pack the data into 3-byte words, 2-byte words, or 1-byte words

during reads from SDRAM and is programmable to unpack with the same capability as during writes to SDRAM. Transferring 2-byte words into system memory can be programmed to be left aligned or right aligned to the 24-bit memory. When left aligned the least significant byte is zeroed and when right aligned the most significant byte can be zeroed or sign extended. Transferring 1-byte words into a 24-bit system memory will result in the data residing in the most significant byte and other two bytes will be zeroed. The interface can be programmed to pick the byte to start transferring data to/from system memory. Writes to SDRAM can be configured to read from the system memory in a Big Endian or Little Endian (default) pattern.

	DESCRIPTION
sdram_clk	Inverted system clock to SDRAM
sdram_cke	SDRAM clock enable to SDRAM
sdram_ras	SDRAM row address strobe
sdram_cas	SDRAM column address strobe
sdram_we	SDRAM write enable
sdram_cs	SDRAM chip select
sdram_addr[13:0]	SDRAM multiplexed column and row addresses
sdram_bank[1:0]	SDRAM bank selection code
sdram_dqm[0]	SDRAM data mask
sdram_data[7:0]	SDRAM data byte for 8 bit interface, lower data byte for 16 bit interface.

Table 284. SDRAM External Signal Pins

The SDRAM controller state machine generates one of six cycle types

- Row Precharge is used to deactivate the open row in a particular bank or the open row in all banks. The SDRAM controller precharges all banks at the same time. The banks will be available after a precharge interval t_{RP}. The controller drives sdram_addr[10] to a one during the precharge command to force all banks to precharge at once. Once a bank is precharge, it is in an idle state and must be activated before it can be read or written.
- Row Activate is used to open (or activate) a row in a particular bank for subsequent read or write access. Sdram_bank[1:0] select the bank to be activated while the sdram_addr[13:11] and sdram_addr[9:0] select the row to be activated or opened. This row remains open for accesses until a precharge command is issued to that bank. A precharge command must be issued before opening a row in the same bank. Since the SDRAM controller works with all banks at once, it precharges all banks then opens the same row in each bank for access.
- Row Refresh causes the SDRAM to read a row from its DRAM array and write it back to make up for any charge leakage in the DRAM cells. This is really an auto refresh command so the address and bank address busses are don't cares for this cycle. It must be issued to each row in the SDRAM within a refresh period. The SDRAM chip keeps track of the address but the controller has to schedule it. Since the controller runs on the system DCLK which is quite variable, users must be careful to program the controller to keep the SDRAM refresh cycles within the manufacturers specifications.
- The Load Mode Register command is issued once after reset to place the SDRAM in the proper CAS latency mode, etc. For the STMP35xx, the value written to the SDRAM mode register is fixed at $0027 which specifies sequential full page burst mode with a CAS latency of two.

- Column Read commands are used by the DMA engine to load data from the SDRAM to the on-chip system RAM. The SDRAM controller reads up to an entire row across all accessed banks with strictly sequential column read commands.
- Column Write commands are used by the DMA engine to copy data from the on-chip system RAM to the SDRAM. The SDRAM controller writes up to an entire row across all accessed banks with strictly sequential column write commands.

The SDRAM controller works on all four banks of an SDRAM chip at a time. It precharges all of them for every transfer and activates all of them for every transfer. It then goes into a page size burst read or write transferring data across all four banks. This is true for 64Mb, 128Mb and 256Mb SDRAM. The following figure shows this effect for the 64Mbit SDRAM case. If two 8 bit wide SDRAMs are attached then the common bank size grows from 2048 byte “pages” to 4096 byte “pages”.

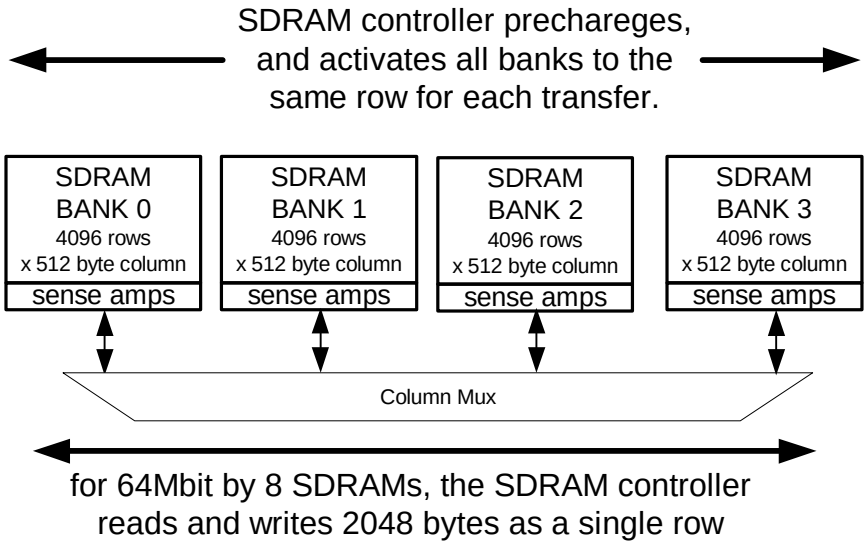


Figure 85. SDRAM accesses all banks as a group

All flip flops int the SDRAM controller are clocked off of DCLK, the same clock used for DMA and DSP functions. The clock sent off-chip to the SDRAM itself can be divided down from the DCLK. In the discussion that follows, we will use sdram_clk for the clock signal sent off chip. Refer to the four bit **DIV** field in Table 286, “SDRAM Control Status Register Description,” on page 225 below. This field selects the number of DCLKs to count for one full sdram_clk cycle. The duty cycle is 50/50 for even cycles and has one additional DCLK in the down phase for odd cycles. All time events in the SDRAM controller are synchronized to the DCLK state where sdram_clk has just gone high. The sdram_clk is inverted as it is driven off chip so that the SDRAM sees a clock that is exactly out of phase with the sdram_clk used within the SDRAM controller. The controller has 16 bit refresh and initialization and a four bit shared delay counter used to guarantee the various timing parameters of and SDRAM. Both of these counters are clocked by DCLK but count the rising edge of the sdram_clk. In the discussion of the various timing register fields below, these two counters are used to implement the required programmable delays for each parameter. All SDRAM specifications give their required timing parameters in terms of nano/micro seconds. All timing parameter fields in the SDRAM controller are given in terms of divided DCLK counts. The DCLK period depends on the settings in the clock control register, Table 35, “Clock Control Register Description,” on page 44. Obviously, if the SDRAM is used with more than one

setting of the DCLK frequency in an application then SDRAM timing parameter bit fields will have to be modified.

Commands given to the SDRAM chips are encoded on the three interface signals sdram_ras, sdram_cas, and sdram_we. On any rising edge of the SDRAM chips clock, it captures these 3 lines and decodes them. When a read command is encoded, the data is available a fixed number of clocks later, depending on how the SDRAM's mode register is set. For the STMP35xx, the mode register is always loaded with a CAS latency of two which tells the SDRAM to return data on the second rising edge after a read command. Many of the timing parameters specified in the SDRAM timing registers set the minimum time between various commands. SDRAM No-op commands are used to fill the gaps and keep all the timing within specification.

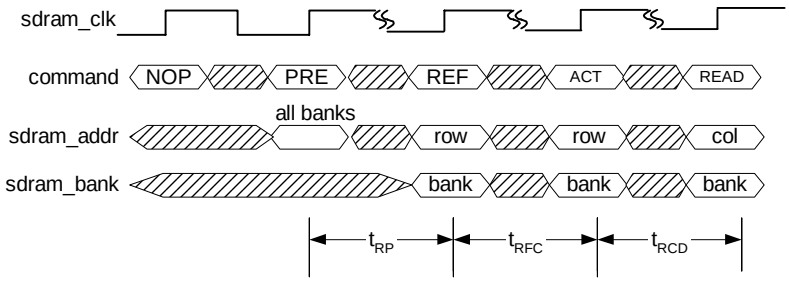


Figure 86. SDRAM Programmable Timing Parameters

19.1. SDRAM Interface Programmable Registers

19.1.1. SDRAM Control Status Register

HW_SDRAM_CSR X:$F900

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SIGN	SDRAM	MULTI	DIV				UNKICK	ASIZE			BIGE	MEM		SBYTE			PWDN	ISAT	LM	KICK	RNW	IE	SDRAMEN

Table 285. HW_SDRAM_CSR

BITS	LABEL	RW	RESET	DEFINITION
23	**SIGN**	RW	0	Sign Extend bit for SDRAM right align reads.
22	**SDRAM**	RW	0	**External bus muxing control –** SDRAM address, control, and data buses are shared with the EMC5600. Setting this signal allows the SDRAM to have control. Default is EMC control.
21	**MULTI**	RW	0	**Multiple external SDRAM mode –** This gives up control of the CSB signal after initialization so that it can be controlled by the GPIO's when there is multiple SDRAMs.
20:17	**DIV**	RW	0001	**Clock Divide –** Allows clock to SDRAM to be divisible from 1 to 15 of system clock.

Table 286. SDRAM Control Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
16	**UKICK**	RW	0	**Unkick –** Writing a one will cleanly stop the current transfer. A transfer halted by Unkick will not interrupt the DSP. After the current transfer stops successfully Unkick will clear.
15:13	**ASIZE**	RW	0	**Align and Connection size –** Controls how bytes or 2-byte words are read or written into system memory. During write to the system memory the byte data can packed onto the 24-bit bus as 3 bytes or 2 bytes or 1 byte. Packing 3 bytes will default starting at the least significant byte. Packing 2 bytes can have two options: left aligned or right aligned. Left aligned will result in the least significant byte assigned equal to zero. The default starting byte is the middle byte (byte 1). Right aligned will result in the most significant bytes assigned a sign extension of bit[15] of the data if the sign extend bit the HW_SDRAM_CSR is set else the most significant bytes will be zeroed. The default starting byte is the least significant byte (byte 0). Packing 1 byte will place the data at the most significant byte of the bus and the lower two bytes will be zeroed. During read from the system memory, data written to the SDRAM will be taken in the same placement as previously described and the zeroed and sign extended data will not be transferred to the SDRAM. For 2-byte words (or 16-bit connection to SDRAM data bus) there is the two options of left aligned or right aligned. For left aligned write to system memory the data will occupy the two most significant bytes of the bus while the least significant byte will be zeroed. For right aligned write to system memory the data will occupy the two least significant bytes of the bus while the most significant byte will have the option of sign extending bit[15] or zeroing. For reads from system memory the word data transferred to the 16-bit SDRAM data bus will have the same alignment mentioned above. The zeroed and sign extended data will not be transferred to the SDRAM. Bit ASIZE2 will be used to differentiate between 8-bit or 16-bit data connection to SDRAM. ASIZE[2:0] = 000 8-bit connection, pack 3 bytes onto system bus (DEFAULT). ASIZE[2:0] = 001 8-bit connection, pack 2 bytes onto system bus with left alignment. ASIZE[2:0] = 010 8-bit connection, pack 2 bytes onto system bus with right alignment. ASIZE[2:0] = 011 8-bit connection, pack 1 byte onto system bus. ASIZE[2:0] =100 16-bit connection, pack 2 bytes onto system with left alignment. ASIZE[2:0] = 101 16-bit connection, pack 2 bytes onto system with right alignment. Note: ASIZE[2] will differentiate between 8-bit or 16-bit connection.
12	**BIGE**	RW	0	**Big Endian –** Controls data is read from system memory and written to SDRAM memory. In big endian mode the most significant byte is transferred first, followed byte the next most significant bytes. Little Endian is the DEFAULT and the least significant byte is transferred first, followed byte the next least significant byte. Only is valid for 2 or 3 byte transfers.
11:10	**MEM**	RW	00	**Memory select –** Chooses X/Y/P memory to write or read data. MEM = 00 X memory MEM = 01 Y memory MEM = 10 P memory
9:8	**SBYTE**	RW	00	**Start Byte –** Start read or write transfer on bytes 0 or 1 or 2. These control bits are closely tied to the Align and Connection Size (ASIZE) control bits. If ASIZE = 0 start byte can be 0,1,2 If ASIZE = 1 start byte can be 1,2 If ASIZE = 2 start byte can be 0,1 If ASIZE = 3 start byte can 2 If ASIZE = 4 start byte can 1 If ASIZE = 5 start byte can 0

Table 286. SDRAM Control Status Register Description (Continued)

BITS	LABEL	RW	RESET	DEFINITION
7	**RSRVD**	RW	0	Reserved.
6	**PWDN**	RW	0	**Powerdown –** Will set the SDRAM into self refresh mode then shut off clocks to the SDRAM and this interface.
5	**ISTAT**	RW	0	**Interrupt Status –** Reading a one indicates pending interrupt, writing back a one will clear interrupt and writing a zero has no effect.
4	**LM**	RW	0	**Load Mode –** loads value of LOAD MODE register into Mode register of SDRAM. After the SDRAMMODE value is loaded into SDRAM LM will clear.
3	**KICK**	RW	0	**Kick writing one will start transfer –** After successful completion kick will clear and interrupt the DSP.
2	**RNW**	RW	0	**Read not write –** writing one will read data from SDRAM to system memory, writing zero will write data to SDRAM from system memory
1	**IE**	RW	0	**Interrupt Enable**
0	**SDRAMEN**	RW	0	**SDRAM Enable Bit –** The SDRAM bit enables the SDRAM port. This bit must be set after the DIV bits are set and any other SDRAM registers are written to. This allows the divided clocks to the SDRAM to stabilize.

Table 286. SDRAM Control Status Register Description (Continued)

19.1.2. SDRAM Type Register

HW_SDRAM_TYPE configures the interface to handle the SDRAM types. The different SDRAM configuration differ from each other by the number of address bits used during column and row commands. This interface can access 64 Mb, 128 Mb, and 256 Mb SDRAM or larger SDRAMs with 8 bit data busses.

HW_SDRAM_TYPE X:$F90F

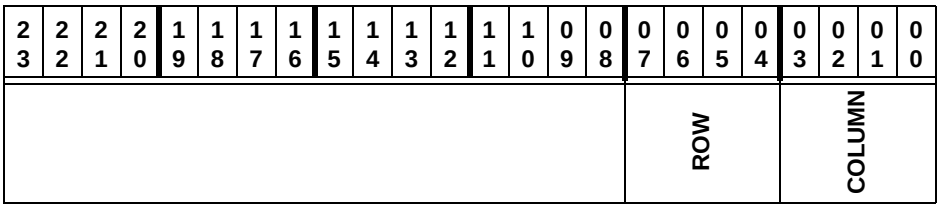


Table 287. HW_SDRAM_TYPE

BITS	LABEL	RW	RESET	DEFINITION
23:6	**RSRVD**	R	0	**Reserved –** Must be written with 0.
5:4	**ROW**	RW	0000	This field specifies the number of address bits in the Row Address 00 - **ROW_WIDTH_11_0** eleven bit row address [11:0] 01 - **ROW_WIDTH_12_0** twelve bit row address [12:0] 10 - **ROW_WIDTH_13_0** thirteen bit row address [13:0] 11 - reserved

Table 288. SDRAM Mode Register Description

BITS	LABEL	RW	RESET	DEFINITION
3	**RSRVD**	RW	g	**Reserved –** Must be written with 0.
2:0	**COLUMN**	RW	0001	This field specifies the number of address bits in the Column Address 0000 - **COL_WIDTH_7_0** eight bit column address [7:0] 0001 - **COL_WIDTH_8_0** nine bit column address [8:0] 0010 - **COL_WIDTH_9_0** ten bit column address [9:0] 0011 - **COL_WIDTH_10_0** eleven bit column address [10:0] 0100 - **COL_WIDTH_11_0** twelve bit column address [11:0] 0101 - **COL_WIDTH_12_0** thirteen bit column address [12:0] 011x - reserved NOTE: column address bit 10 is not used as an address bit in any JEDEC SDRAM.

Table 288. SDRAM Mode Register Description

The SDRAM controller multiplexes the 29 bit SDRAM address onto the 14 bit multiplexed row/column address bus and the 2-bit bank address bus differently depending on the type of SDRAM that is connected to the STMP35xx. For example, a 64Mbit SDRAM has a different number of column address bits than a 16Mbit SDRAM. The controller must adjust the address mapping appropriately. For the 64MBit “By 8” SDRAM there are 9 column address bits. So SDRAM address bits [8:0] are muxed out as column addresses. The controller selects SDRAM address bits [10:9] to drive out the sdram_bank[1:0] signals in this case, finally SDRAM address bits [25:11] are muxed out as row address bits on sdram_addr[13:0]. Thus one selects **ROW_WIDTH_11_0** and **COL_WIDTH_8_0** for a 64Mbit SDRAM.

SDRAM TYPE	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
00,000	unused							**row[11:0]**												**bank**		**column[7:0]**							
00,001	unused							**row[11:0]**												**bank**		**column[8:0]**							
01,001	unused							**row[12:0]**												**bank**		**column[8:0]**							
10,010	unused							**row[13:0]**												**bank**		**column[9:0]**							
10,101	**row[13:0]**																			**bank**		**column[12:0]**							

Table 289. 29 Bit SDRAM Address Mapping to ROW/COLUMN

For a 16Mbit SDRAM, one loads the type register with **ROW_WIDTH_11_0** and **COL_WIDTH_7_0**, as shown in the table above. The largest SDRAM that could be supported by the STMP35xx is 4Gbit (2^29 addressable locations, 8 bits wide). One obtains this mode by setting the type register to **ROW_WIDTH_13_0** and **COL_WIDTH_12_0**.

Complicating all this muxing is the JDEC decision to reserve column address bit 10 to indicate auto precharge for the last cycle on a page. This leaves a hole in the mux patterns as seen on the external pins. A thirteen bit column address (COL_WIDTH_12_0) uses all fourteen bits of the multiplexed row/column address, skipping over bit ten on the multiplexed address bus.

To set these fields, look at the SDRAM you have selected and determine the number of bits in its column address and row address and set the type register field based on these values. Note that some configurations may define more address bits than are actually used in the ROW address.

To connect a 128Mbit “By 8” Micron MT48LC16M8A2 SDRAM to the STMP35xx, use **ROW_WIDTH_11_0** and **COL_WIDTH_9_0**. To connect a 64 Mbit “By 8” Micron MT48LC8M8A2 to the STMP35xx, use **ROW_WIDTH_11_0** and

COL_WIDTH_8_0. NOTE: the STMP35xx only supports SDRAMs with exactly four banks, thus two bank SDRAMS such as the 16Mbit “By 8” Micron MT48LC2M8A2 are not supported.

19.1.3. SDRAM Address Pointer 1 Register

This register holds the lower 24 bits of the 29 bit SDRAM address. The column address bits which are multiplexed out during CAS cycles always come from this register. Similarly, the bank address bits always come from this register. The interface will select the appropriate portion of the row address from this register during RAS cycles. At the end of a transfer, an incremented address will be placed in this register.

HW_SDRAM_ADDR1 X:$F901

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ADDR_LOW																							

Table 290. HW_SDRAM_ADDR1

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ADDR_LOW**	RW	0	Bits 23 through 0 of the 29 bit SDRAM address.

Table 291. SDRAM Address Pointer 1 Register Description

19.1.4. SDRAM Address Pointer 2 Register

This register holds the upper 5 bits of the 29 bit SDRAM address. The interface will select the appropriate portion of the row address from this register during RAS cycles. At the end of a transfer, an incremented address will be placed in this register.

HW_SDRAM_ADDR2 X:$F902

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
																				ADDR_HIGH			

Table 292. HW_SDRAM_ADDR2

BITS	LABEL	RW	RESET	DEFINITION
23:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**ADDR_HIGH**	RW	$00	Bits 28 through 24 of the 29 bit SDRAM address.

Table 293. SDRAM Address Pointer 2 Register Description

19.1.5. SDRAM System Address Pointer Register

SDRAM system address contains the system memory address. This is used with the MEM field of the HW_SDRAM_CSR register to choose reading/writing from X/Y/P memories. At the end of a transfer, the next address will be placed in this register.

HW_SDRAM_SYSADDR X:$F903

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								ADDR															

Table 294. HW_SDRAM_SYSADDR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**ADDR**	RW	0	DMA transfer address in on-chip SRAM.

Table 295. SDRAM System Address Pointer Register1 Description

19.1.6. SDRAM Size Register

SDRAM size register contains the number of transfers to/from system memory from/to the SDRAM.

HW_SDRAM_SIZE X:$F904

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
						SIZE																	

Table 296. HW_SDRAM_SIZE

BITS	LABEL	RW	RESET	DEFINITION
23:18	**RSRVD**	R	0	**Reserved –** Must be written with 0.
17:0	**SIZE**	RW	0	Number of transfers to make, i.e. number of bytes.

Table 297. SDRAM Size Register Description

19.1.7. SDRAM Timer 1 Register

SDRAM timer register programs the delays as specified in the SDRAM specifications in relation to the number of cycles of SDRAM clocks. SDRAM clock is a divided version of system clock. HW_SDRAM_TIMER1 programs the number of SDRAM clock cycles for the initialization delay for SDRAM, delay from a precharge command to the next command (tRP), and delay from refresh command to the next command (tRFC).

HW_SDRAM_TIMER1 X:$F905

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TRFC				TRP				INIT															

Table 298. HW_SDRAM_TIMER1

BITS	LABEL	RW	RESET	DEFINITION
23:20	**TRFC**	RW	6/$6	**Refresh to next command –** Value should be greater or equal to 70ns/(SDRAM period).
19:16	**TRP**	RW	2/$2	**Precharge to next command –** Value should be greater or equal to 20ns/(SDRAM period).
15:0	**INIT**	RW	15000/$3A98	**Initialization** of SDRAM from when SDRAM enable is activated in HW_SDRAM_CSR[0]. Value should be greater or equal to 200us/(SDRAM period).

Table 299. SDRAM Timer 1 Register Description

The value in the **INIT** field is used to delay the initialization sequence after the controller is enabled (SDRAMEN bit in the HW_SDRAM_CSR) until all voltages are stabilized and the SDRAM clock has been running long enough to take the SDRAM chip(s) through their initialization sequence. The inti sequence includes precharge all banks to their idle state issuing an autorefresh cycle and then loading the mode register. For most SDRAMs, this is specified as 200us. If we assume for the moment that the DCLK is derived directly from the 24.0MHz crystal oscillator and that the **DIV** field is set to 8 then we can calculate the value as follows:

The period of the sdram_clk is running at 1/8th the DCLK so it is running at 3.0MHz with a period of 333.3ns. In this case a count of 600 is required to comply. Had the sdram_clk been running at the DCLK (41.6ns), we set this field to 200us divided by 41.6ns or 4807.69. We round this value up to 4808 and load it into the **INIT** field.

19.1.8. SDRAM Timer 2 Register

SDRAM timer register programs the delays as specified in the SDRAM specifications in relation to the number of cycles of SDRAM clocks. HW_SDRAM_TIMER2 programs the number of cycles in between the row and bank activate (ACTIVE) command to the next command (tRCD), the number of cycles in between each refresh commands (tREF/4096), and the number of cycles in between exiting low power mode SELF REFRESH to the next ACTIVE command (tXSR).

HW_SDRAM_TIMER2 X:$F906

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
				TRCD				TREF												TXSR			

Table 300. HW_SDRAM_TIMER2

BITS	LABEL	RW	RESET	DEFINITION
23:20	**RSRVD**	R	0	**Reserved –** Must be written with 0.
19:16	**TRCD**	RW	2/$2	**ACTIVE to next command –** Value should be greater or equal to 20ns/(system period).
15:4	**TREF**	RW	375/$117	**Refresh interval –** Within SDRAM spec the refresh interval for 4096 rows is 64ms thus a refresh must be done every 15us (64ms/4096). Thus this value should be less than or equal to 15us/(system period).
3:0	**TXSR**	RW	6/$6	**Exiting self refresh mode to next command –** Value should be greater or equal to 80ns/(system period).

Table 301. SDRAM Timer 2 Register Description

For TRCD, see Figure 86. “SDRAM Programmable Timing Parameters” on page 225.

TXSR is specified from the time the SDRAM chip has been given its clock enable and sdram_clk has been restarted before it can accept a new command. See the SDRAM manufactures specifications for this value.

19.1.9. System Memory Modulo Base Address Register

HW_SDRAM_BAR programs the base address for system modulo access. The companion register is the DSP modulo register HW_SDRAM_MR. Once the system access matches the address created by adding the HW_SDRAM_BAR and HW_SDRAM_MR it returns to the HW_SDRAM_BAR address for the next access.

HW_SDRAM_BAR X:$F907

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								BAR															

Table 302. HW_SDRAM_BAR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**BAR**	RW	0	

Table 303. SDRAM System Memory Modulo Base Address Register Description

19.1.10. System Memory Modulo Register

HW_SDRAM_MR programs the modulo offset for system modulo access. The companion register is the DSP modulo base address register HW_SDRAM_BAR. The offset value is added to the HW_SDRAM_BAR to the determine the limits of the modulo buffer.

HW_SDRAM_MR X:$F908

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MOD															

Table 304. HW_SDRAM_MR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**MOD**	RW	$00FFFF	**Modulo offset 1-FFFE –** If 0000 or FFFF is programmed system access will be linear.

Table 305. SDRAM System Memory Modulo Register Description

19.1.11. SDRAM Transfer Count Register

HW_SDRAM_CNT holds the value of the number of transfers to or from the SDRAM at the end of the transfer. (READ ONLY)

HW_SDRAM_CNT X:$F90D

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								COUNT															

Table 306. HW_SDRAM_CNT

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**COUNT**	R	0	**Number of transfers completed.**

Table 307. SDRAM Transfer Count Register Description

19.1.12. SDRAM Mode Register

HW_SDRAM_MODE holds the value of the Mode register to be programmed into the SDRAM during initialization. This register is to accommodate future SDRAM capabilities but is defaulted to a sequential full page burst mode with a CAS latency of two.

HW_SDRAM_MODE X:$F90E

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
										VALUE													

Table 308. HW_SDRAM_MODE

BITS	LABEL	RW	RESET	DEFINITION
23:14	**RSRVD**	R	0	**Reserved –** Must be written with 0.
13:0	**VALUE**	R	$0027	Holds the value to be place into the Mode Register of the SDRAM. Defaults to sequential full page burst mode with a CAS latency of 2

Table 309. SDRAM Mode Register Description

20. SWIZZLE

The SWIZZLE block is a DSP configurable module that is used to manipulate programmed I/O data or data within memory. One or two immediate words can be programmed into the module for manipulation. The resulting data can be read, after various manipulations have been performed, in the SWIZZLE registers. SWIZZLE features include Endianess, bit reversing, returning only specific bytes or words with the data sign extended or zeroed, barrel shifting left or right, and division by the fixed integer 3.

The SWIZZLE block also can be used to manipulate data within X/Y/P memory. Data can be changed in place or changed then moved to a different memory location.

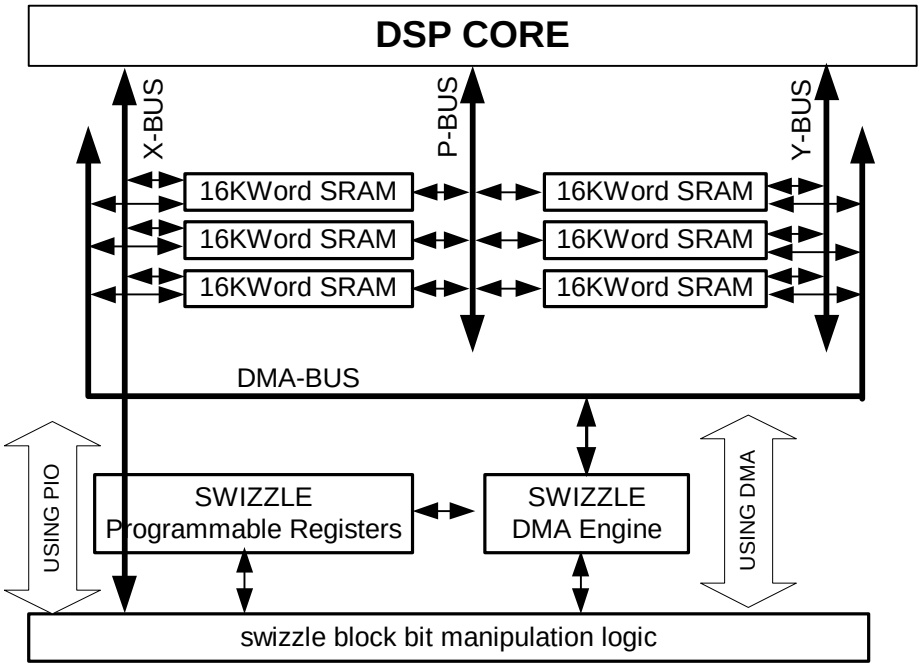


Figure 87. Swizzle PIO & DMA data flows

For DMA operation, a memory to memory move paradigm is used, with a source address pointer, a destination address pointer and a transfer size. Once kicked, the source stream is read, the specified manipulations are performed and the resultant data is written back to on-chip RAM.

In addition to the data manipulations available in previous generations of the SWIZZLE block, a divide by the fixed integer 3 is available. This division is only available as a DSP PIO operation.

20.1. SWIZZLE Registers

20.1.1. SWIZZLE Control and Status Register 1

SWIZZLE CSR1 controls the basic function of the module and the DSP programmable manipulation.

HW_SWIZZLE_CS1R X:$F380

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
													NEWADD	CLK_OFF	MEM	SHIFT				SIGN	LNR	LA	EN

Table 310. HW_SWIZZLE_CS1R

BITS	LABEL	RW	RESET	DEFINITION
23:11	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10	**NEWADD**	RW	0	**Place data into new memory location –** If this bit is cleared, the value in the HW_SWIZZLEDESTADDRR register is ignored. 0 Manipulate data in memory without moving it 1 Manipulate data in memory and put it in a new location
9	**CLK_OFF**	RW	0	**Turn clocks off –** Clocks must be turned on before any other registers can be accessed. 0 SWIZZLE clocks turned on 1 SWIZZLE clocks turned off
8	**MEM**	RW	0	**Manipulate data in memory –** The SWIZZLE block can be used to manipulate data written into the HW_SWIZZLEDATAR0 & HW_SWIZZLEDATA2R registers, or can be used to manipulate blocks of data in on-chip memory. If this bit is clear, then the data in the HW_SWIZZLESOURCER, HW_SWIZZLEDESTADDRR, & HW_SWIZZLESIZER registers are ignored. 0 Manipulate data in registers 1 Manipulate data in memory
7:4	**SHIFT**	RW	0000	**Barrel Shift from 0 to 15 –** This field can be used for 16 bit shifts, however it is recommend that all new programs use the new shift value in **HW_SWIZZLE_SIZER_NEW_SHIFT**.
3	**SIGN**	RW	0	**Sign extend data of pass_lsB,isB,msB modes** 0 Data is not sign-extended 1 Data is sign-extended
2	**LNR**	RW	0	**Left barrel shift –** Used in conjunction with the SHIFT field of this register. 0 Barrel shift to the right 1 Barrel shift to the left
1	**LA**	RW	0	**Left Align data of pass_LSB, ISB, MSB modes –** Default is the data is right aligned with the 24-bit memory bus.
0	**EN**	RW	0	**SWIZZLE enable**

Table 311. SWIZZLE Control and Status Register 1 Description

20.1.2. SWIZZLE Control and Status Register 2

SWIZZLE CSR2 is the rest of the controls to manipulate data in memory.

HW_SWIZZLE_CS2R X:$F381

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								UNKICK	SBYTEDEST		BS_EN	P16I	P16L	PMSB	PISB	PLSB	BITREV	BIGE	DESASEL		SASEL		KICK

Table 312. HW_SWIZZLE_CS2R

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**UNKICK**	RW	0	**Will halt memory swizzling –** Will return to zero after successful halt. Address of the next SWIZZLE will be located in HW_SWIZZLESOURCER and HW_SWIZZLEDESTADDRR
14:13	**SBYTEDEST**	RW	00	**Start Byte of Destination –** When the destination is a different location. SBYTEDEST can be used to place the word starting at an offset byte of the destination. SBYTEDEST = 00 will place the word starting at the LSByte, SBYTEDEST = 01 will place the word starting at the ISByte, and SBYTEDEST = 10 will place the word starting at the MSBYTE of the destination.
12	**BS_EN**	RW	0	**Barrel shift enable –** In memory swizzling mode this used in conjunction CSR1 SHIFT bits.
11	**P16I**	RW	0	**Pass Intermediate Significant word enable –** In memory SWIZZLE mode this will take the input word and only return the most significant and middle byte. Used in conjunction with SIGN bit of CSR1.
10	**P16L**	RW	0	**Pass Least Significant word enable –** In memory SWIZZLE mode this will take the input word and only return the middle and least significant byte. Used in conjunction with SIGN bit of CSR1.
9	**PMSB**	RW	0	**Pass Most Significant byte enable –** In memory SWIZZLE mode this will take the input word and only return the most significant byte. The end location will have this byte in the least significant byte position of the 24-bit word. Used in conjunction with the SIGN bit of the CSR1.
8	**PISB**	RW	0	**Pass Intermediate byte enable –** In memory SWIZZLE mode this will take the input word and only return the middle byte. The end location will have this byte in the least significant byte position of the 24-bit word. Used in conjunction with the SIGN bit of the CSR1.
7	**PLSB**	RW	0	**Pass Least Significant byte enable –** In memory SWIZZLE mode this will take the input word and only return the least significant byte. The end location will have this byte in the least significant byte position of the 24-bit word. Used in conjunction with the SIGN bit of the CSR1.
6	**BITREV**	RW	0	**Bit Reversed enable –** In memory SWIZZLE mode this will take the input word and 24-bit reverse data to the resulting location.
5	**BIGE**	RW	0	**Big Endian enable –** In memory SWIZZLE mode this will take the input word and change to Big Endian word.

Table 313. SWIZZLE Control and Status Register 2 Description

BITS	LABEL	RW	RESET	DEFINITION
4:3	**DESASEL**	RW	00	**Destination Memory select –** Chooses X/Y/P memory to write or read data. 00 X space 01 Y space 10 P space 11 Reserved
2:1	**SASEL**	RW	00	**Source Memory select –** Chooses X/Y/P memory to write or read data. 00 X space 01 Y space 10 P space 11 Reserved
0	**KICK**	RW	0	**Kick bit –** Writing one will start transfer. After successful completion kick will clear.

Table 313. SWIZZLE Control and Status Register 2 Description (Continued)

20.1.3. SWIZZLE Transfer Size

Number of words to be swizzled. This register also contains the new shift amount field.

HW_SWIZZLE_SIZER X:$F382

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			NEW_SHIFT					SIZE															

Table 314. HW_SWIZZLE_SIZER

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	000	**Reserved –** Must be written with 0.
20:16	**NW_SHIFT**	RW	$00	This five bit field can specify shift amounts up to 24 bits. The old shift field in **HW_SWIZZLE_CS1R_SHIFT** is used when this field is set to zero. When **NEW_SHIFT** is non-zero then it over-rides any value in the old shift field.
15:0	**SIZE**	RW	$0000	Number of words of memory to be manipulated by the SWIZZLE module.

Table 315. SWIZZLE Transfer Size Description

20.1.4. SWIZZLE Source Address Register

Source address for memory manipulation mode. Used in conjunction with CSR1 Memory mode enable.

HW_SWIZZLE_SOURCER X:$F383

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**ADD**	RW	$0000	Source address of the data in memory to be manipulated by the SWIZZLE module.

Table 316. SWIZZLE Source Address Register Description

20.1.5. SWIZZLE DATA1 Register

DSP programmed data to be swizzled and placed on output SWIZZLE registers

HW_SWIZZLE_DATA1R X:$F384

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DAT**	RW	$000000	

Table 317. SWIZZLE DATA1 Register Description

20.1.6. SWIZZLE DATA2 Register

DSP programmed data to be swizzled and placed on output SWIZZLE registers. This is used only for resulting data for the HW_SWIZZLEPASSMSWR register.

HW_SWIZZLE_DATA2R X:$F385

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DAT**	RW	$000000	

Table 318. SWIZZLE DATA2 Register Description

20.1.7. SWIZZLE Destination Address Register

SWIZZLE destination address in memory mode.

HW_SWIZZLE_DESTADDRR X:$F386

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSVD**	R	0	Reserved
15:0	**ADR**	RW	$0000	Destination address for memory based SWIZZLE operations.

Table 319. SWIZZLE Destination Address Register Description

20.1.8. SWIZZLE Big Endian Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_BIGENDIANR X:$F387

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DAT**	R		

Table 320. SWIZZLE Big Endian Register Description

20.1.9. SWIZZLE Bit Reversed Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_BITREVR X:$F388

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DAT**	R	$000000	

Table 321. SWIZZLE Bit Reversed Register Description

20.1.10. SWIZZLE Pass Least Significant Byte Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_PASSLSBR X:$F389

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DAT**	R	$000000	

Table 322. SWIZZLE Pass Least Significant Byte Register Description

20.1.11. SWIZZLE Pass Intermediate Significant Byte Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_PASSISBR X:$F38A

BITS	LABEL	RW	RESET	DEFINITION
23:0	DAT	R	$000000	

Table 323. SWIZZLE Pass Intermediate Significant Byte Register Description

20.1.12. SWIZZLE Pass Most Significant Byte Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_PASSMSBR X:$F38B

BITS	LABEL	RW	RESET	DEFINITION
23:0	DAT	R	$000000	

Table 324. SWIZZLE Pass Most Significant Byte Register Description

20.1.13. SWIZZLE Pass Least Significant Word Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_PASSLSWR X:$F38C

BITS	LABEL	RW	RESET	DEFINITION
23:0	DAT	R	$000000	

Table 325. SWIZZLE Pass Least Significant Word Register Description

20.1.14. SWIZZLE Pass Intermediate Significant Word Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R.

HW_SWIZZLE_PASSISWR X:$F38D

BITS	LABEL	RW	RESET	DEFINITION
23:0	DAT	R	$000000	

Table 326. SWIZZLE Pass Intermediate Significant Word Register Description

20.1.15. SWIZZLE Pass Most Significant Word Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with the memmode of the HW_SWIZZLECS1R and the HW_SWIZZLEDATA2R register.

HW_SWIZZLE_PASSMSWR X:$F38E

BITS	LABEL	RW	RESET	DEFINITION
23:0	DAT	R	$000000	

Table 327. SWIZZLE Pass Most Significant Word Register Description

20.1.16. SWIZZLE Barrel Shift Register

Results of data programed into the HW_SWIZZLEDATA1R register. Used in conjunction with SHIFT field of the HW_SWIZZLECS1R register.

HW_SWIZZLE_BARRELR X:$F38F

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DAT**	R	$000000	

Table 328. SWIZZLE Barrel Shift Register Description

20.1.17. SWIZZLE DIV3 Lower Register

Lower 24 bits of swizzle results from divide by three.

HW_SWIZZLE_DIV3L X:$F390

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DIV3_LOWER																							

Table 329. HW_SWIZZLE_DIV3L

BITS	LABEL	RW	RESET	DEFINITION
23:0	**DIV3_LOWER**	R	000	Lower 24 bit result of dividing the 32 bit value in data reg 2 concatenated with data reg 1 by the fixed integer 3.

Table 330. SWIZZLE Transfer Size Description

20.1.18. SWIZZLE DIV3 Upper Register

Remainder and upper 8 bits of swizzle results from divide by three.

HW_SWIZZLE_DIV3U X:$F391

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		REMAINDER														DIV3_UPPER							

Table 331. HW_SWIZZLE_SIZER

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	000	**Reserved –** Must be written with 0.
21:20	**REMAINDER**	RW	$00	Two-bit remainder resulting from dividing the 32 bit value in data reg 2 concatenated with data reg 1 by the fixed integer 3.
19:8	**RSRVD**	R	$000	**Reserved –** Must be written with 0.
7:0	**DIV3_UPPER**	RW	$00	Upper 8 bit result of dividing the 32 bit value in data reg 2 concatenated with data reg 1 by the fixed integer 3.

Table 332. SWIZZLE Transfer Size Description

The dividend for the fixed integer divide by three swizzle operation comes from the concatenation of Data Register 2 with Data Register 1, as shown in Table 333.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DATA_REG_2[7:0]								DATA_REG_1[23:0]																							

Table 333. SWIZZLE 32 bit divide by fixed integer 3

20.2. SWIZZLE Data Manipulation Examples

The following example shows how the SWIZZLE block can be used in a programmed I/O application. Data written to the Swizzle data registers and to the control and status registers causes nine different views of the source data to be readable through the various pass register views, the big endian register view, and the barrel shifter view. Data can also be modified in on-chip RAM in similar ways by setting up the SWIZZLE DMA registers.

	Example 1						Example 2					
Setting these register values												
HW_SWIZZLEDATA1R	A	B	C	D	E	F	A	B	C	D	E	F
HW_SWIZZLEDATA2R	0	1	2	3	4	5	0	1	2	3	4	5
HW_SWIZZLECS1R	0	0	0	0	0	1	0	0	0	0	4	9
Yields these register values												
HW_SWIZZLEBIGENDIANR	E	F	C	D	A	B	E	F	C	B	A	B
HW_SWIZZLEBITREVR	F	7	B	3	D	5	F	7	B	3	D	5
HW_SWIZZLEPASSLSBR	0	0	0	0	E	F	F	F	F	F	E	F
HW_SWIZZLEPASSISBR	0	0	0	0	C	D	F	F	F	F	C	D
HW_SWIZZLEPASSMSBR	0	0	0	0	A	B	F	F	F	F	A	B
HW_SWIZZLEPASSLSWR	0	0	C	D	E	F	F	F	C	D	E	F
HW_SWIZZLEPASSISWR	0	0	A	B	C	D	F	F	A	B	C	D
HW_SWIZZLEPASSMSWR	0	0	4	5	A	B	0	0	4	5	A	B
HW_SWIZZLEBARRELR	A	B	C	D	E	F	F	A	B	C	D	E
	No sign extension						Sign extension on					
	No barrel shift						Barrel shift to the right by 4 bits					

Table 334. SWIZZLE Data Manipulations Examples

21. REAL-TIME CLOCK/ALARM/WATCHDOG RESET & PERSISTENT BITS

These features share a common source of one millisecond time pulses and utilize persistent storage when the chip is in its powered down state. The one-millisecond time base is derived from the 24.0MHz crystal oscillator, even during chip power down states. The crystal oscillator power net is the only circuit powered up during these chip-wide power-down states.

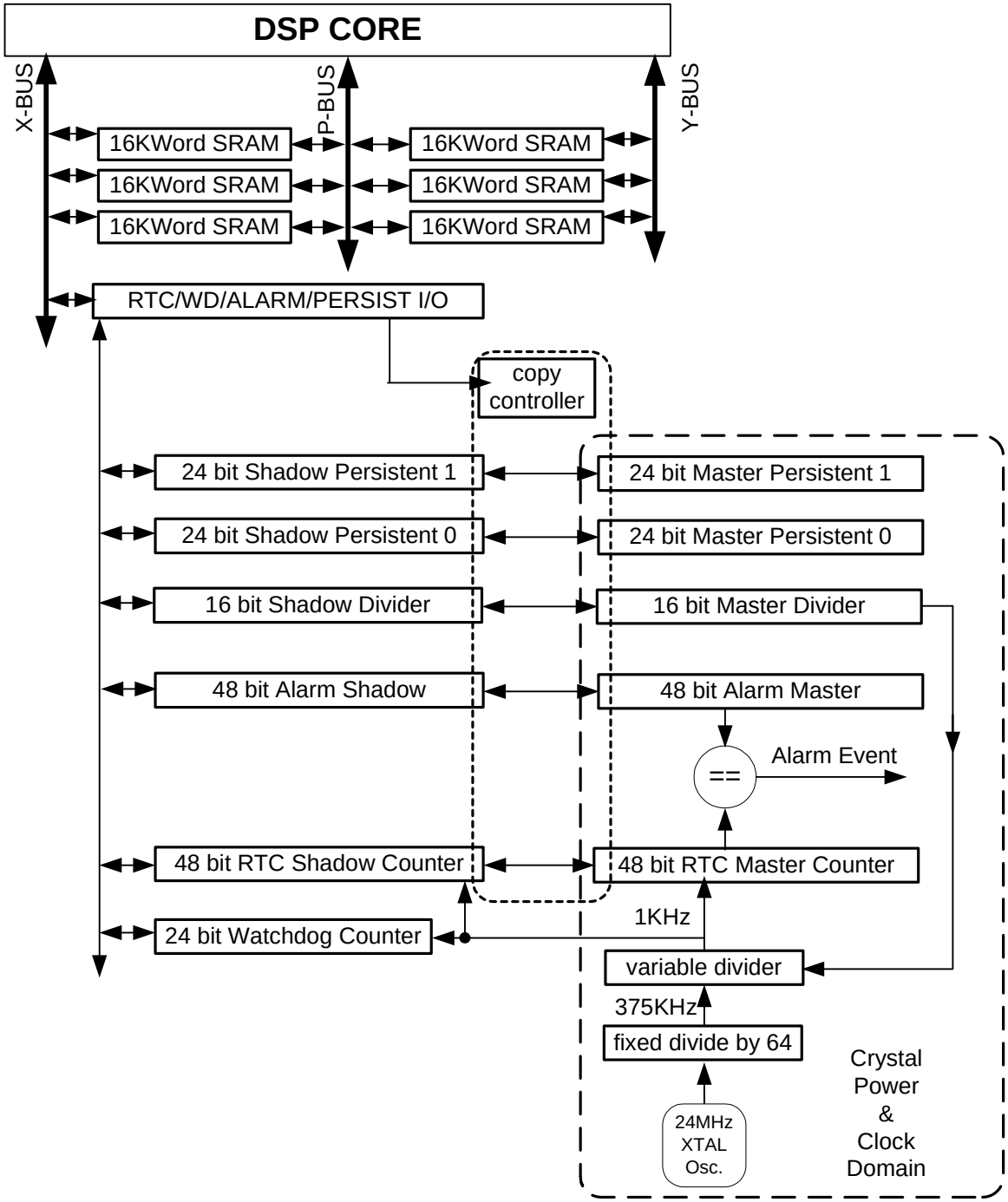


Figure 88. RTC, Watchdog, Alarm, Persistent Bits

As a result, the real time clock count, alarm functions and persistent bit storage are kept in the crystal oscillator clock and power domain. Shadow versions of these values are maintained in the DSP's power and dclk clock domain when the chip is in a power-up state. When the chip transitions from power-off to power-on, the master values are copied to shadow values by the copy controller. Whenever software writes to a shadow register then the copy controller copies the new value into the master register.

Some of the persistent bits are used to control features which can continue to operate after power-down, such as the milli-second counter and the alarm function. Other persistent bits are available to store application state information over power downs. NOTE: the term power-down, as used refers to a state in which the DCDC converter and various parts of the crystal power domain are still powered-up but the rest of the chip is powered-down. If the battery is removed, then the persistent bits, the alarm value and the milli-second counter value will be lost.

Immediately after reset it can take several hundred clocks for the copy controller to complete the copy process from the analog domain to the digital domain. Software can not rely on the contents of the milliseconds counter, alarm or persistent bits until this copy is complete. Therefore, software must wait until all bits in the HW_RTCCSR_STALE_REGS field have been reset to zero by the copy controller before reading the initial state of these values.

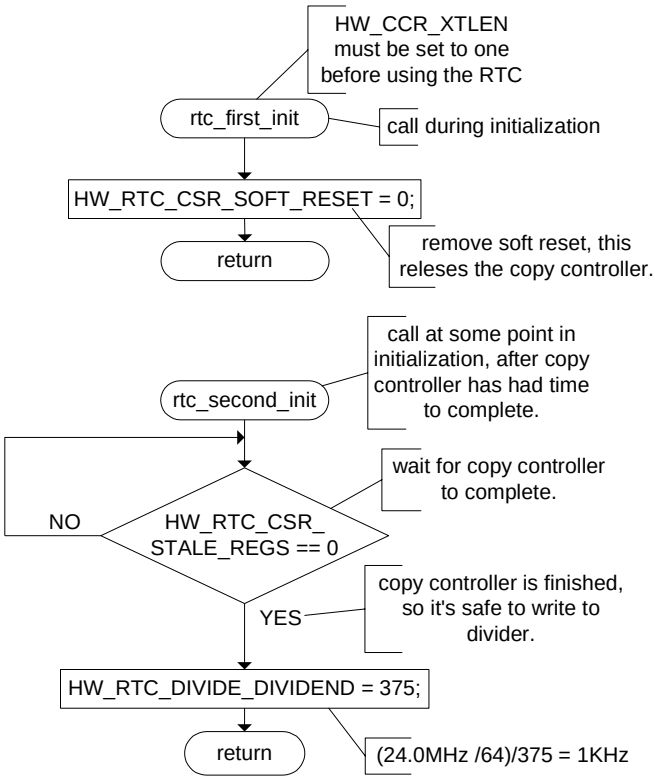


Figure 89. RTC initialization sequence

Before writing to a new value is written to a shadow register by the DSP, software must first confirm that the corresponding bit of HW_RTCCSR_NEW_REGS is a zero. This will insure that a value previously written to the register has been completely handled by the copy state machine. Failure to obey this constraint could cause a newer updated value to be lost.

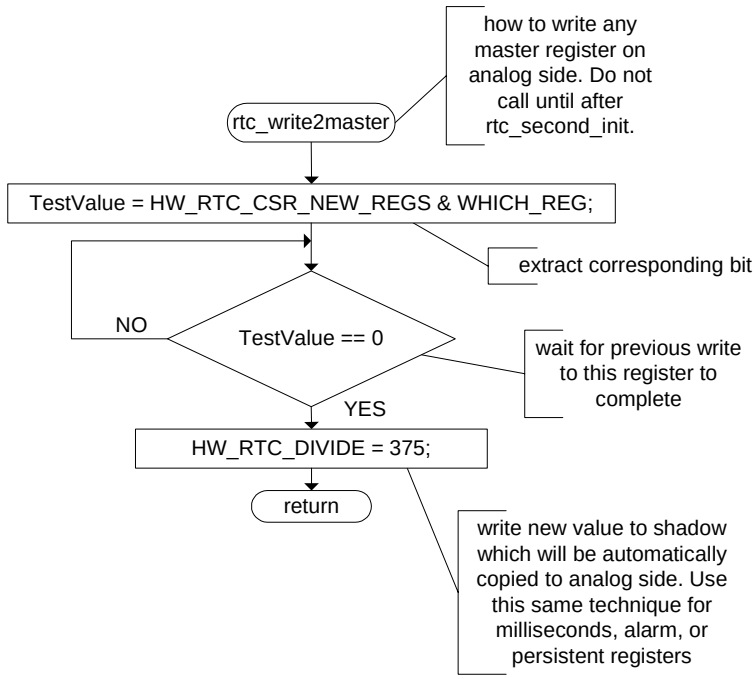


Figure 90. RTC Writing to a Master Register from DSP

21.1. Real Time Clock

The real-time clock is a DSP-accessible, continuously running 48-bit counter that increments every millisecond. A 48-bit millisecond counter has enough resolution to count up to 8896 years with millisecond accuracy. The RTC will continue to count time as long as a voltage is applied to the BATT pin, irrespective of whether the rest of the chip is powered up. Reset has no effect on the RTC registers. This hardware assumes a 24.0 MHz crystal is being used. If a different speed crystal is used, then the times measured by this block should be scaled up or down according to the crystal speed or the crystal oscillator divider should be changed.

Note that the hardware contains a fixed divide by 64 that generates the clock to the RTC. For a 24MHz crystal this produces a 375KHz clock for use in the analog portion of the RTC hardware. A variable divider specified in **HW_RTC_DIVIDE_DIVIDEND** register determines how much to further divide this value to obtain a 1millisecond increment signal for the RTC master milliseconds, shadow milliseconds counter and the watchdog timer. Warning, the default value of this divider register is not correct and must be changed at first power on.

For consistency across applications, it is recommended that the milli-second timer should be referenced to January 1, 1980 at a 48-bit value of zero (same epoch reference as PC) in applications that use it as a time-of-day clock.

The register-set is composed of two 24-bit registers which must be read in a specific order to prevent inconsistencies between the upper and lower words. The upper data word (24-bits) must be accessed before the lower data-word (24-bits).

21.2. Watchdog Reset Registers

The watchdog reset is a DSP configurable device. Programmed by software to generate a chip wide RESET after **HW_RTC_WATCHDOG** milliseconds, the module will generate this reset if software does not re-write this register before this time elapses. The watchdog timer decrements once for every tick of the 1KHz clock generated by **HW_RTC_DIVIDE_DIVIDEND**, see Figure 88. “RTC, Watchdog, Alarm, Persistent Bits” on page 243.

The WATCHDOG TIMER is initially disabled and set to count 16,777,215 milliseconds before generating a watchdog reset.

21.3. Alarm Clock

The alarm clock function allows an application to specify a future instant at which the chip should be “awakened”, i.e. if powered down it can be powered up and the DSP can be interrupted. The alarm clock setting is a DSP-accessible, 48-bit value that is continuously matched against the 48-bit milli-second counter. When the two values are equal, an alarm event is triggered. Persistent bits indicate whether an alarm event should power-up the chip from its powered down state. In addition to or instead of powering up the chip, the alarm event can also cause a DSP interrupt.

21.4. Real Time Clock Programmable Registers

This section describes the programmable registers of the real time clock, including the watch dog registers, alarm registers and persistent registers.

21.4.1. RTC Control and Status Register

Contains the control and status bits for the RTC.

HW_RTC_CSR X:$F500

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SOFT_RESET			FORCED_UPDATE				NEW_REGS					CLKTUNE				STALE_REGS					WATCHDOG_EN	ALARM_INT	ALARM_EN

Table 335. HW_RTC_CSR

BITS	LABEL	RW	RESET 1	RESET 2	DEFINITION
23	**SOFT_RESET**	RW	1	1	Set to zero to remove the soft reset and allow normal operation.
22:21	**RSRVD**	R	000	000	**Reserved –** Must be written with 0.
20	**FORCE_UPDATE**	RW	0	0	Set to one to force an update of all digital shadow registers.
19:17	**RSRVD**	R	000	000	**Reserved –** Must be written with 0.

Table 336. RTC Control and Status Register Description

BITS	LABEL	RW	RESET 1	RESET 2	DEFINITION
16:12	**NEW_REGS**	R	$00	$00	These read only bits indicate that more current data has been written to a shadow register and that a write to the persistent area is pending. 1xxxx = persistent register 1 is stale x1xxx = persistent register 0 is stale xx1xx = alarm count is stale xxx1x = milli-seconds count is stale xxxx1 = crystal divide register is stale
11:9	**CLKTUNE**	R	000	000	Each bit is connected to a delay line tap, setting the MSB bypasses the other two switches, etc. The following skew adjustments are approximate. Contact SigmaTel before setting any of these bits. 000 -2.3ns 001 -729ps 01X -97ps 1XX +831ps
8:4	**STALE_REGS**	R	$1F	$00	These read only bits indicate that more current data is available in the persistent area. An update of the shadow register will be required before accurate data can be obtained. 1xxxx = persistent register 1 is stale x1xxx = persistent register 0 is stale xx1xx = alarm count is stale xxx1x = milli-seconds count is stale xxxx1 = crystal divide register is stale
3	**RSRVD**	R	0	0	**Reserved –** Must be written with 0.
2	**WATCHDOG_EN**	RW			Set to one to enable the watch dog timer function. If the watch dog timer then counts down to zero, a chip wide reset will be issued.
1	**ALARM_INT**	RW	0	0	The alarm interrupt bit is set to one if the alarm goes off on the analog side. Write a one to this bit position to clear the interrupt bit. When this bit is set to one and ALARM_EN is set to one then interrupt 27 is signaled on vector $005E.
0	**ALARM_EN**	RW	0	0	Set this bit to one to enable an alarm interrupt if the alarm “buzzes”.

Table 336. RTC Control and Status Register Description

21.4.2. RTC Milli-Seconds 0 Register

Contains the lower data word (24 bits) of the RTC millisecond count

HW_RTC_MSECONDS0 X:$F501

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RTC																							

Table 337. HW_RTC_MSECONDS0

BITS	LABEL	RW	RESET	DEFINITION
23:0	**RTC**	RW	0	**Bits [23:0] of the RTC millisecond counter –** This field initializes to a random value, and continues to count millisecond as long as the battery power is applied and the crystal oscillator is powered up, regardless of whether the rest of the chip is powered up or down. This register is a shadow of the master counter maintained in the crystal clock/power domain. The master counter continues to count as long as battery power is applied and the crystal oscillator is powered up. When power is restored to the rest of the chip and the chip wide reset is removed, then the master value is copied over to the shadow. At every milli-second tick thereafter, both counters increment by one in their respective clock domains.

Table 338. RTC Milli-Seconds 0 Register Description

21.4.3. RTC Milli-Seconds 1 Register

Contains the upper data word (24 bits) of the RTC millisecond count

HW_RTC_MSECONDS1 X:$F502

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RTC																							

Table 339. HW_RTC_MSECONDS1

BITS	LABEL	RW	RESET	DEFINITION
23:0	**RTC**	RW	0	**Bits [47:24] of the RTC millisecond counter –** This field initializes to a random value, and continues to count millisecond as long as the battery power is applied and the crystal oscillator is powered up, regardless of whether the rest of the chip is powered up or down. This register is a shadow of the master counter maintained in the crystal clock/power domain. The master counter continues to count as long as battery power is applied and the crystal oscillator is powered up. When power is restored to the rest of the chip and the chip wide reset is removed, then the master value is copied over to the shadow. At every milli-second tick thereafter, both counters increment by one in their respective clock domains.

Table 340. RTC Milli-Seconds 1Register Description

21.4.4. Watchdog Reset Count Register

This register contains the 24-bit delay (expressed in milliseconds) after which the watchdog timer will reset the device when the watchdog timer is enabled. The watch dog timer count value is not shadowed. After a power-up it is always reset to $FFFFFF and the enable is set to zero.

HW_RTC_WATCHDOG X:$F503

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
WATCHDOG																							

Table 341. HW_RTC_WATCHDOG

BITS	LABEL	RW	RESET	DEFINITION
23:0	**WATCHDOG**	RW	$FFFFFF	**The number of milliseconds before a watchdog reset is initiated –** This counter only decrements when the watchdog timer is enabled.

Table 342. Watchdog Reset Count Register Description

21.4.5. RTC Divider Register

This register contains the dividend which is used to divide the analog domain clock to obtain a one milli-second “tick” for use in the real time clock milli-second counter and the watch dog timer. The RTC contains a fixed divide by 64 ahead of **HW_RTC_DIVIDE_DIVIDEND**. For a 24.0MHz crystal, this yields a 375KHz clock. Setting **HW_RTC_DIVIDE_DIVIDEND** to decimal 375 ($177) yields a 1KHz tick.

375KHz = 24.0MHz / 64

1KHz = (24.0MHz / 64) / 375

HW_RTC_DIVIDE X:$F506

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								DIVIDEND															

Table 343. HW_RTC_DIVIDE

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**DIVIDEND**	RW	$5DC0	375KHz divider. WARNING: default value is wrong for all applications. Set this to $0177 for a 24.0MHz oscillator.

Table 344. Watchdog Reset Enable Register Description

21.4.6. RTC Alarm 0 Register

Contains the lower data word (24 bits) of the Alarm millisecond value.

HW_RTC_ALARM0 X:$F504

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ALARM0																							

Table 345. HW_RTC_ALARM0

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ALARM0**	RW	0	**Bits [23:0] of the RTC alarm value –** This register is a shadow of the master alarm value maintained in the crystal clock/power domain. The master alarm value retains its value as long as battery power is applied. When power is restored to the rest of the chip and the chip wide reset is removed, then the master alarm value is copied over to the shadow. At every milli-second tick thereafter, the master alarm value is compared against the master milli-second counter. If there is an exact match between all 48 bits of the master milli-second counter and all 48 bits of the master alarm value then the alarm event occurs. When a new value is written to the shadow register it is copied to the master alarm value register. Always write the upper bits (ALARM1) first.

Table 346. RTC Alarm 0 Register Description

21.4.7. RTC Alarm 1 Register

Contains the upper data word (24 bits) of the Alarm millisecond value.

HW_RTC_ALARM1 X:$F505

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ALARM1																							

Table 347. HW_RTC_ALARM1

BITS	LABEL	RW	RESET	DEFINITION
23:0	**ALARM1**	RW	0	**Bits [47:24] of the RTC alarm value –** This register is a shadow of the master alarm value maintained in the crystal clock/power domain. The master alarm value retains its value as long as battery power is applied. When power is restored to the rest of the chip and the chip wide reset is removed, then the master alarm value is copied over to the shadow. At every milli-second tick thereafter, the master alarm value is compared against the master milli-second counter. If there is an exact match between all 48 bits of the master milli-second counter and all 48 bits of the master alarm value then the alarm event occurs. When a new value is written to the shadow register it is copied to the master alarm value register. Always write the upper bits (ALARM1) first.

Table 348. RTC Alarm10 Register Description

21.4.8. RTC Persistent Register 0

This register contains the 24 persistent bits of Persistent Register 0. These bits retain their value through the various power phase of the DCDC converter controller. As long as the battery is not removed, they retain their state.

HW_RTC_PERSIST0 X:$F507

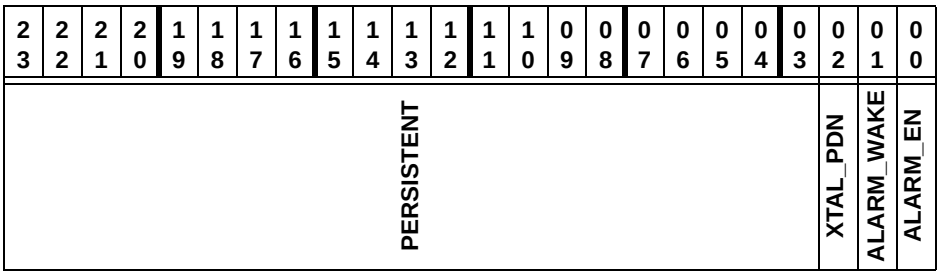


Table 349. HW_RTC_PERSIST0

BITS	LABEL	RW	RESET 1	RESET 2	DEFINITION
23:3	**PERISTENT**	RW	0	0	**Additional Persistent Bits.** These bits retain their value as long as the battery remains connected.
2	**XTAL_PDN**	RW	0	1	**XTAL Power Down.** Default state upon first power-up is to turn off the crystal oscillator during subsequent power down states of the DCDC converters. This bit affects the state of the crystal oscillator when the chip is placed in its various power-down states. During power up modes and entrance to such states the crystal oscillator is automatically restarted. The crystal power down bit, when set to one, removes the last vestige of AC power when the chip is powered down. The power switch and the DC circuits that monitor it are sufficient to restart the chip. The alarm function can not be used when the crystal is powered down. **NOTE: HW_DCDC_PERSIST_SLEEP_XTAL_ENABLE** must also be set to zero put the crystal to sleep.
1	**ALARM_WAKE**	RW	0	0	Set to one to wake up the chip from a power down state upon the arrival of the alarm event. The alarm event must be enabled by **ALARM_EN** and the crystal oscillator must not be in its powered down state.
0	**ALARM_EN**	RW	0	0	Set to one to enable the alarm clock function. The alarm event will occur whether the chip is in the powered-on or powered-off state. In the powered off state, it will wake up the chip and return it to a powered-on state, iff the **ALARM_WAKE** bit is also set to one. The 48 bit alarm register is compared to the 48 bit milli-second counter, if an exact match occurs, and the **ALARM_EN** bit is set to one then an alarm event is triggered.

Table 350. RTC Persistent Register 0 Description

21.4.9. RTC Persistent Register 1

This register contains the 24 persistent bits of Persistent Register 1. These bits retain their value through the various power phases of the DCDC converter controller. As long as the battery is not removed, they retain their state.

HW_RTC_PERSIST1 X:$F508

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PERSISTENT																							

Table 351. HW_RTC_PERSIST1

BITS	LABEL	RW	RESET	DEFINITION
23:1	**RSRVD**	R	0	**Reserved –** Must be written with 0.
0	**PERSISTENT**	RW	0	**Additional Persistent Bits.** These bits retain their value as long as the battery remains connected, even if the chip is in one of its powered down states.

Table 352. RTC Persistent Register 1 Description

22. I²S SERIAL AUDIO INTERFACE

The chip includes a standard 3-channel I²S Serial Audio Interface (SAI). This allows the chip to receive data from an external host or A/D or transmit data to an external D/A. The chip must be configured as slave device on the I²S bus. Data can be transferred between the DSP core and the SAI by polling status flags or by servicing interrupts.

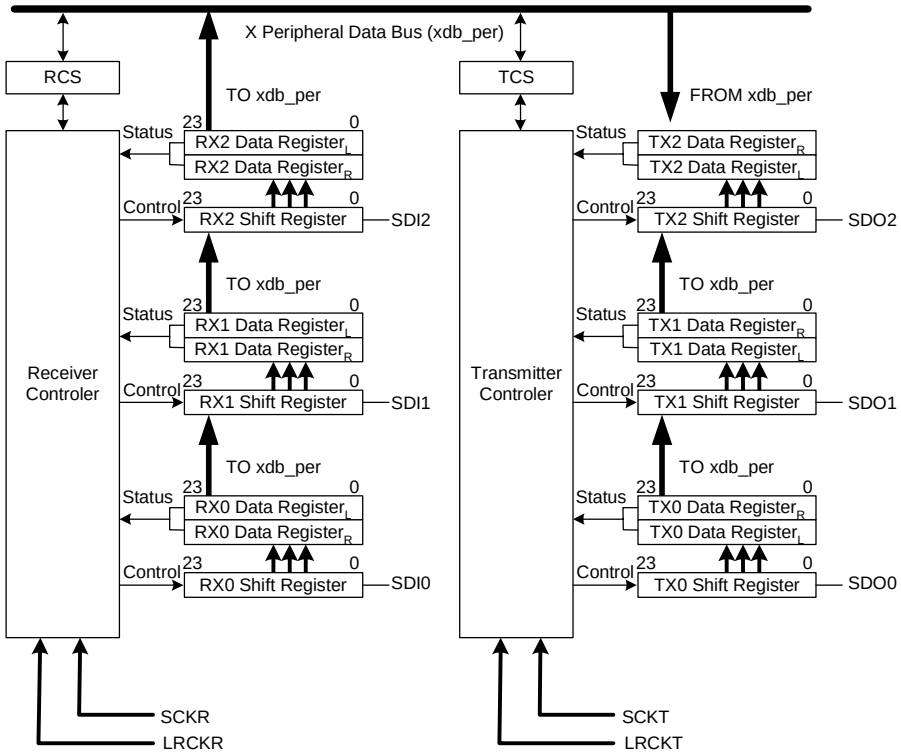


Figure 91. I²S Block Diagram

22.1. I²S External Pins

Name	Description	Pin # I2S_SELECT=0	Pin # I2S_SELECT=1
I2S_BCLK	Input, I²S serial bit clock	91/G3	NA/K3
I2S_WCLK	Input, I²S left/right word clock	92/H3	NA/K1
I2S_DataI0	Input, I²S input data 0	95/J1	NA/J2
I2S_DataI1	Input, I²S input data 1	94/H2	94/H2
I2S_DataI2	Input, I²S input data 2	93/H4	93/H4
I2S_DataO0	Output, I²S output data 0	90/G4	90/G4
I2S_DataO1	Output, I²S output data 1	89/F3	89/F3
I2S_DataO2	Output, I²S output data 2	88/F4	88/F4

Note: The I2S_Select bit of the HW_SPARER register shown on page 47, controls the pinout of these pins.

22.2. I²S Receive and Transmit Registers

22.2.1. Receivers

The SAI contains three 2-channel receivers. These receivers shift serial data from the I2S_DataIx pins using the bit and word clocks. It fills the data register (SAIRX0,1,2R) until the word clock or RCS word length indicate that the word has ended. The next word is placed in the other receiver channel.

After both words have been clocked into the data registers the Read Data Ready (RDR) Flag is set. The DSP should read the data registers and clear the RDR flag before the next word is ready to move from the shift register to the data register. The DSP can either poll the RDR flag or receive an interrupt if the RXIE Bit is set.

Both channels of the read data registers are muxed into the same DSP address. The first read will retrieve the data from the left channel data register. The second will retrieve the right channel data. Subsequent reads will alternately retrieve the left and right channel data.

22.2.1.1. I²S Receive Status Control Register

This register is used both to program initial count values and to monitor ongoing counts, depending on the timer mode.

HW_SAIR_CSR X:$FFF0

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
							ROFCL	RDR	ROFL		RXIE	RDWJ	RREL	RCKP	RLRS	RDIR	RWL			RMME	REN2	REN1	REN0

Table 353. HW_SAIR_CSR

BITS	LABEL	RW	RESET	DEFINITION
23:17	**RSRVD**	R	0	**Reserved –** Must be written with 0.
16	**ROFCL**	RW	0	**Receiver Data Overflow Clear –** When this bit is set by the DSP both the ROFL and RDR flags are cleared. This bit is implemented as a one-shot and always reads a zero. Note: if ROFL is cleared in the last instruction of an Interrupt Service Routine triggered by this bit, then the Interrupt will be executed twice.
15	**RDR**	R	0	**Receiver Data Ready Flag –** Indicates that both left and right data words have been received. This bit is cleared when both left and right data are read from the receiver data registers of all enabled receivers, or by writing a one to the ROFCL bit.
14	**ROFL**	R	0	**Receiver Data Overflow –** Indicates that an overflow condition was detected. This condition occurs when the RDR Flag is set and a new data word is transferred from the shift register to the receiver data register. Writing a one to the ROFCL bit clears this bit.
13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12	**RXIE**	RW	0	**Receiver Interrupt Enable for DSP –** Interrupt enable for DSP.

Table 354. I²S Receive Status Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
11	**RDWJ**	RW	0	**Receiver Data Word Justification –** Determines which portion of the 24-bit portion of the received 32-bit word will be transferred from the shift register to the data register when programmed to receive 32 bit words. 0 The first 24 bits received are transferred to the data register. 1 The last 24 bits received are transferred to the data register. When the receiver is programmed to receive 24 or 16 bit words this bit functions as follows: 0 Means that either the SCKR bit-counter (when it reaches 16 or 24) or a LRCKR transition, whichever comes first, can transfer data to the data registers. 1 Means that a LRCKR transition only will terminate the transfer and the last 16 or 24 bits received before the LRCKR transition gets transferred to the data registers.
10	**RREL**	RW	0	**Receiver Relative Timing –** Determines the relative timing of the LRCKR signal as referred to the serial data inputs. 0 The transition of LRCKR occurs together with the first bit data. 1 The transition of LRCKR occurs one SCKR earlier (I^2S format).
9	**RCKP**	RW	0	**Receiver Clock Polarity –** Defines the polarity of the receiver clock. 1 Positive Clock Polarity. Means the LRCKR and SDI lines change synchronously with the positive edge of the clock, and are considered valid during negative transitions. 0 Negative Clock Polarity. Means the LRCKR and SDI lines change synchronously with the negative edge of the clock, and are considered valid during positive transitions
8	**RLRS**	RW	0	**Receiver Left Right Selection –** Defines the polarity of the LRCKR. 0 If LRCKR = 0 then Left Word, if LRCKR = 1 then Right Word. 1 If LRCKR = 0 then Right Word, if LRCKR = 1 then Left Word.
7	**RDIR**	RW	0	**Receiver Data Shift Direction –** Determines the shift direction of the received data. 0 MSB First 1 LSB First.
6:5	**RWL**	RW	0	**Receiver World Length Control –** Selects the length of the data word being received by the SAI. 00 16 Bits 01 24 Bits 10 32 Bits 11 Reserved.
4	**RSRVD**	R	0	**Reserved –** Must be written with 0.
3	**RMME**	RW	0	**Receiver master mode enable** 1 Receiver in master mode. 0 Receiver in slave mode. Enable Master Mode for the receiver. i.e. Enables use of the Transmitter clocks from the PLL to shift in data. A `1' in this bit configures the receiver into Master Mode.
2	**REN2**	RW	0	**Receiver Enable –** Enable bit for Receiver channel 2.
1	**REN1**	RW	0	**Receiver Enable –** Enable bit for Receiver channel 1.
0	**REN0**	RW	0	**Receiver Enable –** Enable bit for Receiver channel 0.

Table 354. I^2S Receive Status Control Register Description (Continued)

22.2.1.2. I^2S Receive Status DataI0 Register

HW_SAIRX0R X:$FFF1

BITS	LABEL	RW	RESET	DEFINITION
23:0	**RECEIVE0**	RW	0	**Data received from I2S_DATAI0**

Table 355. I^2S Receive Status DataI0 Register Description

22.2.1.3. I^2S Receive Status DataI1 Register

HW_SAIRX1R X:$FFF2

BITS	LABEL	RW	RESET	DEFINITION
23:0	**RECEIVE1**	RW	0	**Data received from I2S_DATAI1**

Table 356. I^2S Receive Status DataI1 Register Description

22.2.1.4. I^2S Receive Status DataI2 Register

HW_SAIRX2R X:$FFF3

BITS	LABEL	RW	RESET	DEFINITION
23:0	**RECEIVE2**	RW	0	**Data received from I2S_DATAI2**

Table 357. I^2S Receive StatusDataI2 Register Description

22.2.1.5. Handling Different Word Lengths

It is possible that an external SAI and the chip's SAI could be programmed to different size word lengths. The normal case is handled by terminating (performing a parallel load of the data register) the transfer when a LRCKR transition occurs. However, when the chip's SAI is programmed to receive 16 bit words and the external SAI is programmed to transmit 24 bit words (for example) the parallel load should occur as soon as the 16 bits are serially shifted in. This implies that a bit counter must be used to terminate the transfer.

Another case would be when the external SAI is programmed to transmit fewer bits than the chip's SAI is programmed to receive. For example, if the external SAI is programmed to transmit 16 bits and the chip's SAI is programmed to receive 24 bits the transfer should be terminated by the LRCKR.

22.2.2. Transmitters

Each I^2S channel has a transmitter. The transmitter serially outputs data from the left/right data registers (SAITX0,1,2R). Each I^2S channel can carry two data streams (i.e. audio channels). The data can be arranged in 16, 24 or 32 bit words. The chip's I^2S transmitters must operate in slave mode (clocks generated by the master). The transmit data must be loaded into the transmit registers by the DSP before the master begins clocking. After the master clock copies the second word into the transmit shift register it sets the Transmit Data Empty (TDE) flag to signal the DSP for more data. The Transmit Status/Control Register (TCS) configures the I^2S transmitters.

The left and right channel transmit registers also share the same DSP memory. The first word write will go to the left channel data register. The second will go to the right. Subsequent rights will alternate between left and right data registers.

22.2.2.1. I^2S Transmit Status Control Register

This register is used both to program initial count values and to monitor ongoing counts, depending on the timer mode.

HW_SAITCSR X:$FFF5

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
							TUFCL	TDE	TUFL		TXIE	TDWE	TREL	TCKP	TLRS	TDIR	TWL			TMME	TEN2	TEN1	TEN0

Table 358. HW_SAITCSR

BITS	LABEL	RW	RESET	DEFINITION
23:17	**RSRVD**	R	0	**Reserved –** Must be written with 0.
16	**TUFCL**	RW	0	**Transmitter Data Underflow Clear –** When this bit is set by the DSP both the TUFL and TDE flags are cleared. This bit is implemented as a one-shot and always reads a zero. *Note: if TUFL is cleared in the last instruction of an Interrupt Service Routine triggered by this bit, then the Interrupt will be executed twice.*
15	**TDE**	R	0	**Transmitter Data Empty Flag –** Indicates that both left and right data words have been down loaded to the shift register. This bit is cleared by writing a one to the TUFCL bit (as explained above).
14	**TUFL**	R	0	**Transmitter Data Underflow –** Indicates that an underflow condition was detected. This condition occurs if all the transmit data registers are not written to in time (i.e. TDE is still set when the transmitter loads the shift register with new Left Data, implying that old data is re-transmitted.) This bit is cleared by writing a one to the TUFCL bit.
13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12	**TXIE**	RW	0	**Transmitter Interrupt Enable for DSP –** Interrupt enable for DSP.
11	**TDWE**	RW	0	**Transmitter Data Word Expansion –** Determines the way that the 24-bit data word to be transmitted is expanded to 32 bits during transmission. 0 The last bit is transmitted 8 times AFTER transmitting the 24-bit data word from the transmit data register. 1 The first bit is transmitted eight times BEFORE transmitting the 24-bit data word from the transmit data register is transmitted eight times.
10	**TREL**	RW	0	**Transmitter Relative Timing –** Determines the relative timing of the LRCKT signal as referred to the serial data inputs. 0 The transition of LRCKT occurs together with the first bit data. 1 The transition of LRCKT occurs one SCKT earlier (I 2 S format).
9	**TCKP**	RW	0	**Transmitter Clock Polarity –** Defines the polarity of the Transmitter clock. 1 Positive Clock Polarity. Means the LRCKT and SDO lines change synchronously with the positive edge of the clock, and are considered valid during negative transitions. 0 Negative Clock Polarity. Means the LRCKT and SDO lines change synchronously with the negative edge of the clock, and are considered valid during positive transitions
8	**TLRS**	RW	0	**Transmitter Left Right Selection –** Defines the polarity of the LRCKT. 0 If LRCKT = 0 then Left Word, if LRCKT = 1 then Right Word. 1 If LRCKT = 0 then Right Word, if LRCKT = 1 then Left Word.
7	**TDIR**	RW	0	**Transmitter Data Shift Direction –** Determines the shift direction of the transmitted data. 0 MSB First 1 LSB First.

Table 359. I^2S Transmit Status Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
6:5	**TWL**	RW	00	**Transmitter World Length Control –** Selects the length of the data word being transmitted by the SAI. 00 16 Bits 01 24 Bits 10 32 Bits 11 Reserved.
4	**RSRVD**	R	0	**Reserved –** Must be written with 0.
3	**TMME**	RW	0	**Transmitter master mode enable –** Reserved, must be set to 0.
2	**TEN2**	RW	0	**Transmitter Enable –** Enable bit for Transmitter channel 2.
1	**TEN1**	RW	0	**Transmitter Enable –** Enable bit for Transmitter channel 1.
0	**TEN0**	RW	0	**Transmitter Enable –** Enable bit for Transmitter channel 0.

Table 359. I^2S Transmit Status Control Register Description (Continued)

22.2.2.2. I^2S Transmit Status DataO0 Register

HW_SAITX0R X:$FFF6

BITS	LABEL	RW	RESET	DEFINITION
23:0	**TRANSMIT0**	RW	0	**Data to be sent I2S_DataO0 –** Most significant bit is sent first. 16 bit transfer data should be placed at most significant bytes (bits 23:16)

Table 360. I^2S Transmit Status DataI0 Register Description

22.2.2.3. I^2S Transmit Status DataO1 Register

HW_SAITX1R X:$FFF7

BITS	LABEL	RW	RESET	DEFINITION
23:0	**TRANSMIT1**	RW	0	**Data to be sent I2S_DataO1 –** Most significant bit is sent first. 16 bit transfer data should be placed at most significant bytes (bits 23:16)

Table 361. I^2S Transmit Status DataI1 Register Description

22.2.2.4. Transmit Status I^2S DataO2 Register

HW_SAITX2R X:$FFF8

BITS	LABEL	RW	RESET	DEFINITION
23:0	**TRANSMIT2**	RW	0	**Data to be sent I2S_DataO2 –** Most significant bit is sent first. 16 bit transfer data should be placed at most significant bytes (bits 23:16)

Table 362. I^2S Transmit Status DataI2 Register Description

22.2.3. Timing

Figure 92 shows the SAI timings for both the RDR and TDE flags. Internal flags allow some leeway in reading and writing the data registers as shown on the diagrams. The incoming and outgoing data is shifted in/out through a buffer shift register. This gives a full LRCKR half-period to service the RDR interrupt and read the left word received before it is overwritten by the next left word received. The right word will not be overwritten until the following LRCKR transition. Similarly for transmit there is a half-period of LRCKT from TDE being enabled to service the interrupt and write the next left/right pair to be transmitted.

22.2.4. Setting SAI Mode

To enable the pins into SAI mode, set the I2S_SELECT field of the HW_SPARER register. Unless over-ridden by the GPIO, etc., the I2S interface is always available. The I2S_SELECT bit, when set to one, invokes an alternate pin-out mode for channel 0 input in the larger 144-pin fpBGA package.

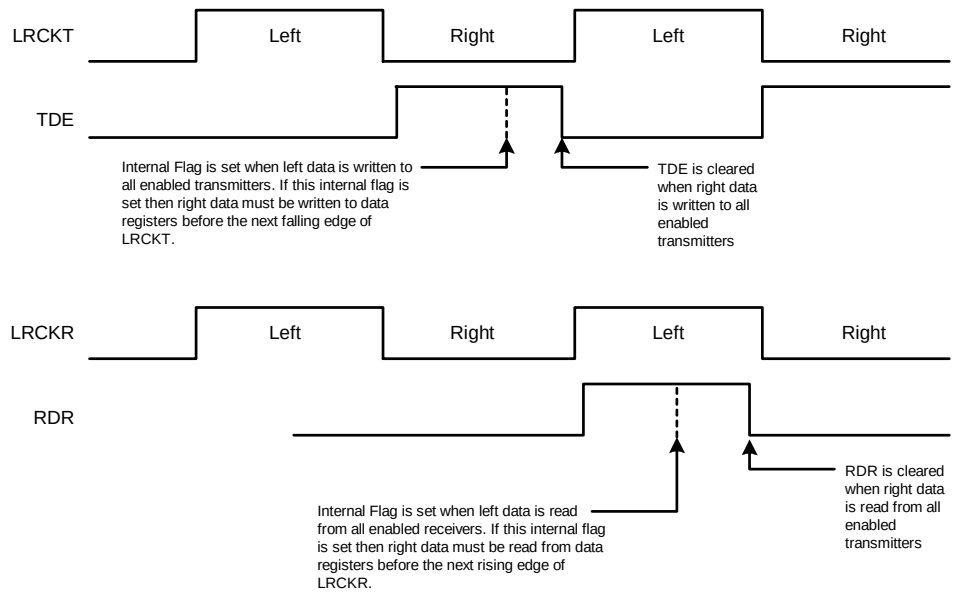


Figure 92. I²S Receive and Transmit Data Timing

22.2.5. Interrupts

The SAI interrupt vectors are located at the addresses shown in Table 363:

PRIORITY	PROGRAM ADDRESS
Receiver Overflow	P:$0016
Transmitter Underflow	P:$0012
Receiver Data Ready	P:$0014
Transmitter Data Empty	P:$0010

Table 363. SAI interrupts and Priorities

23. GENERAL PURPOSE INPUT/OUTPUT (GPIO)

The chip contains several GPIO modules. These modules provide flexible software control of each pin. Each digital pin can either be controlled by a hardware interface (SPI, Flash, etc.) or by a GPIO module. Every digital pin on the chip except for TESTMODE, ONCE_DRN, ONCE_DSI, ONCE_DSO and ONCE_DSK can be used as a GPIO pin. The GPIO configuration registers control the pin connection (GPIO or normal interface) and the pin function (if it is in GPIO mode). The GPIO configuration is independent for all of the pins. For example, the SmartMedia/NAND SM_CE3n pin can be configured as a GPIO pin while the other SmartMedia/NAND pins are connected to the Flash interface. If a pin is switched to be a GPIO, then the internal module connected to that pin will see a logic 0 on that pin, irrespective of the actual status of that pin.

The GPIO module is the only means of communicating with devices that don't use one of the chip's hardware interfaces. For example, LED's, buttons and other peripherals will require software that uses pins in GPIO mode. The pins are also used in GPIO mode for functions such as Key Scan, Backlights, SmartMedia card insert, etc.

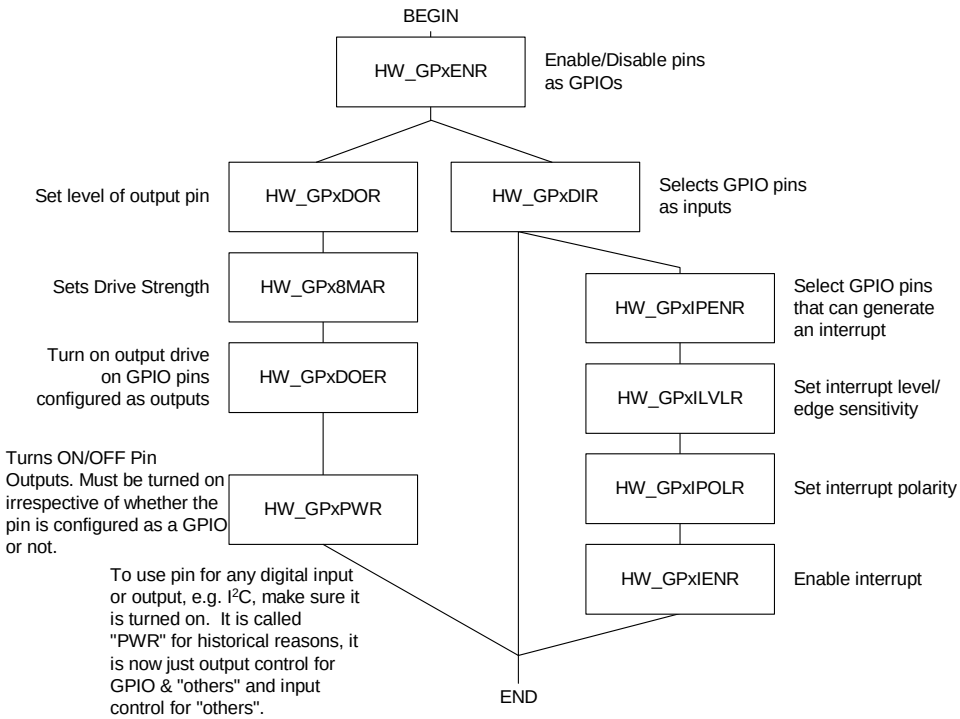


Figure 93. GPIO Setup Flow Chart

23.1. GPIO Interface

There are four GPIO Pin Registers (4 banks) on the chip used to configure digital pins as GPIO or their designated function: GPIO0, GPIO1, GPIO2 and GPIO3 (See Section 23.2.12. on page 266 for details). The following registers exist within each of these four banks to configure the chips digital pins. Some pins only exist in the 144-pin package options. The registers that control those pins exist but perform no useful function.

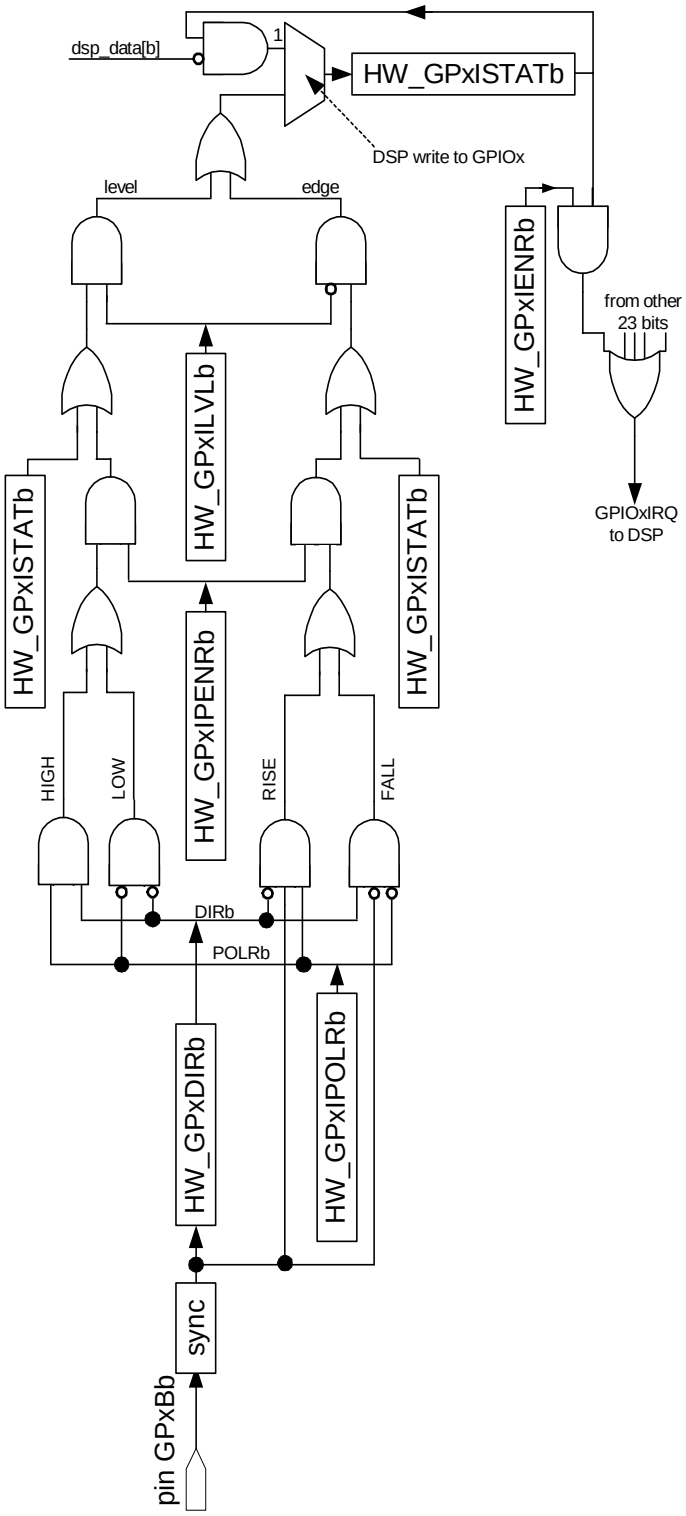


Figure 94. GPIO Interrupt Generation

23.2. GPIO Registers

23.2.1. GPIO Enable Register

This register enables most of the digital pins on the chip to be GPIOs or perform assigned functionality, like interfacing to an LED/buttons.

HW_GP0ENR X:$F400
HW_GP1ENR X:$F410
HW_GP2ENR X:$F420
HW_GP3ENR X:$F430

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**EN**	RW	0	**Module Enable bits –** These pins reset to 0. 0 Pin configured to perform assigned function 1 Pin configured as GPIO

Table 364. GPIO Enable Register Description

23.2.2. GPIO Data Out Register

This register allows GPIO to send data out on a pin-by-pin basis. Always set this register before GPIO Data Out Enable because as soon as the enable bit is set, the current data on the pin is sent out.

HW_GP0DOR X:$F401
HW_GP1DOR X:$F411
HW_GP2DOR X:$F421
HW_GP3DOR X:$F431

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**DO**	RW	0	Output bits if GPIO is set in GPIO Enable and the GPIO Output Enable bit is set.

Table 365. GPIO Data Out Register Description

23.2.3. GPIO Data In Register

This register allows GPIO to input data on a pin-by-pin basis.

HW_GP0DIR X:$F402
HW_GP1DIR X:$F412
HW_GP2DIR X:$F422
HW_GP3DIR X:$F432

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**DI**	R	0	Input bits if pin is in GPIO Mode and set as an input.

Table 366. GPIO Data In Register Description

23.2.4. GPIO Data Out Enable Register

This register allows GPIO to enable data output on a pin-by-pin basis.

HW_GP0DOER X:$F403
HW_GP1DOER X:$F413
HW_GP2DOER X:$F423
HW_GP3DOER X:$F433

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**DOE**	RW	0	**Output enable bits** 0 Input 1 Output

Table 367. GPIO Data Out Enable Register Description

23.2.5. GPIO Interrupt Pin Enable Register

This register allows GPIO to define specific pins to be interrupt pins.

HW_GP0IPENR X:$F404
HW_GP1IPENR X:$F414
HW_GP2IPENR X:$F424
HW_GP3IPENR X:$F434

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**IPEN**	RW	0	**Interrupt enable bits** 0 Corresponding pin is not interrupt pin 1 Corresponding pin is an interrupt pin

Table 368. GPIO Interrupt Pin Enable Register Description

23.2.6. GPIO Interrupt Enable Register

This register allows GPIO pins to enable specific interrupts to assert a DSP interrupt. Note that this register distinguishes between polling and interrupt driven routines. If a particular pin functions as an interrupt pin (as defined by GPIO Interrupt Pin Enable register), it's assertion will be reported in the interrupt status register but the interrupt signal to DSP will not be asserted if GPIO Interrupt Enable bit is also not asserted. Once an interrupt has been received in the interrupt collector and it has been determined that it is GPIO bank 0, 1, 2, or 3, then this register is used to determine which pin asserted the interrupt. See Table 372 for Interrupt Options.

HW_GP0IENR X:$F405
HW_GP1IENR X:$F415
HW_GP2IENR X:$F425
HW_GP3IENR X:$F435

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**EN**	RW	0	**Interrupt enable bits** 0 Interrupt Disable 1 Interrupt Enable

Table 369. GPIO Interrupt Enable Register Description

23.2.7. GPIO Interrupt Level Register

This register allows GPIO to define specific pins to be level or edge sensitive if they are enabled in GPIO Interrupt Enable register. See Table 372 for Interrupt Options.

HW_GP0ILVLR X:$F406
HW_GP1ILVLR X:$F416
HW_GP2ILVLR X:$F426
HW_GP3ILVLR X:$F436

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**ILVL**	RW	0	**Interrupt level bits** 0 Edge Sensitive 1 Level Sensitive

Table 370. GPIO Interrupt Level Register Description

23.2.8. GPIO Interrupt Polarity Register

This register allows GPIO to define specific pins to Active High/Low or Positive/Negative edge interrupt based on programming of corresponding bit in GPIO Interrupt Enable and GPIO Interrupt Level register. See Table 372 for Interrupt Options.

HW_GP0IPOLR X:$F407
HW_GP1IPOLR X:$F417
HW_GP2IPOLR X:$F427
HW_GP3IPOLR X:$F437

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**IPOL**	RW	0	**Interrupt Polarity bits** 0 Active low level / falling edge 1 Active high level / rising edge

Table 371. GPIO Interrupt Polarity Register Description

HW_GPXIENR (INT ENABLE)	HW_GPXILVLR (INT LEVEL)	HW_GPXIPOLR (INT POLARITY)	DESCRIPTION
0	X	X	Disable Interrupt Pin Functionality
1	1	0	Active Low Interrupt
1	1	1	Active High Interrupt
1	0	0	Falling Edge Interrupt
1	0	1	Rising Edge Interrupt

Table 372. GPIO Interrupt Options Table

23.2.9. GPIO Interrupt Status Register

This register allows GPIO to monitor and clear interrupts if corresponding bit is programmed as an interrupt pin. Combination of interrupt status and enables assert interrupt to DSP.

HW_GP0ISTATR X:$F408
HW_GP1ISTATR X:$F418
HW_GP2ISTATR X:$F428
HW_GP3ISTATR X:$F438

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**ISTAT**	RW	0	**Interrupt status bits –** Reading 1 indicates a pending interrupt. The bits in interrupt status register are set irrespective of Interrupt Enable bits. To clear edge interrupt, set the proper bit to 1.

Table 373. GPIO Interrupt Status Register Description

23.2.10. GPIO Pin Power Register

This register controls whether each pin is powered up irrespective of whether one pin is configured as a GPIO pin or not. When a pin is powered down its input will read as 0 irrespective of the voltage on the pin. Similarly, a pin that is powered down will always be high impedance (i.e. tristated) even if the pin is configured as an output. All pins are powered down after a chip reset.

HW_GP0PWR X:$F409
HW_GP1PWR X:$F419
HW_GP2PWR X:$F429
HW_GP3PWR X:$F439

BIT	LABEL	RW	RESET	DESCRIPTION
23:0	**PWR**	RW	0	0 Pin is powered down (reset value) 1 Pin is powered up (i.e. active)

Table 374. GPIO Pin Power Register Description

23.2.11. GPIO Pin Drive Strength Register

This register controls the output drive strengths of groups of pins, when the pins are configured as outputs. Note that unlike other GPIO registers, each pin does not get its own control bit. Instead pins are assigned in groups of 8 to an individual control bit. Each group of pins can be configured as 4mA drivers, or 8mA drivers. The required setting will depend on the application, but in general it is recommended that as many pins as possible are configured in 4mA mode. Doing so will increase the number of simultaneously pins that can be supported without adverse effects on the rest of the system, and will reduce the momentary peak load on the DCDC converters, and will reduce EMI emissions. A conservative limit on the maximum number of simultaneously switching output pins for the chip would be 24 4mA pins, or 12 8mA pins, or an equivalent combination of 4mA and 8mA pins.

It is possible to save some power by gating the clocks in the GPIO modules, this functionality is controlled by the CLKGATE field of the GPIO Pin Drive Strength register. When the clocks are gated, the DSP will no longer be able to write to any of the GPIO control registers, except for this one. When the clocks are gated, GPIO interrupts will continue to be generated, and the DSP will have full read access to all registers, including the GPIO Data Input registers. The DSP will need to clear the CLKGATE bit before it can change the value of pin configured as a GPIO output.

HW_GP08MA	X:$F40A
HW_GP18MA	X:$F41A
HW_GP28MA	X:$F42A
HW_GP38MA	X:$F43A

BIT	LABEL	RW	RESET	DESCRIPTION
23	**CLKGATE**	RW	0	0 Clocks not gated 1 Clocks gated
22:4	**RSRVD**	R	0	**Reserved –** Must be written with 0.
2	**PDS2**	RW	0	**Controls bits [23:15]** 0 4 mA driver 1 8 mA driver
1	**PDS1**	RW	0	**Controls bits [16:8]** 0 4 mA driver 1 8 mA driver
0	**PDS0**	RW	0	**Controls bits [7:0]** 0 4 mA driver 1 8 mA driver

Table 375. GPIO Pin Drive Strength Register Description

23.2.12. GPIO Register Pin Assignments

23.2.12.1. GPIO0 (Bank 0)

GPIO0 X:$F400–$F40A

BIT	LABEL	PIN NAME (NOTE 1)	DESCRIPTION
23:20	**RSRVD**		**Reserved –** Must be written with 0.
19	**GP0B19**	GP19	This pin is available in all packages.
18	**GP0B18**	GP18	This pin is available in all packages.
17	**GP0B17**	GP17	This pin is available in all packages.
16	**GP0B16**	GP16	This pin is available in all packages.
15	**GP0B15**	GP15	This pin is available in all packages.
14	**GP0B14**	GP14	This pin is available in all packages.
13	**GP0B13**	GP13	This pin is available in all packages.
12	**GP0B12**	GP12	This pin is available in all packages.
11	**GP0B11**	GP11	This pin is available in all packages.
10	**GP0B10**	GP10	This pin is available in all packages.
9	**GP0B9**	GP9	This pin is available in all packages.
8	**GP0B8**	GP8	This pin is available in all packages.
7	**GP0B7**	GP7	This pin is available in all packages.
6	**GP0B6**	GP6	This pin is available in all packages.
5	**GP0B5**	GP5	This pin is available in all packages.
4	**GP0B4**	GP4	This pin is available in all packages.
3	**GP0B3**	GP3	This pin is available in all packages.
2	**GP0B2**	GP2	This pin is available in all packages.
1	**GP0B1**	GP1	This pin is available in all packages.
0	**GP0B0**	GP0	This pin is available in all packages.

Note: 1. Please see Table 494. “General Purpose Input/Output Pins” on page 375.

Table 376. GPIO0 Pin Register (Bank 0) Description

23.2.12.2. GPIO1 (Bank 1)

GPIO1 X:$F410–$F41A

BIT	LABEL	PIN NAME (NOTE 1)	DESCRIPTION
23	**GP1B23**	GP47	This pin is available in all packages.
22	**GP1B22**	GP46	This pin is available in all packages.
21	**GP1B21**	GP45	This pin is available in all packages.
20	**GP1B20**	GP44	This pin is available in all packages.
19	**GP1B19**	GP43	This pin is available in all packages.
18	**GP1B18**	GP42	This pin is available in all packages.
17	**GP1B17**	GP41	This pin is available in all packages.
16	**GP1B16**	GP40	This pin is available in all packages.
15	**GP1B15**	GP39	This pin is available in all packages.
14	**GP1B14**	GP38	This pin is available in all packages.
13	**GP1B13**	GP37	This pin is available in all packages.
12	**GP1B12**	GP36	This pin is available in all packages.
11	**GP1B11**	GP35	This pin is available in all packages.
10	**GP1B10**	GP34	This pin is available in all packages.

Table 377. GPIO1 Pin Register (Bank 1) Description

BIT	LABEL	PIN NAME (NOTE 1)	DESCRIPTION
9	**GP1B9**	GP33	This pin is available in all packages.
8	**GP1B8**	GP32	This pin is available in all packages.
7	**GP1B7**	GP31	This pin is available in all packages.
6	**GP1B6**	GP30	This pin is available in all packages.
5	**GP1B5**	GP29	This pin is available in all packages.
4	**GP1B4**	GP28	This pin is available in all packages.
3	**GP1B3**	GP27	This pin is available in all packages.
2	**GP1B2**	GP26	This pin is available in all packages.
1	**GP1B1**	GP25	This pin is available in all packages.
0	**GP1B0**	GP24	This pin is available in all packages.

Note: 1. Please see Table 494. “General Purpose Input/Output Pins” on page 375.

Table 377. GPIO1 Pin Register (Bank 1) Description (Continued)

23.2.12.3. GPIO2 (Bank 2)

GPIO2 X:$F420 – $F42A

BIT	LABEL	PIN NAME (NOTE 1)	DESCRIPTION
23	**RSRVD**		**Reserved –** Must be written with 0.
22	**GP2B22**	GP70	This pin is only available in 144-pin packages.
21	**GP2B21**	GP69	This pin is only available in 144-pin packages.
20	**GP2B20**	GP68	This pin is only available in 144-pin packages.
19	**GP2B19**	GP67	This pin is only available in 144-pin packages.
18	**GP2B18**	GP66	This pin is only available in 144-pin packages.
17	**GP2B17**	GP65	This pin is only available in 144-pin packages.
16	**GP2B16**	GP64	This pin is only available in 144-pin packages.
15	**GP2B15**	GP63	This pin is only available in 144-pin packages.
14	**GP2B14**	GP62	This pin is only available in 144-pin packages.
13	**GP2B13**	GP61	This pin is only available in 144-pin packages.
12:9	**RSRVD**		**Reserved –** Must be written with 0.
8	**GP2B8**	GP56	This pin is available in all packages.
7	**GP2B7**	GP55	This pin is available in all packages.
6	**GP2B6**	GP54	This pin is available in all packages.
5	**GP2B5**	GP53	This pin is available in all packages.
4	**GP2B4**	GP52	This pin is available in all packages.
3	**GP2B3**	GP51	This pin is available in all packages.
2	**GP2B2**	GP50	This pin is available in all packages.
1	**GP2B1**	GP49	This pin is available in all packages.
0	**GP2B0**	GP48	This pin is available in all packages.

Note: 1. Please see Table 494. “General Purpose Input/Output Pins” on page 375.

Table 378. GPIO2 Pin Register (Bank 2) Description

23.2.12.4. GPIO3 (Bank 3)

GPIO3 X:$F430 – $F43A

BIT	LABEL	PIN NAME (NOTE 1)	DESCRIPTION
23:22	**RSRVD**		**Reserved –** Must be written with 0.
21	**GP3B21**	GP93	This pin is only available in 144-pin packages
20	**GP3B20**	GP92	This pin is only available in 144-pin packages
19	**GP3B19**	GP91	This pin is only available in 144-pin packages
18	**GP3B18**	GP90	This pin is only available in 144-pin packages
17	**GP3B17**	GP89	This pin is only available in 144-pin packages
16	**GP3B16**	GP88	This pin is only available in 144-pin packages
15	**GP3B15**	GP87	This pin is only available in 144-pin packages
14	**GP3B14**	GP86	This pin is only available in 144-pin packages
13	**GP3B13**	GP85	This pin is only available in 144-pin packages
12	**GP3B12**	GP84	This pin is only available in 144-pin packages
11	**GP3B11**	GP83	This pin is only available in 144-pin packages
10	**GP3B10**	GP82	This pin is only available in 144-pin packages
9	**GP3B9**	GP81	This pin is only available in 144-pin packages
8	**GP3B8**	GP80	This pin is only available in 144-pin packages
7	**GP3B7**	GP79	This pin is only available in 144-pin packages
6	**GP3B6**	GP78	This pin is only available in 144-pin packages
5	**GP3B5**	GP77	This pin is only available in 144-pin packages
4	**GP3B4**	GP76	This pin is only available in 144-pin packages
3	**GP3B3**	GP75	This pin is only available in 144-pin packages
2	**GP3B2**	GP74	This pin is only available in 144-pin packages
1	**GP3B1**	GP73	This pin is only available in 144-pin packages
0	**GP3B0**	GP72	This pin is only available in 144-pin packages

Note: 1. Please see Table 494. “General Purpose Input/Output Pins” on page 375..

Table 379. GPIO2 Pin Register (Bank 3) Description

24. DAC

The DAC behaves as a DMA device, outputting data from a modulo (circular) buffer in X/Y/P memory specified by software configuration. Software configuration begins by programming the DAC Base Address Register (**HW_DACBAR**), the DAC Modulo Register (**HW_DACMR**), and the DAC Current Position Register (**HW_DACCPR**) to specify the base address and modulo of the buffer to be used by the DAC as well as the current position in the DAC buffer. Next, the buffer is filled by software. The sample rate of the DAC may be specified in the DAC Sample Rate Register (**HW_DACSRR**). The sample rate of the DAC can be adjusted with fine resolution using this register. The number of samples available is specified in the DAC Word Count Register (**HW_DACWCR**). The interrupt threshold word count is specified in the DAC interrupt control Register (**HW_DACICR**). Finally, transmit is enabled, and the DAC begins to output data. As each DSP word is processed, the DAC Word Count Register (**HW_DACWCR**) is decremented and the DAC Current Position Register (**HW_DACCPR**) is incremented. When the word count decrements to the value in the interrupt control register, an interrupt will occur if interrupts have been enabled in the DAC Control Status Register (**HW_DACCSR**). Software is then responsible for writing to **HW_DACWCR** register indicating more data is available. If the **HW_DACWCR** is not updated before the last sample in the buffer is needed by the DAC, an exception will occur, and a DAC underflow interrupt will be generated.

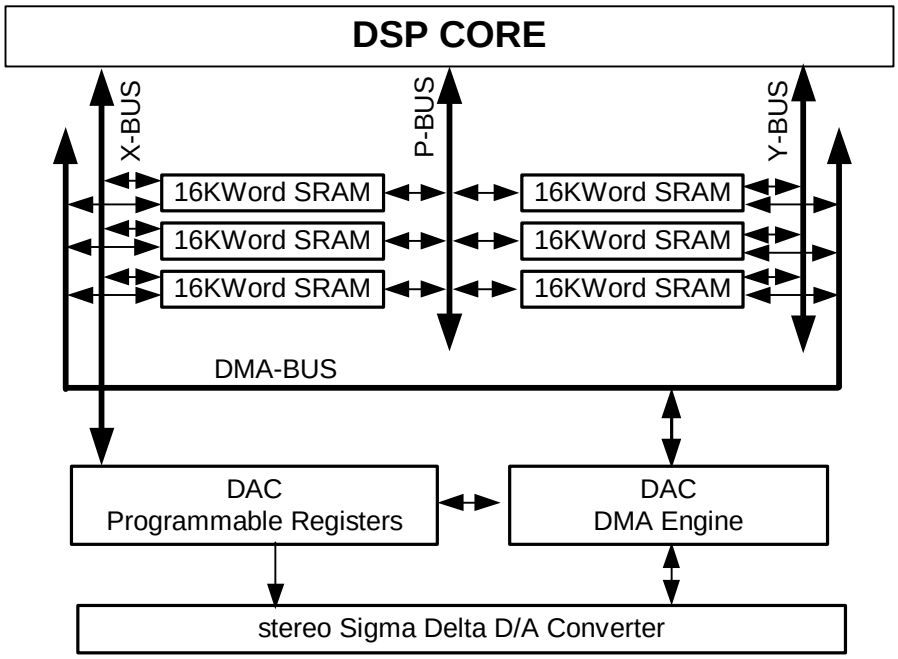


Figure 95. Stereo Sigma Delta D/A and DMA

DAC data in stored in X/Y/P memory with the left channel sample in the lowest address, followed by the corresponding right channel sample in the next memory location. The DAC data should always consist of an even number of samples to allow left and right channels, it is not possible to play mono data unless the mono samples are each repeated twice in memory, once for the left channel and once for the right channel. The data is stored as 24 bit 2's compliment values, where the full scale value depends on the programming of the sample rate converter. The DAC

expects data samples that already have been 2x over-sampled by code in the DSP, i.e. to play 44.1k sample/sec data, there should be 44.1k * 2 = 88.2k samples/sec per channel in the buffer in on-chip memory.

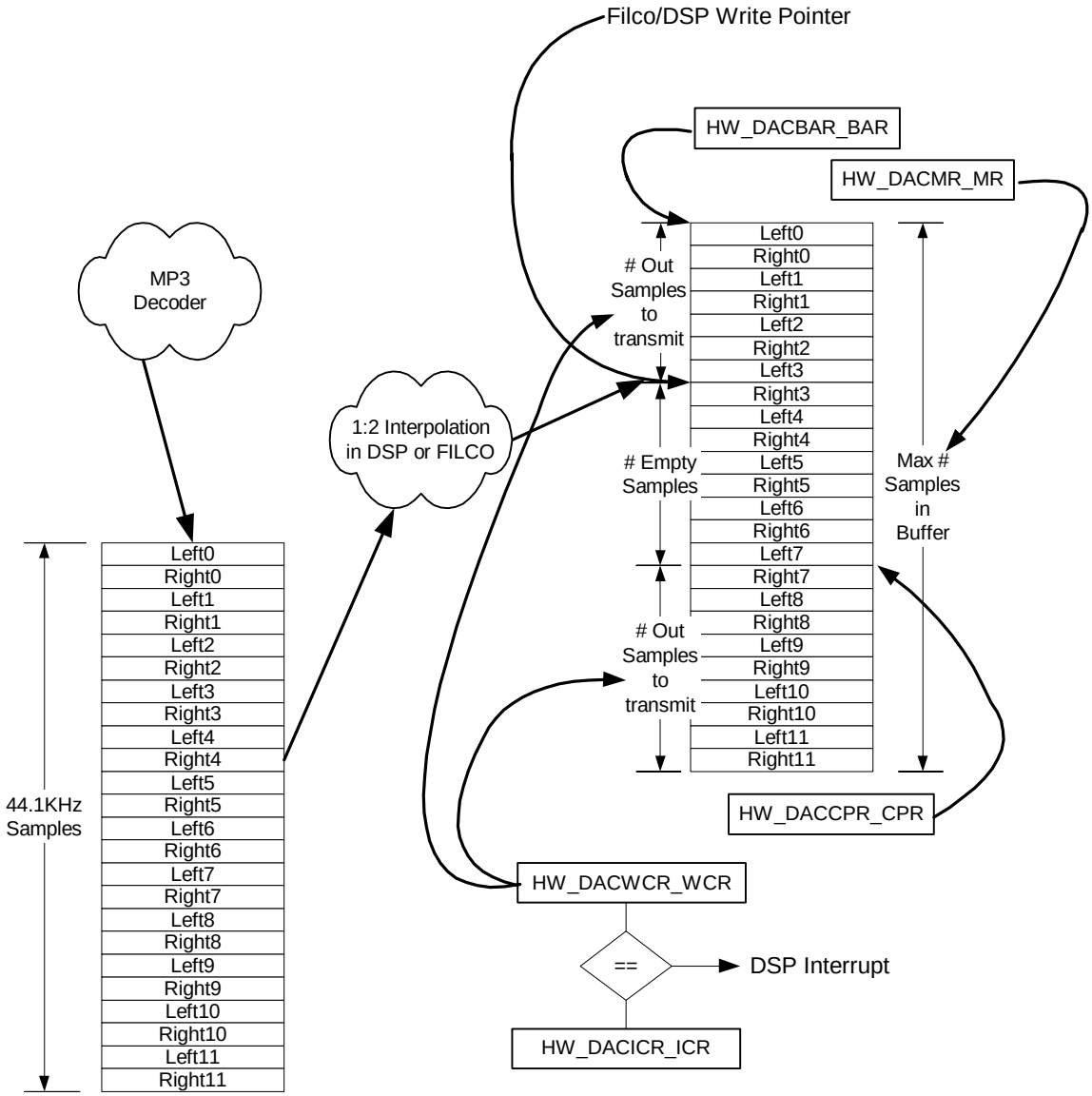


Figure 96. Stereo Sigma Delta D/A DMA Buffer Registers

A producer/consumer double buffer mechanism can be defined by setting **HW_DACICR_ICR** to half the size of **HW_DACMR_MR**. This will let the DAC consume half the buffer before the next DAC interrupt is generated.

Before the DAC can be turned on and used, the crystal clock used by the DAC and mixer must be enabled using the ACKEN bit of the Clock Control Register (**HW_CCR**). Also the DAC and Mixer analog circuitry must be powered up using the PR1 & PR2 bits of the Mixer Powerdown Control/Status Register (**HW_MIXPWRDNR**).

24.1. DAC Registers

24.1.1. DAC Base Address Register

The DAC Base Address Register is used to specify the base address of the modulo buffer used by the DAC port. The buffer's base address must zero the k LSBs, where 2^k >= HW_DACMR.

HW_DACBAR X:$F805

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								BAR															

Table 380. HW_DACBAR

BIT	LABEL	RW	RESET	DESCRIPTION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**BAR**	RW	0	Base address for the DAC output sample buffer in memory.

Table 381. DAC Base Address Register Description

24.1.2. DAC Modulo Register

The DAC Modulo Register specifies the modulus of the DAC buffer. The modulus is specified in the same manner as the Mn modulo registers of the Address Generation Unit of the DSP core. For example, writing $000001 to the HW_DACMR indicates a modulo buffer of size $000002.

HW_DACMR X:$F804

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														MR									

Table 382. HW_DACMR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**MR**	RW	0	**Modulo** (i.e. size) of the DAC output sample buffer in memory.

Table 383. DAC Modulo Register Description

24.1.3. DAC Current Position Register

The DAC Current Position Register indicates the address offset from the address specified in the HW_DACBAR DAC Base Address Register where the DAC will read the next output sample.

HW_DACCPR X:$F803

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														CPR									

Table 384. HW_DACCPR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**CPR**	RW	0	**Current read position** of the DAC output sample buffer in memory.

Table 385. DAC Current Position Register Description

24.1.4. DAC Word Count Register

The DAC Word Count Register specifies the number of words remaining in the on-chip memory buffer to be output. Software should fill the memory buffer with sample data, then program the **HW_DACWCR** to indicate the number of words available to the DAC to be output. For each sample output, the **HW_DACWCR** is decremented twice, one each for the left and right samples. When the **HW_DACWCR_WCR** == **HW_DACICR_IPT**, a transmit interrupt is generated. If the **HW_DACWCR_WCR** reaches zero and is not updated before the DAC requires another sample (i.e., under-run), then and exception occurs (**HW_DACCSR_TXEXC** is set). Writing to the **HW_DACWCR_WCR** also clears any pending DAC interrupts.

The hardware avoids the race condition between the DMA decrementing **HW_DACWCR_WCR** and software attempting to indicate the addition of more samples to the DAC buffer. When the DAC is not enabled (**HW_DACCSR_TXEN**==0), then writes to **HW_DACWCR_WCR** directly replace the current contents. When the DAC is enabled (**HW_DACCSR_TXEN**==1), then writes to **HW_DACWCR_WCR** add to the current contents of **HW_DACWCR_WCR**, as follows:

HW_DACWCR_WCR = **HW_DACWCR_WCR** + DSP_WRITE_VALUE

If the DMA attempts to decrement the word count on the same cycle that the DSP attempts to write to it then the new value becomes:

HW_DACWCR_WCR = **HW_DACWCR_WCR** + DSP_WRITE_VALUE -1

This register counts words in memory, not samples. Since the DAC operates on stereo data, the number of words will be twice the number of samples.

HW_DACWCR X:$F802

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														WCR									

Table 386. HW_DACWCR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**WCR**	RW	0	Number of DAC samples remaining until a DAC interrupt will be generated.

Table 387. DAC Word Count Register Description

24.1.5. DAC INTERRUPT CONTROL REGISTER

The DAC interrupt control register specifies the number of samples remaining in the DAC output buffer at which to trigger an interrupt. This allows the interrupt to be generated well ahead of a completely empty buffer.

HW_DACICR X:$F806

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														IPT									

Table 388. HW_DACICR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**IPT**	RW	0	DAC interrupt will be generated on the sample when **HW_DACWCR_WCR** == **HW_DACICR_IPT**. Since this register initializes to zero, the default behavior is the same as for the STMP3410 where an interrupt is generated on the sample when **HW_DACWCR_WCR** reaches zero.

Table 389. DAC Interrupt Control Register Description

24.1.6. DAC Sample Rate Register

The DAC Sample Rate Register is programmed to specify the sample rate of the DAC.

HW_DACSRR X:$F801

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	SR																						

Table 390. HW_DACSRR

BIT	LABEL	RW	RESET	DESCRIPTION
23	**RSRVD**	R	0	**Reserved –** Must be written with 0.
22:0	**SR**	RW	0	**Sample Rate** to use for DAC playback.

Table 391. DAC Sample Rate Register Description

Most readers can use the values in Table 392 and simply skip the rest of the detailed discussion in this section. The following table contains the required value of the **HW_DACSRR** register for various common sample rates. If the crystal is changed from the normal 24.0 MHz rate, the $Fanalog_{DAC}$ frequency will change, and the **HW_DACSRR** register must be changed to compensate. Values for both 24.0MHz

and 24.576MHz are given in the table. The table assumes **HW_CCR_DACDIV** is set to zero to cause the crystal clock to be divided by four ($Fanalog_{DAC}$ = 6.0MHz).

.

	24.0 MHZ	24.576 MHZ
SAMPLE RATE	**HW_DACSRR_SR VALUE**	**HW_DACSRR_SR VALUE**
$Fsample_{DAC}$		
48,000 Hz	$06D000	$070000
44,100 Hz	$0780DF	$07B51E
32,000 Hz	$0AB800	$0B0000
24,000 Hz	$0EA000	$0F0000
22,050 Hz	$1001BE	$106A3B
16,000 Hz	$167000	$170000
12,000 Hz	$1E4000	$1F0000
11,025 Hz	$21037C	$21D476
8,000 Hz	$2DE000	$5F0000

Table 392. Values for the HW_DACSRR registers

For any of the desired sample rates, the value of the **HW_DACSRR** register is calculated according to the following formula:

$$OSR_{DAC} = (65536 * 8 * Fanalog_{DAC}) / (128 * Fsample_{DAC})$$
$$= (4096 * Fanalog_{DAC}) / Fsample_{DAC}$$

If computed with the above explicit operator precedence, the resulting over sample rate (OSR_{DAC}) will be a 24 bit scaled fixed point representation of the desired decimation rate. In order to load the **HW_DACSRR_SR** field, we need to account for the interpolator's pos_zero comparison. To do this, we subtract one from the whole number portion of the OSRDAC. We load the SR bit field with:

HW_DACSRR_SR = (OSR_{DAC} - $010000)

The value stored in the **HW_DACSRR** is interpreted as a 24 bit unsigned value.

WARNING: The computation of the scaled fixed point OSR_{DAC} value has an intermediate result precision of 35 bits before the division. Long Long (64-bit) C variables should be used to avoid errors due to loss of precision. These are static values that are hard coded into applications for the nine sample rates of interest. Results should be rounded not truncated.

NOTE: While the OSR calculation, above, uses the desired sample rate, the buffer must actually contain audio samples at twice this sample rate. Therefore to play a 44.1KHz audio stream through the DAC, one must present an 88.2KHz sample rate interpolated version of the original samples in the buffer that will be fetched by the DAC's DMA, see Figure 96. "Stereo Sigma Delta D/A DMA Buffer Registers" on page 270. Thus a 2:1 interpolation algorithm must have been run on the 44.1KHz stream before it can be passed to the DMA. This algorithm can run either in DSP software or in the Filter coprocessor, see Section 13. "FILTER COPROCESSOR (FILCO)" on page 151.

The 1-bit sigma delta D/A converter is always sampled on a submultiple of the 24.0MHz crystal oscillator frequency as specified in the **HW_CCR_DACDIV** register, see Figure 97. "Stereo Sigma Delta D/A Converter" on page 275. This divider generates sample strobes at $Fanalog_{DAC}$ where the divisors available come

from the set {4,6,8,12,16,24}. With **HW_CCR_DACDIV** set to zero, to divide by 4, $Fanalog_{DAC}$ becomes 6.0MHz for a 24.0 MHz crystal. The sample strobe derived from this divider is used to interpolate the 1 bit D/A values. The 1-bit sigma delta modulator is effectively running at $Fanalog_{DAC}$. As shown in Figure 97, the 24 bit 2x sample rate D/A values are extracted from on-chip RAM via the DMA. They are filtered to band limit the audio stream. This filter runs on xtal_clk but filters samples at the source sample rate which is slower than the output D/A sample. In the process, this filter performs a fixed 1:8 interpolation or up-sample of the already 2X up-sampled input stream. Thus the output of the 1:8 fixed interpolation filter is running at 16X $Fsample_{DAC}$. A single 24 bit sample at the output of the fixed filter is further interpolated up to the 1-bit D/A rate. The variable rate sample, hold and interpolate block performs this function.

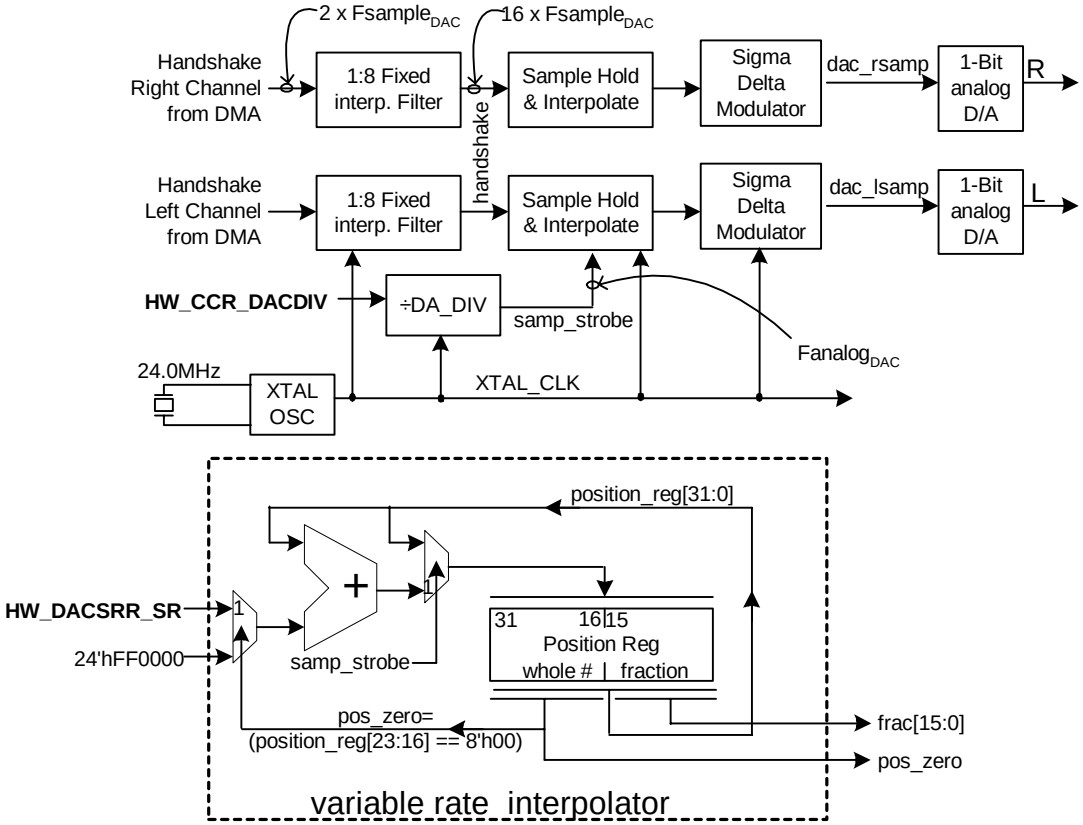


Figure 97. Stereo Sigma Delta D/A Converter

It stalls the filter pipeline and DMA source, using the handshake lines that connect with the previous filter stage to supply samples at the correct over-sample ratio. The 1-bit DAC runs at the fixed sample rate of $Fanalog_{DAC}$ while the DMA fetches samples in burst at irregular intervals to maintain the required input to the modulator.

In this case, the 1-bit D/A is running at 136.054 times the desired sample rate of 44.1KHz or 6MHz. The sample hold and interpolate block accepts a new sample from the filter at a 44.1KHz times 2 times 8 or 705.6KHz. It passes interpolated samples to the modulator at a 6.0MHz rate. The sample, hold and interpolate block passes a source sample from the fixed 1:8 interpolation filter to the sigma delta modulator corresponding to every 8.503 $Fanalog_{DAC}$ samples. Recall that this is a

variable rate interpolation stage which changes for every Over Sample Rate (OSR) setting in use.

If the desired sample rate $Fsample_{DAC}$= 44.1KHz, for example, the sample hold and interpolate block will accept samples from fixed interpolation filter at 705.6KHz, i.e. 16 times the desired sample rate. There is a handshake pair (request/ack) between the variable rate sample hold and interpolate block and the fixed interpolating filter block. This hand shake is used to pace the samples from the filter and DMA buffer to 88.2KHz (2x the desired 44.1KHz) rate. An interpolation filter routine was used within the DSP to generate the 88.2KHz sample stream from the 44.1KHz source.

Of course, this is not all the sample rates of interest! There are the members of the 48KHz family, whose members satisfy the property:

$$24.576MHz = Q*Fsample_{DAC}.$$

These sample rates include 48KHz, 32KHz, 24KHz, 16KHz, and 12KHz. There are also the members of the 44.1KHz family, whose members satisfy the property:

$$16.9344MHz = Q*Fsample_{ADC}$$

where Q comes from the set of integers. These sample rates include 44.1KHz, 22.05KHz and 11.025KHz. Since 24.0MHz and 16.9344MHz are relatively prime, members of the 44.1KHz family are related to the 24.0MHz source clock by the relationship $24.0MHz=P*Fsample_{ADC}$, where P is a rational number.

The D/A converter block includes a variable rate or rational interpolator as shown in Figure 97 to accommodate these sample rates. Rational numbers in the DAC are approximated with a scaled fixed point 24 bit value. In this case, the decimal point falls between bit 15 and bit 16. Therefore the lower two bytes hold the fractional part while the upper byte holds the whole number portion of the scaled fixed point. The position register uses this scaled fixed point representation for the number of 1-bit samples to be interpolated (generated) to find the next sample to be sent to the sigma delta modulator. Whenever the whole number part (viz. bits 23:16) are zero, then the next DMA sample is consumed. For playback at 44.1KHz, we have to set the interpolator to generate 67.027 new interpolated samples between every DMA sample.

To accomplish this we load **HW_DACSRR_SR** with a value of $0780DF in the 24 bit scaled fixed point representation or 7.503 decimal. Referring to the bottom of Figure 97, at reset the position register is loaded with zero so the pos_zero boolean is true. Thus the value $0780DF will be continuously added to the position_reg. After the first addition, pos_zero is no longer true so the value $FF0000 is added to the position-reg. This is effectively a value of minus 1 in the whole number portion resulting in a position register value of $0680DF (6.503). Thus seven 1-bit samples will be generated. At the eighth sample, the DAC will consume the next 24-bit value from the fixed 1:8 filter. Since the upper part of the position register is once again zero, the $0780DF is added to it. The lower 16-bits still contain $00DF so the new position register is $781BE or 7.507decimal. After enough samples are fetched, the fractional part summation will overflow into an integral sample increase causing an additional interpolation cycle. Thus the fractional part of the rational interpolator periodically corrects the diffused interpolation error and corrects for it.

24.1.7. DAC Control Status Register

HW_DACCSR X:$F800

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CLKGT													VUP	RVUP	LVUP	DAC_HPPOP	DMASEL		LPBK	TXEXC	TXI	TXIEN	TXEN

Table 393. HW_DACCSR

BITS	LABEL	RW	RESET	DEFINITION
23	**CLKGT**	RW	0	**Clock gate –** Used to disable the clocks to the DAC module to conserve power when the DAC is not in use. Must be set to 0 before writing to any other DAC registers. 0 - Clocks not gated 1 - Clocks are gated off
22:11	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10	**VUP**	R	0	**Volume Update Pending on Either Channel–** This bit is set one if a volume update is pending, i.e. waiting for zero crossing, on either the left or right channels or both.
9	**RVUP**			**Volume Update Pending on Right Channel–** This bit is set one if a volume update is pending, i.e. waiting for zero crossing, on the right channel.
8	**LVUP**	R	0	**Volume Update Pending on Left Channel–** This bit is set one if a volume update is pending, i.e. waiting for zero crossing, on the left channel.
7	**DAC_HPPOP_EN**	RW	0	**DAC POP Suppressor Enable –** Set this bit to one to enable the pop suppression algorithm. Set this bit to zero to disable the pop suppressor and cause the DAC to mimic the STMP3410 behavior. The pop suppressor works by sending a continuous alternating zero and one stream of samples (101010101010) whenever the DAC is disabled (**HW_DACCSR_TXEN** =0).
6:5	**DMASEL**	RW	00	**Memory space to use for DMA transfers** 00 - X space 01 - Y space 10 - P space 11 - Reserved
4	**LPBK**	RW	0	**Loopback –** When set, the LPBK bit connects the ADC single bit data from the ADC analog circuitry directly to the DAC analog circuitry to perform an analog loopback test without utilizing the DSP core.
3	**TXEXC**	RW	0	**Transmit Exception –** The TXEXC bit is a status bit indicating a transmit exception has occurred. A transmit exception occurs when the DAC needs to read a sample but the HW_DACWCR is zero indicating no samples are available. The TXEXC bit is cleared by writing a 0 to this location. The TXEXC can also be cleared by writing to the HW_DACWCR register.

Table 394. DAC Control Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
2	**TXI**	RW	0	**Transmit Interrupt –** The TXI bit is a status bit indicating a transmit interrupt has occurred. The TXI bit is cleared by writing a 0 to this location. The Transmit Interrupt can also be cleared by writing to the HW_DACWCR register.
1	**TXIEN**	RW	0	**Transmit Interrupt Enable –** The TXIEN bit enables interrupt generation for this port. When the HW_DACWCR reaches zero, an interrupt is generated.
0	**TXEN**	RW	0	**Transmit Enable –** Setting the TXEN bit causes the DAC Port to begin outputting samples from the modulo buffer specified by the HW_DACBAR, HW_DACMR, and HW_DACWCR. When TXEN is cleared, the TXI and TXEXC bits are cleared. No other DAC registers are cleared.

Table 394. DAC Control Status Register Description (Continued)

25. ADC

The ADC behaves as a DMA device which places data in a modulo (circular) buffer in X/Y/P memory specified by software configuration. Software configuration begins by programming the ADC Base Address Register (**HW_ADCBAR**), the ADC Modulo Register (**HW_ADCMR)**, and the ADC Current Position Register (**HW_ADCCPR**) to specify the base address and modulo of the buffer to be used by the ADC as well as the current position in the ADC buffer. The sample rate of the ADC must be specified in the ADC Sample Rate Register (**HW_ADCSRR**). The number of available sample entries are specified in the ADC Word Count Register (**HW_ADCWCR**). The ADC interrupt threshold word count is specified in the ADC Interrupt Control Register (**HW_ADCICR**). At this point sample capture is enabled and the ADC begins to input stereo data. As each DSP word is input, the **HW_ADCWCR** is decremented and the ADC Current Position Register (**HW_ADCCPR**) is incremented. When the word count decrements to the value in the **HW_ADCICR_IPT**, an interrupt will occur if interrupts have been enabled in the ADC Control Status Register (**HW_ADCCSR**). Software is then responsible for writing to **HW_ADCWCR** register indicating space for more data is available. If the **HW_ADCWCR** is not updated before the last free sample location needs to be written to on-chip RAM by the ADC, an exception will occur and an ADC overflow interrupt will be generated.

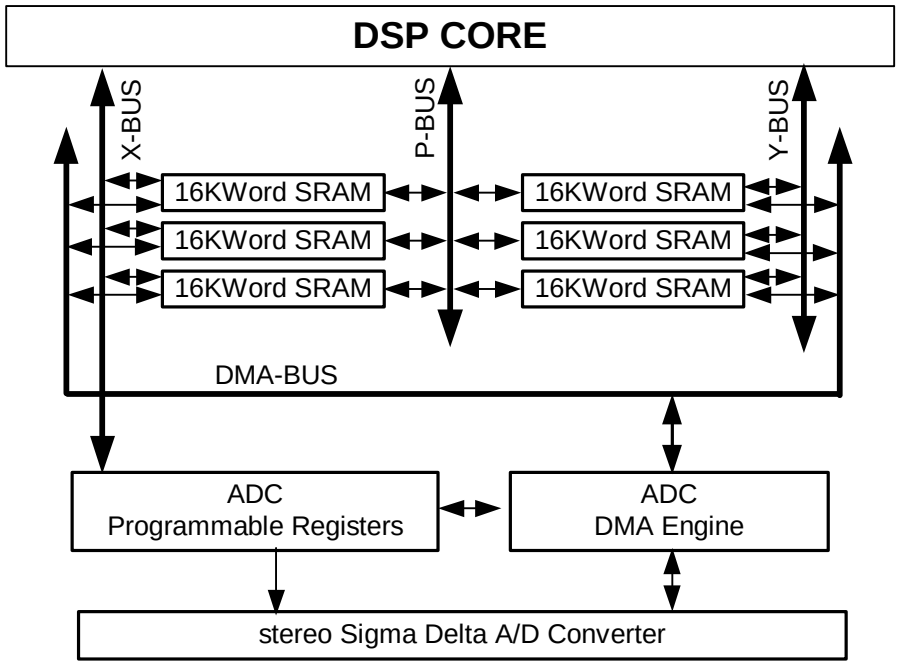


Figure 98. Stereo Sigma Delta A/D and DMA

ADC data is stored in X/Y/P memory with the left channel sample in the lowest address, followed by the corresponding right channel sample in the next memory location. The ADC data always consists of an even number of words to allow left and right channels, it is not possible to record mono data unless the stereo data is recorded and one of the channels is thrown away by the DSP software. The data is stored as 24 bit 2's compliment values, where the full scale value depends on the programming of the sample rate converter. The ADC generates data samples that are 8x over-sampled, and that need to be decimated to the base rate by code in the DSP. For example, to record 44.1k sample/sec data, the ADC will generate 44.1k * 8

= 352.8k samples/sec per channel in memory, which must be filtered and decimated down to 44.1k samples/sec by code executing on the DSP or by utilizing three 2:1 decimation stages in the Filter Coprocessor.

WARNING: While the ADC is designed to produce 8X over sampled streams relative to the desired sample rate, these are 8X over sampled at a lower precision. DSP filtering should always filter and decimate not just decimate the result.

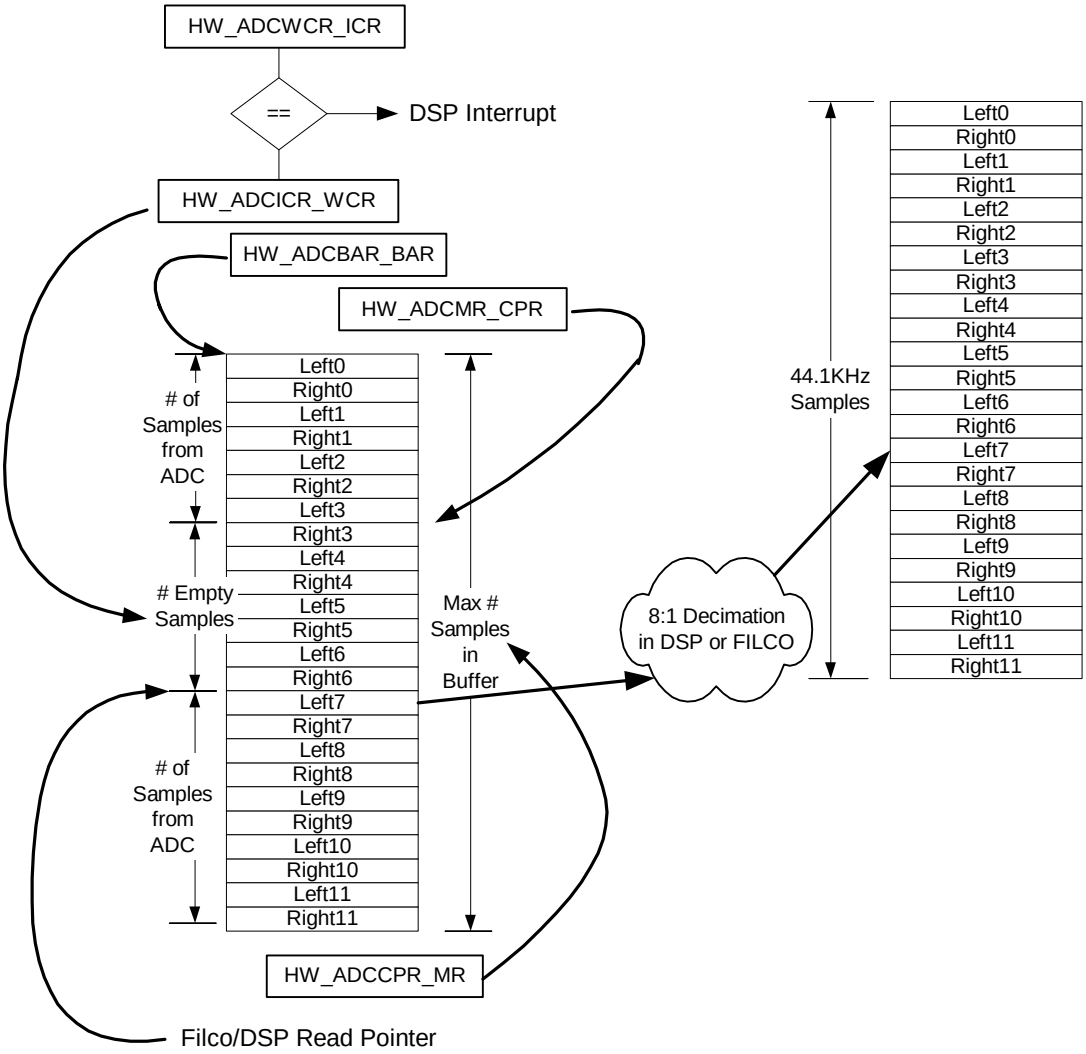


Figure 99. Stereo Sigma Delta A/D and DMA Buffers

A producer/consumer double buffer mechanism can be defined by setting **HW_ADCICR_IPT** to half the size of **HW_ADCWCR_WCR**. The ADC can then fill half the buffer before generating an interrupt. Figure 99 shows that ADC utilizes a circular buffer whose base address is specified in **HW_ADCBAR_BAR**. The total number of words in the circular is specified in **HW_ADCMR_MR**. The word that will next by written with a new ADC sample value is pointed to by the current position register, **HW_ADCCPR_CPR**. The decimation software or the FILCO has a read pointer that points to the next sample to be read. The difference in these two positions corresponds to the total number of unused locations in the buffer. The software is responsible for notifying the ADC of the number of words it or FILCO have read

from the buffer. It does this by writing the number consumed to **HW_ADCWCR_WCR**. The ADC adds the write value to its internal word count. The total word count maintained by the ADC represents the number of unfilled locations available. If that size falls to zero an overflow is reported on the next sample captured. As that word count falls below the **HW_ADCICR_IPT** then an interrupt is generated, notifying software that more samples are available in the buffer.

ADC write addresses are generated by logically *OR*ing the BAR and current position registers together. Therefore the BAR must be aligned with next larger power of two than the modulo value. The modulo value does not have to be a power of two.

Before the ADC can be turned on and used, the crystal clock used by the ADC must be enabled using the **ACKEN** bit of the **HW_CCR** Clock Control register. Also the ADC analog circuitry must be powered up using the **PR0** bit of the **HW_MIXPWRDNR** Mixer Powerdown Control/Status register. The value in **HW_CCR_ADCDIV** may have to be modified for some sample rates and crystal frequency combinations.

25.1. ADC Registers

25.1.1. ADC Base Address Register

The **HW_ADCBAR** is used to specify the base address of the modulo buffer used by the ADC. The buffer's base address must zero the k LSBs, where 2^k >= **HW_ADCMR**.

HW_ADCBAR X:$FB05

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								BAR															

Table 395. HW_ADCBAR

BIT	LABEL	RW	RESET	DESCRIPTION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15:0	**BAR**	RW	0	**Base address** for the ADC input sample buffer in memory.

Table 396. ADC Base Address Register Description

25.1.2. ADC Modulo Register

The ADC Modulo Register specifies the modulus of the ADC buffer. The modulus specified in the same manner as the Mn modulo registers of the Address Generation Unit. For example, writing $0001 to the **HW_ADCMR_MR** indicates a modulo buffer of size $0002.

HW_ADCMR X:$FB04

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														MR									

Table 397. HW_ADCMR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**MR**	RW	0	**Modulo** (i.e. size) of the ADC input sample buffer in memory.

Table 398. ADC Modulo Register Description

25.1.3. ADC Current Position Register

The ADC Current Position Register indicates the address offset from the address specified by the ADC Base Address Register (**HW_ADCBAR_BAR**) where the ADC will write the next output sample.

HW_ADCCPR X:$FB03

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														CPR									

Table 399. HW_ADCCPR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**CPR**	RW	0	**Current write position** of the ADC input sample buffer in memory.

Table 400. ADC Current Position Register Description

25.1.4. ADC Word Count Register

The ADC Word Count Register specifies the number of words that are available in the circular buffer to hold new samples. Software should allocate a memory buffer to hold the sample data, then program the **HW_ADCWCR_WCR** to indicate the number of words available to the ADC to hold the samples being captured. For each sample captured, the **HW_ADCWCR_WCR** is decremented twice, once each for the left and right samples. When the **HW_ADCWCR_WCR** == **HW_ADCICR_IPT**, a transmit interrupt is generated. If the **HW_ADCWCR_WCR** reaches zero before the ADC needs to write another sample to on-chip RAM (i.e., overrun), then an exception occurs (**HW_ADCCSR_TXEXC** is set). Writing to the **HW_ADCWCR_WCR** clears any pending interrupt.

The hardware avoids the race condition between the DMA decrementing **HW_ADCWCR_WCR** and software attempting to indicate the availability of more sample space by writing to **HW_ADCWCR_WCR**. When the ADC is not enabled (**HW_ADCCSR_TXEN** == 0) then writes to the **HW_ADCWCR_WCR** replace the current contents. When the ADC is enabled, (HW_ADCCSR_TXEN = 1) then writes to the **HW_ADCWCR_WCR** register add to the current contents as follows:

HW_ADCWCR_WCR = **HW_ADCWCR_WCR** + DSP_WRITE_VALUE

If the DMA attempts to decrement the word count on the same cycle that the DSP attempts to write to it, then the new value becomes:

HW_ADCWCR_WCR = **HW_ADCWCR_WCR** + DSP_WRITE_VALUE - 1

This register counts words in memory, not samples. Since the ADC operates on stereo data, the number of words will be twice the number of samples.

HW_ADCWCR X:$FB02

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														WCR									

Table 401. HW_ADCWCR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**WCR**	RW	0	Number of ADC samples remaining until an ADC interrupt will be generated.

Table 402. ADC Word Count Register Description

25.1.5. ADC INTERRUPT CONTROL REGISTER

The ADC interrupt control register specifies the number of empty sample entries remaining in the ADC input buffer at which to trigger an interrupt. This allows the interrupt to be generated well ahead of a completely full buffer.

HW_ADCICR X:$FB06

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														IPT									

Table 403. HW_ADCICR

BIT	LABEL	RW	RESET	DESCRIPTION
23:10	**RSRVD**	R	0	**Reserved –** Must be written with 0.
9:0	**IPT**	RW	0	DAC interrupt will be generated on the sample when **HW_ADCWCR_WCR** == **HW_ADCICR_IPT**. Since this register initializes to zero, the default behavior is the same as for the STMP3410 where an interrupt is generated on the sample when **HW_ADCWCR_WCR** reaches zero.

Table 404. ADC Interrupt Control Register Description

25.1.6. ADC Sample Rate Register

The ADC Sample Rate Register is programmed to specify the sample rate of the ADC.

HW_ADCSRR X:$FB01

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	SR																						

Table 405. HW_ADCSRR

BIT	LABEL	RW	RESET	DESCRIPTION
23	**RSRVD**	R	0	**Reserved –** Must be written with 0.
22:0	**SR**	RW	0	**Sample Rate** to use for ADC recording.

Table 406. ADC Sample Rate Register Description

Most readers can simply use the values in Table 407 and skip the following detailed discussion. The following table contains the required value of the **HW_ADCSSR_SR** register for various common sample rates. If the crystal is changed from the normal 24.0 MHz rate, the Fanalog frequency will change, and the **HW_ADCSSR_SR** register must be changed to compensate. Values for both 24.0MHz and 24.576MHz are provided. The table assumes that **HW_CCR_ADCDIV** is set to zero, i.e. $Fanalog_{ADC}$ = 6.0MHz/6.144MHz.

	24.0MHz, Fanalog = 6.0MHz		**24.576MHz, Fanalog = 6.144MHz**	
SAMPLE RATE	**HW_ADCSRR_SR**	**SCALE FACTOR**	**HW_ADCSRR_SR**	**SCALE FACTOR**
$Fsample_{ADC}$	Value		Value	
48,000 Hz	$0EA000	3106	$0F0000	3106
44,100 Hz	$1001BE	2621	$106A3B	2621
32,000 Hz	$0F000	1380	$170000	1380
24,000 Hz	$1E4000	776	$1F0000	776
22,050 Hz	$21037C	655	$21D476	655
16,000 Hz	$2DE000	345	$2F0000	345
12,000 Hz	$3D8000	194	$3F0000	194
11,025 Hz	$4306F7	164	$44A8ED	164
8,000 Hz	$5CC000	86	$5F0000	86

Table 407. Values for the HW_ADCSRR_SR register

For any of the desired sample rates, the value of the **HW_ADCSSR_SR** register is calculated according to the following formula:

$$OSR_{ADC} = (65536 * 16 * Fanalog_{ADC}) / (128 * Fsample_{ADC})$$
$$= (8192* Fanalog_{ADC}) / Fsample_{ADC}$$

If computed with the above explicit operator precedence, the resulting over sample rate (OSR_{ADC}) will be a 24 bit scaled fixed point representation of the desired decimation rate. In order to load the **HW_ADCSRR_SR** field, we need to account for the decimators pos_zero comparison. To do this, we subtract one from the whole number portion of the OSRADC. We load the SR bit field with:

HW_ADCSSR_SR = (OSR_{ADC} - $010000)

The value stored in the **HW_ADCSRR_SR** is interpreted as a 24 bit unsigned value.

WARNING: The computation of the scaled fixed point OSR_{ADC} value has an intermediate result precision of 35 bits before the division. These are static values that are hard coded into applications for the nine sample rates of interest. Long Long (64-bit) C variables should be used to avoid errors due to loss of precision.

NOTE: While the OSR calculation, above, uses the desired sample rate, the buffer will actually contain audio samples at 8 times this sample rate. Therefore to record a 44.1KHz audio stream from the ADC, one must accept the 352.8KHz sample stream that is DMAed to on-chip memory. To record it at 44.1KHz, one must run an

8:1 decimation algorithm in the DSP or run three stages of the FILCO's 2:1 decimation FIR filter, see Figure 99. "Stereo Sigma Delta A/D and DMA Buffers" on page 280.

The 1-bit sigma delta A/D converter is always sampled on a submultiple of the 24.0MHz crystal oscillator frequency as specified in the **HW_CCR_ADCDIV** register, see Figure 100. "Variable Rate A/D Converter" on page 286. This divider generates sample strobes at $Fanalog_{ADC}$ where the divisors available come from the set {4,6,8,12,16,24}. It is recommended that ADCDIV always be set to 000 so that a 6.0MHz 1-bit A/D sample rate is used. The sample strobe is used to integrate the 1-bit A/D values. As shown in Figure 100, these integrated values are filtered and then delivered to the ADC DMA to write into on-chip RAM.

Notice that the integrators run continuously while the filters produce samples at the decimated rate. Depending on the decimation or over-sample ratio of the CIC filter engine, the integrators will produce samples of various precisions and scale factors. The filtered values written to RAM are signed 24 bit numbers with the conversion data LSB justified, i.e. down-scaled in the lower end of the 24 bit word. The scale factor column of Table 407, "Values for the HW_ADCSRR_SR register," on page 284 gives the approximate values to multiply the resultant samples to achieve a normalize fixed point scaled fixed point with the fixed point set between the sign bit and the first data bit.

If the variable rate decimator is set to decimate by exactly 16, then this setting causes it to produce samples that are 8X over-sampled with respect to the desired frequency (8x16=128 total over-sample rate). A filter and decimation routine in the DSP is then used to produce stereo or mono audio samples at the desired $Fsample_{ADC}$ for use in various recording applications. With $Fanalog_{ADC}$ at 6.0MHz and an oversample rate of exactly 128 gives us an FsampleADC of 46.875KHz. Thus there are no standard audio stream sample rates that are exactly interpolated with a 24.0MHz crystal.

The 48KHz family of sample rates satisfy the property:

$24.576MHz = Q*Fsample_{ADC}$ where Q comes from the set of integers.

These sample rates include 48KHz, 32KHz, 24KHz, 16KHz, 12KHz and 8KHz. There are also the members of the 44.1KHz family, whose members satisfy the property:

$16.9344MHz = Q*Fsample_{ADC}$ where Q comes from the set of integers.

These sample rates include 44.1KHz, 22.05KHz and 11.025KHz. Since 24.576KHz and 16.9344MHz are relatively prime to 24.0MHz, members of the 48KHz family and 44.1KHz family are related to the 24.0MHz source clock by the relationship $24.0MHz=P*Fsample_{ADC}$, where P is a rational number.

The A/D block includes a variable rate or rational decimator as shown in Figure 100 to accommodate these sample rates. Rational numbers in the ADC are approximated with a scaled fixed point 24 bit value. In this case, the decimal point falls between bit 15 and bit 16. Therefore the lower two bytes hold the fractional part while the upper byte holds the whole number portion of the scaled fixed point. The position register uses this scaled fixed point representation to hold the number of 1-bit samples to be dropped (decimated) to find the next sample at which to produce a filtered multi-bit sigma delta A/D value to send to the DMA. Whenever the whole number part (viz. bits 23:16) are zero, then a sample is produced. For recording at 44.1KHz, we have to set the decimator to skip 17.0068 (P=2500/147) 1-bit samples and produce a filtered multi-bit result. Recall that all 17.0068 1-bit samples are integrated before producing the result. To accomplish this we load **HW_ADCSRR_SR**

with a value of $1001BE or 16 as the whole number portion. Referring to the bottom of Figure 100, at reset the position_reg is loaded with zero so the pos_zero boolean is true. Thus the value $1001BE will be continuously added to the position_reg. After the first addition, pos_zero is no longer true so the value $FF0000 is added to the position-reg. This is effectively a value of minus 1 in the whole number portion so sixteen 1-bit samples will be skipped. On the seventeenth sample, the ADC will produce a 24 bit result and will forward it to the on-chip RAM, via the DMA.

Notice that the lower 16 bits of the **HW_ADCSRR_SR** value will be accumulate into the position register and will eventually overflow into the whole number portion causing an additional sample to be skipped.

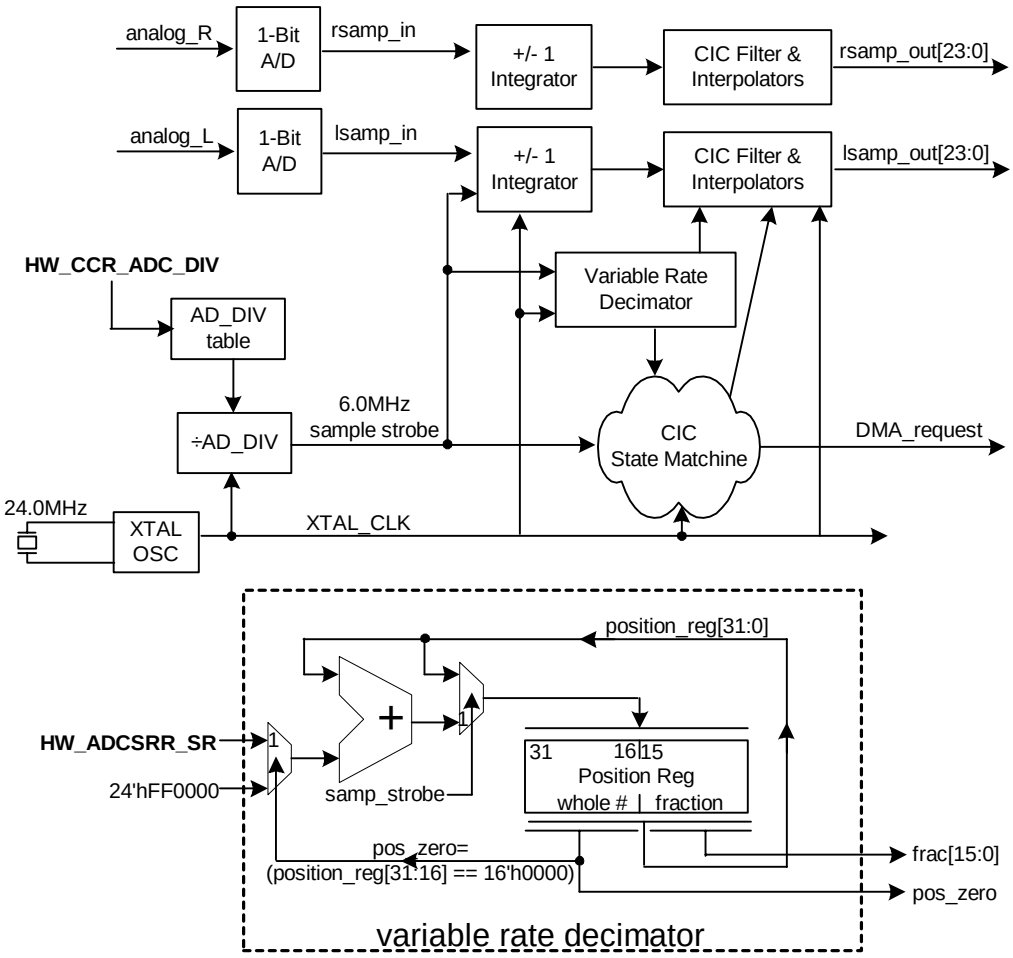


Figure 100. Variable Rate A/D Converter

As shown in Table 392, “Values for the HW_DACSRR registers,” on page 274, at $Fsample_{ADC}$ =44.1KHz, $Fanalog_{ADC}$ is 6.00MHz so the ratio of these two frequencies is 136.0544. This is the over-sample ratio at which the 1-bit A/D over samples the desired sample rate. Since we have set the decimation rate to 1:17.0068 we will see 352.8KSamples/second in the on-chip buffer or 8X over sample of our desired 44.1KHz sample rate. Either DSP software or the FILCO coprocessor is used to low pass filter and decimate this resultant sample stream to 44.1KHz.

The range of values of the samples stored into the on-chip RAM is proportional to the square of the over sample rate (OSR) used in the capture process. The larger the OSR, the longer period the integrators run in the ADC. As result the range of values scene for the same signal wave form captured at the same sample rate but with two different OSR will be different. For example, an 8KHz microphone captured at F_{ADC} = 6.0MHz will be 36 times smaller than the values resulting from capturing the same source signal at F_{ADC} = 1.0MHz. The peak range of values seen in a capture of a signal at 44.1KHz with $Fanalog_{ADC}$ = 6.0MHz is +/-3200 decimal. The oversimple ratio in this case is OSR= 136.054. We can calculate a magnitude constant, K_{filter} for ADC's filter from this as $K_{filter} = OSR^2/\text{Peak Value} = (136.054)^2/3200 = 5.7846$. For any OSR in any sample rate, the peak value can be approximated by $Value_{peak} = OSR^2 / K_{filter}$.

In signal processing, one frequently normalizes the range of values to +/-1.0 as seen in a fixed point scaled integer[1]. For a 24 bit DSP, the fixed point is placed between bit 23 and the sign bit (bit 24) (bit 1 = 2^0). So the desired maximum excursion is then +/- 2^{23} or +/-8388608, decimal. One can calculate a normalization constant to multiply all incoming samples for each sampling condition from the following equation:

$$\text{ScaleFactor} = 2^{23} * K_{filter} / OSR^2$$

If the incoming sample stream is multiplied, sample by sample, by ScaleFactor, then normalized +/-1.0 samples result. The reader may wish to reduce the size of ScaleFactor slightly to allow for out or range samples.

1.A normalized two's complement 24 bit number can not actually express a value of +1.0 without overflowing.

25.1.7. ADC Control Status Register

HW_ADCCSR X:$FB00

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CLKGT												LPFMODE					DMASEL		LPBK	TXEXC	TXI	TXIEN	TXEN

Table 408. HW_ADCCSR

BITS	LABEL	RW	RESET	DEFINITION
23	**CLKGT**	RW	0	**Clock gate –** Used to disable the clocks to the ADC module to conserve power when the ADC is not in use. This bit must be set to 0 before writing to any other ADC registers, and while the ADC is operating. 0 - Clocks enabled 1 - Clocks gated off
22:12	**RSRVD**	R	0	**Reserved –** Must be written with 0.
11:8	**LPFMODE**	RW	0000	**Low Pass Filter Mode –** The ADC implements a hardware low pass filter on the one bit values coming from the 1-bit sigma delta A/D converter. This field controls the low pass filter mode for both the left and the right channel. 0000 - Low pass filtering disabled (3410 compatibility mode) 0001 - input data summed across 4 samples, then divide by 2 0010 - input data summed across 8 samples, then divide by 2 0011 - input data summed across 16 samples, then divide by 2 0100 - input data summed across 1 sample, then divide by 2 0101 - input data summed across 4 samples, then divide by 4 0110 - input data summed across 8 samples, then divide by 4 0111 - input data summed across 16 samples, then divide by 4 1000 - input data summed across 1 sample, then divide by 4 1001 - input data summed across 4 samples, then divide by 8 1010 - input data summed across 8 samples, then divide by 8 1011 - input data summed across 16 samples, then divide by 8 1100 - input data summed across 1 sample, then divide by 8 1101 - input data summed across 4 samples, then divide by 16 1110 - input data summed across 8 samples, then divide by 16 1111 - input data summed across 16 samples, then divide by 16
7	**RSRVD**	R	0	**Reserved –** Must be written with 0.
6:5	**DMASEL**	RW	00	**Memory space** to use for DMA transfers 00 - X space 01 - Y space 10 - P space 11 - Reserved
4	**LPBK**	RW	0	**Loopback –** When set, the LPBK bit connects the DAC single bit data output from the DAC digital circuitry directly to the ADC digital circuitry to perform a digital loopback test without utilizing the analog circuitry.

Table 409. ADC Control Status Register Description

BITS	LABEL	RW	RESET	DEFINITION
3	**TXEXC**	RW	0	**Transmit Exception –** The TXEXC bit is a status bit indicating a transmit exception has occurred. A transmit exception occurs when the ADC needs to write a sample but the HW_ADCWCR is zero indicating no room is available in the module buffer. This register can also be cleared by writing to the HW_ADCWCR register. The TXEXC bit is cleared by writing a 0 to this location.
2	**TXI**	RW	0	**Transmit Interrupt –** The TXI bit is a status bit indicating a transmit interrupt has occurred. This register can also be cleared by writing to the HW_ADCWCR register. The TXI bit is cleared by writing a 0 to this location.
1	**TXIEN**	RW	0	**Transmit Interrupt Enable –** This bit enables interrupt generation for the ADC. When the HW_ADCWCR register reaches zero, an interrupt is generated.
0	**TXEN**	RW	0	**Transmit Enable –** Setting the TXEN bit causes the ADC Port to begin outputting samples into the modulo buffer specified by the HW_ADCBAR, HW_ADCMR, and HW_ADCWCR registers. When TXEN is cleared, the TXI and TXEXC bits are cleared. No other ADC registers are cleared.

Table 409. ADC Control Status Register Description (Continued)

26. MIXER

26.1. Mixer Address Registers

ADDRESS	DESCRIPTION
X:$FA03	Codec/mixer test register
X:$FA04	Mixer master volume register
X:$FA05	Mixer Microphone in volume register
X:$FA06	Mixer Line In volume register
X:$FA07	Mixer FM In volume register
X:$FA08	Mixer DAC In volume register
X:$FA09	Mixer Record Select register
X:$FA0A	Analog ADC gain register
X:$FA0B	Mixer Power down/control stat register
X:$FA1C	Mixer Test Register

Table 410. Mixer Address Registers

26.2. Mixer Block Diagram

The analog level of the signals mixed throughout the chip can be controlled through registers. The block diagram is shown in Figure 102. The microphone channel is a mono source, so its signal is available on both the left and right channels.

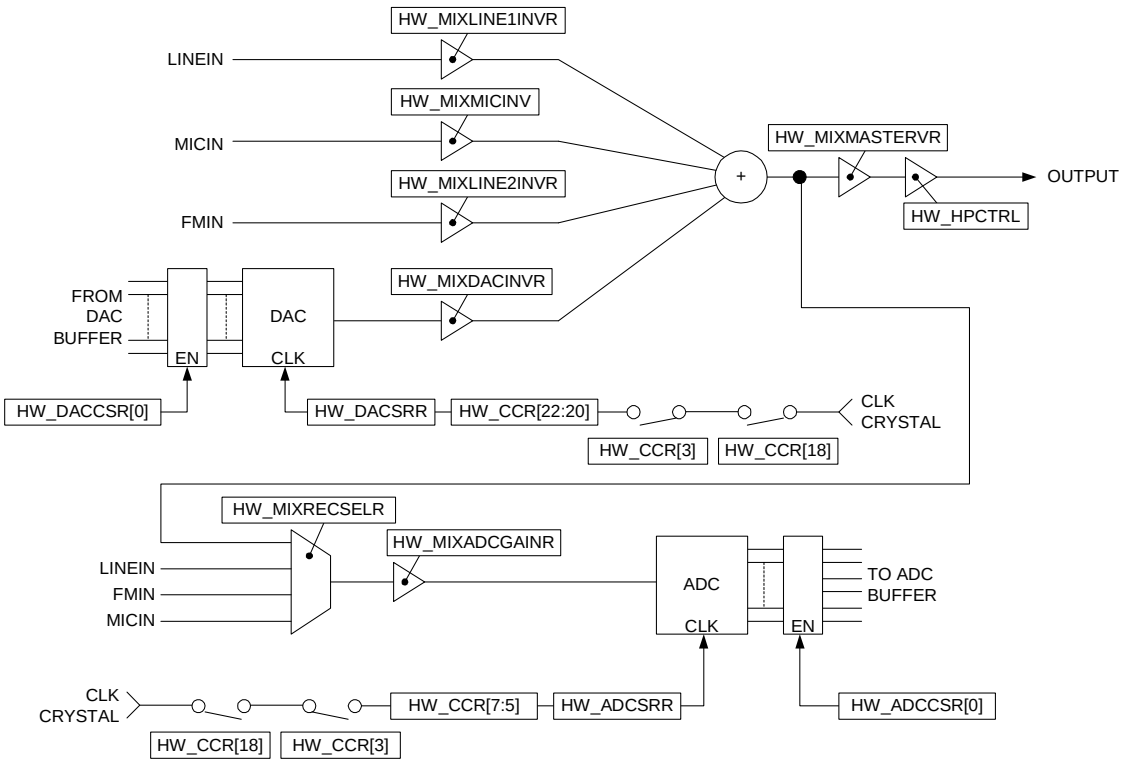


Figure 101. Mixer Flow Diagram

Note that when the mixer is powered down (using the PR2 field of the HW_MIXPWRDNR register), the DAC can still play audio through the headphone amplifier to the output. This mode has the dual advantage of saving the power

consumption of the analog mixer, as well as eliminating the mixer as a source of SNR or THD loss.

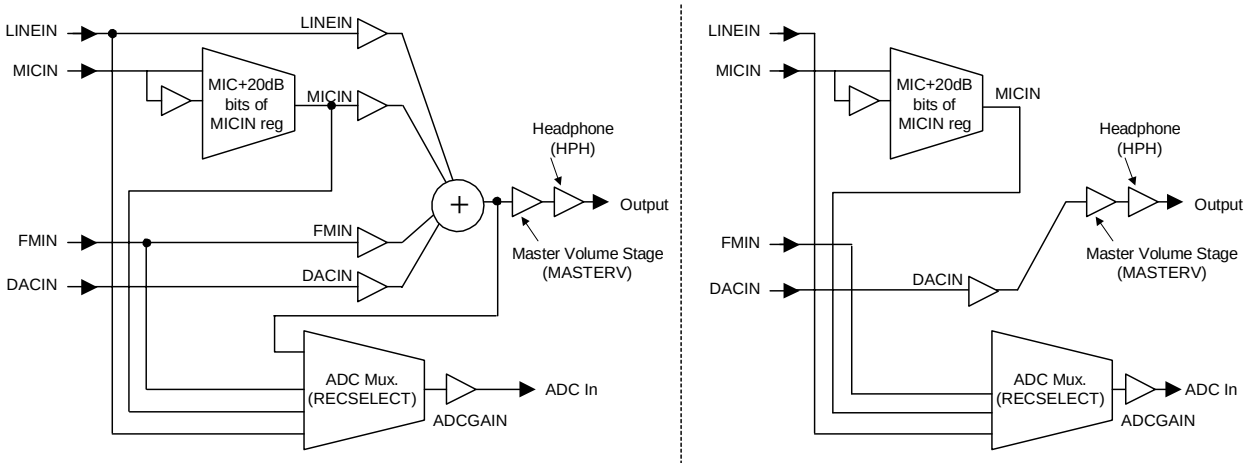


Figure 102. Mixer Block Diagram

26.3. Mixer Programming Model

The various mixer control registers are summarized below.

26.3.1. Mixer Master Volume Register

The ML field adjusts the level for the left channel master output, while the MR field adjusts the level for the right channel master output. Each increment of the ML/MR fields represents 2.0dB[1] of attenuation of the master volume output on the mixer. The Mute bit will silence the output regardless of the current settings of the ML/MR bits.

HW_MIXMASTERVR X:$FA04

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MUTE			ML									MR			

Table 411. HW_MIXMASTERVR

1.The master volume level at a value of **ML** or **MR**=00010 on the STMP35xx corresponds to the same level as a value of **ML** or **MR**=00000 on the STMP3410. NOTE: dB levels are approximate values at each code point.

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**MUTE**	RW	1	**Mixer Master mute control** 0 unmuted 1 muted
14:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:8	**ML**	RW	00010	**Mixer Master volume control for left channel**
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**MR**	RW	00010	**Mixer Master volume control for right channel**

Table 412. Mixer Master Volume Register Description

MUTE	ML/MR	LEVEL
0	0 0000	+6.0 dB
0	1 1111	-56 dB
1	X XXXX	-infinity

Table 413. Mixer Master Volume Register values

26.3.2. Analog Mixer Volume Registers

This section refers to the MicIn, Line-In, Line-In 2 (also know as FM-In), and DAC registers listed below. The GL fields adjust the gain for the left channel of the analog input, while the GR fields adjust the gain for the right channel of the analog input. For the MicIn, there is only one channel adjusted by the GN field, although the results of this volume stage are available on both channels of the mixer. Each increment of the GL/GR/GN fields represents -1.5 dB of adjustment in the gain level. All analog mixer inputs have a mute bit which will silence this analog input regardless of the settings of the GL/GR/GN bits. The MicIn additionally has a 20 dB boost bit which provides a fixed 20 dB boost to the microphone signal when set. Unlike the master volume, each analog mixer input has the ability to apply gain or attenuation to the analog signal.

MUTE	GL/GR/GN	LEVEL
0	0 0000	+12 dB
0	0 1000	0 dB
0	1 1111	-34.5 dB
1	X XXXX	-infinity

Table 414. Mixer Volume Register values

26.3.2.1. Mixer Microphone-In Volume Register

HW_MIXMICINVR X:$FA05

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MUTE									20DB		GN				

Table 415. HW_MIXMICINVR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**MUTE**	RW	1	**Mixer Microphone In mute**
14:7	**RSRVD**	R	0	**Reserved –** Must be written with 0.
6	**20DB**	RW	0	**Mixer Microphone In boost** 0 no boost 1 20 dB boost
5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**GN**	RW	01000	**Mixer Microphone-In volume**

Table 416. Mixer Microphone-In Volume Register Description

The external microphone needs a bias voltage to enable it to operate. This bias voltage can be generated externally using discrete components as shown in Figure 103, or if the LINE1R pin is available, it can be used to supply a bias voltage from an on-chip generator, as shown in Figure 104. To enable the generation of the microphone bias voltage on the LINE1R pin, the **MICBIAS** bits in the **HW_MIXTBR** need to be set to the required value. Also the bias voltage used comes from the LRADC, so the LRADC auxiliary channel must be powered up using the AUXPWD bit in the **HW_LRADC1_CTRL_PWD** register.

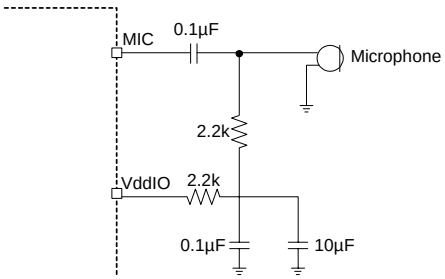


Figure 103. External Microphone Bias Genera-

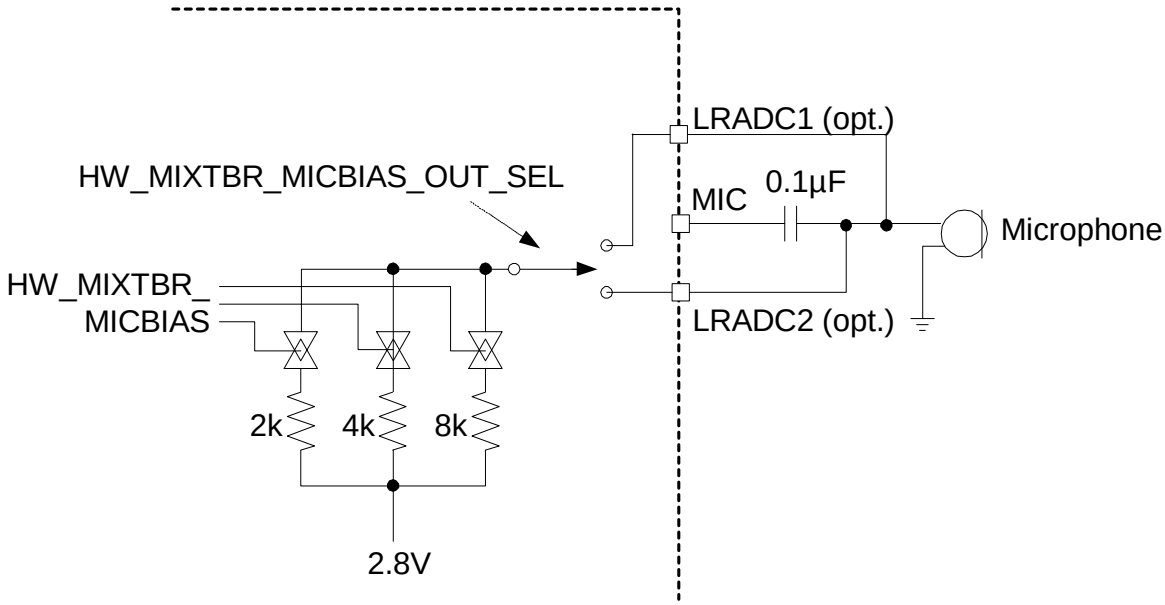


Figure 104. Internal Microphone Bias Generation

26.3.2.2. Mixer Line-In Volume Register

HW_MIXLINE1INVR X:$FA06

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MUTE			GL								GR				

Table 417. HW_MIXLINE1INVR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**MUTE**	RW	1	**Mixer Line-In mute**
14:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:8	**GL**	RW	01000	**Mixer Line-In left channel volume**
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**GR**	RW	01000	**Mixer Line-In right channel volume**

Table 418. Mixer Line-In Volume Register Description

26.3.2.3. Mixer Line-In 2 Volume Register

Line-In 2 is also know as FM-In, although nothing about this input restricts it to only be used with an FM source, it can also be used with other sources. This input channel is only available in 144-pin package versions of the chip.

HW_MIXLINE2INVR X:$FA07

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MUTE			GL								GR				

Table 419. HW_MIXLINE2INVR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**MUTE**	RW	1	**Mixer Line-in 2 mute**
14:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:8	**GL**	RW	01000	**Mixer Line-in 2 left channel volume**
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**GR**	RW	01000	**Mixer Line-in 2 right channel volume**

Table 420. Mixer Line-in 2 Volume Register Description

26.3.2.4. Mixer DAC In Volume Register

HW_MIXDACINVR X:$FA08

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MUTE			GL								GR				

Table 421. HW_MIXDACINVR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**MUTE**	RW	1	**Mixer DAC mute –** This bit is only valid when PR2 in HW_MIXPWRDNR = 0.
14:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12:8	**GL**	RW	01000	**Mixer DAC Left channel volume**
7:5	**RSRVD**	R	0	**Reserved –** Must be written with 0.
4:0	**GR**	RW	01000	**Mixer DAC Right channel volume**

Table 422. Mixer DAC In Volume Register Description

26.3.2.5. Mixer Record Select Register

HW_MIXRECSELR X:$FA09

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
													SL								SR		

Table 423. HW_MIXRECSELR

BITS	LABEL	RW	RESET	DEFINITION
23:12	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10:8	**SL**	RW	000	**Mixer Record Left channel select**
7:3	**RSRVD**	R	0	**Reserved –** Must be written with 0.
2:0	**SR**	RW	000	**Mixer Record Right channel select**

Table 424. Mixer Record Select Register Description

SR	RIGHT RECORD SELECT
000	Mic-in
001	Reserved
010	Reserved
011	Line-in2 (AKA FM-in)
100	Line-in
101	StereoMix
110	Reserved
111	Reserved

SL	LEFT RECORD SELECT
000	Mic-in
001	Reserved
010	Reserved
011	Line-in2 (AKA FM-in)
100	Line-in
101	StereoMix
110	Reserved
111	Reserved

Table 425. Mixer ADC Select Register

26.3.3. Mixer ADC Gain Register

The record gain register provides gain only to the analog signal being input into the ADC. The GL field adjusts the left channel, while the GR field adjusts the right channel. Each increment of these fields represents -1.5 dB of gain. The mute bit will silence the input regardless of the settings of the GL/GR fields.

HW_MIXADCGAINR X:$FA0A

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
								MUTE				GL								GR			

Table 426. HW_MIXADCGAINR

BITS	LABEL	RW	RESET	DEFINITION
23:16	**RSRVD**	R	0	**Reserved –** Must be written with 0.
15	**MUTE**	RW	1	**Mixer ADC Gain mute**
14:12	**RSRVD**	R	0	**Reserved –** Must be written with 0.
11:8	**GL**	RW	0000	**Mixer ADC Gain left channel**
7:4	**RSRVD**	R	0	**Reserved –** Must be written with 0.
3:0	**GR**	RW	0000	**Mixer ADC Gain right channel**

Table 427. Mixer ADC Gain Register Description

MUTE	GL/GR	LEVEL
0	1111	+22.5 dB
0	0000	0 dB
1	XXXX	-infinity

Table 428. Mixer ADC Gain Register

26.3.4. Mixer Power Down Control/Status Register

HW_MIXPWRDNR X:$FA0B

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
												PR2	PR1	PR0									

Table 429. HW_MIXPWRDNR

BITS	LABEL	RW	RESET	DEFINITION
23:12	**RSRVD**	R	0	**Reserved –** Must be written with 0.
11	**PR2**	RW	1	**Power down analog mixer** 0 Analog mixer powered up 1 Analog mixer powered down When the analog mixer is powered down, it is still possible to play audio through the DAC. The reference voltage generator is still powered up, and if the DAC is still powered up and playing audio, the audio is routed directly into the master volume stage, bypassing the analog mixer. Aside from saving power, this mode of operation has the added advantage of providing better SNR/THD performance.
10	**PR1**	RW	1	**Power down DAC analog circuitry** 0 DAC & input mux powered up 1 DAC & input mux powered down
9	**PR0**	RW	1	**Power down ADC & input mux analog circuitry** 0 ADC & input mux powered up 1 ADC & input mux powered down
8:0	**RSRVD**	R	0	**Reserved –** Must be written with 0.

Table 430. Mixer Power Down Control/Stat Register Description

BIT	FUNCTION
PR0	PCM in ADCs and Input Mux Powerdown
PR1	PCM out DACs Powerdown
PR2	Analog Mixer Powerdown (Vref still on)

(Please see Figure 102.)

Table 431. Mixer Power Down Register

26.3.5. Codec/Mixer Test Register

The Codec/Mixer test register contains some bits that control functionality that was designed as options, and that was not expected to be used. It also contains control bits for functions that are available for general use. Please pay careful attention to the notes in each bit field which indicate whether that field is available for general usage. If it is marked unavailable then be certain to only write its reset value when modifying other bits in the register. The programmer shouldn't change the unavailable bit fields in this register without being directed to do so by SigmaTel, as undesired operation may result. The organization of the Codec/Mixer Test register is shown below.

HW_MIXTBR X:$FA03

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DZCMA	DZCMI	DZCLI	DZCFM	DZCDA	EZD	MICBIAS1	PWDADC	MICBIAS0	ADTHD	XBGC	XBCO	VCOS	FX2	PSRN	MIC_BIAS_OUT_SEL	DCKI	PCPCD	PCPCU	ASD2X	ACKI	HOLD_GND	PLL_ZERO_VT	

Table 432. HW_MIXTBR

BITS	LABEL	RW	RESET	DEFINITION
23	**DZCMA**	RW	0	**Enable zero crossing volume update for master volume stage.** This bit field is available for general usage.
22	**DZCMI**	RW	0	**Enable zero crossing volume update for microphone volume stage.** This bit field is available for general usage.
21	**DZCLI**	RW	0	**Enable zero crossing volume update for line-in volume stage.** This bit field is available for general usage.
20	**DZCFM**	RW	0	**Enable zero crossing volume update for line-in 2 (also known as FM-in) volume stage.** This bit field is available for general usage.
19	**DZCDA**	RW	0	**Enable zero crossing volume update for DAC volume stage –** (Use only when mixer is powered up, HW_MIXPWRDNR PR2 = 0. DAC zero crossing volume updates are done digitally when the mixer is powered down, HW_MIXPWRDNR PR2 = 1)**.** This bit field is available for general usage.
18	**EZD**	RW	0	**Enable zero crossing detect for the volume stage determined by bits 23-19 –** Note that one and only one of the bits 23-19 must be set high for the zero crossing detect to properly function. Also, there is a 10Hz timeout used, so if a zero crossing is not detected within a 10Hz period, the volume is automatically updated. Only one bit in the range 23-19 can be enable at a time for zero crossing detection. This bit field is available for general usage.
17	**MICBIAS1**	RW	0	SEE MICBIAS field below. **WARNING** this is a split field with **PWDADC** in the middle.
16	**PWDADC**	RW	0	**Power down ADC right channel –** This powers down one ADC channel when the ADC is being used to record only a mono channel. When set, some power will be saved in the analog portion of the ADC. The ADC will still generate stereo data, however only the samples for the left channel will be valid. This bit reserved for use only by SigmaTel, always set it to zero.

Table 433. Codec/Mixer Test Register Description

BITS	LABEL	RW	RESET	DEFINITION
17,15	**MICBIAS**	RW	00	**Microphone resistor select –** Provides an option for reducing board complexity by integrating some bias circuitry for the microphone. This circuit provides a regulated supply connected via a selectable resistor value to the line-in right pin. This allows users to connect the microphone to the LRADC1 or LRADC2 pins and then externally place a capacitor to the microphone input to implement the microphone function. This functionality cannot be used at the same time that the selected LRADC input is needed. See the microphone volume register for a figure detailing the circuit. **WARNING** this two bit field is split by **PWDADC** in bit 16. 00 Mic bias off 10 4kohm 01 2kohm 11 8kohm
14	**ADTHD**	RW	0	**Turn off ADC dither.** This bit is reserved for use only by SigmaTel, always set it to zero.
13	**XBGC**	RW	0	**Causes the xtal oscillator to use the band gap bias current, instead of its self-generated bias current.** This bit is reserved for use only by SigmaTel, always set it to zero.
12	**XBCO**	RW	0	**Turns off xtal internal bias current.** This bit is reserved for use only by SigmaTel, always set it to zero.
11	**VCOS**	RW	0	**Speed up PLL VCO.** This bit is reserved for use only by SigmaTel, always set it to zero.
10	**FX2**	RW	0	**Double frequency reference to the PLL.** This bit is reserved for use only by SigmaTel, always set it to zero.
9	**PSRN**	RW	0	**Disable fast falling edge power down function of PSWITCH pin.** This bit is reserved for use only by SigmaTel, always set it to zero.
8	**MIC_BIAS**	RW	0	**Enable Mic Bias.** This bit controls the mic bias source selection. 0 - mic bias from LRADC1 1 - mic bias from LRADC2
7	**DCKI**	RW	0	**Invert DAC clock.** This bit is reserved for use only by SigmaTel, always set it to zero.
6	**PCPCD**	RW	0	**Decrease PLL charge pump current 2X.** This bit is reserved for use only by SigmaTel, always set it to zero.
5	**PCPCU**	RW	0	**Increase PLL charge pump current 2X.** This bit is reserved for use only by SigmaTel, always set it to zero.
4	**ASD2X**	RW	0	**Slow ADC dither.** This bit is reserved for use only by SigmaTel, always set it to zero.
3	**ACKI**	RW	0	**Invert ADC clock.** This bit is reserved for use only by SigmaTel, always set it to zero.
2	**HOLD_GND**	RW	0	**Hold at Ground.** This bit can be used to reduce “pops” at startup. It causes the headphone output to be held at ground.
1	**PLL_ZERO_VT**	RW	0	**Select Zero Vt Devices.** This bit controls the special PLL zero V_T mode. 0 - PLL Normal V_T 1 - PLL Zero V_T
0	**RSRVD**	R	0	**Reserved –** Must be written with 0.

Table 433. Codec/Mixer Test Register Description (Continued)

26.3.6. Reference Control Register

HW_REF_CTRL X:$FA19

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
				LOW_PWR	LWREF	BIASC		PWRDWNS	ADJADC	ADCREFV				ADJV	VAGVAL				ADJDAC	DACVBGVAL			

Table 434. HW_REF_CTRL

BITS	LABEL	RW	RESET	DEFINITION
23:20	**RSRVD**	R	0	**Reserved –** Must be written with 0.
19	**LOW_PWR**	RW	0	Lowers power in the band gap amplifier. This mode is useful in USB suspend or standby when absolute band gap accuracy is not critical.
18	**LWREF**	RW	0	lower voltage in 13:10, 8:5, 3:0 by 20%
17:16	**BIASC**	RW	0	**Bias current control** 00 nominal 10 -10% 01 -20% 11 +10% These bits control the bias currents sent to all the analog circuits from the band gap generator.
15	**PWRDWNS**	RW	0	**Powers down selfbias circuit –** The reference uses a self bias circuit during powerup that can be turned off with this bit. However, care must be taken that the DC-DC converter must be using the bandgap-generated current before the selfbias is powered down (bit [15] in $fa14).
14	**ADJADC**	RW	0	**Adjust ADC reference** using bits [13:10], default 1.5V derived from the bandgap
13:10	**ADCREFV**	RW	0	**ADC reference value** 1111 1.600V 1001 1.450V 0100 1.325V 1110 1.575V 1000 1.425V 0011 1.300V 1101 1.550V 0111 1.400V 0010 1.275V 1100 1.525V 0110 1.375V 0001 1.250V 1011 1.500V 0101 1.350V 0000 1.225V 1010 1.475V These bits adjust the value of the reference used by the ADC, and can be used to allow acceptable analog performance at low supply voltages. Voltages shown can be lowered 20% lower by setting bit 18.
9	**ADJV**	RW	0	**Adjust Vag –** Default is a resistor divider from the analog power supply. These bits adjust the value of the analog reference used throughout the codec. This value should be programmed to be near half of the analog target supply voltage.
8:5	**VAGVAL**	RW	0	**Vag Value** 1111 1.000V 1001 0.850V 0100 0.725V 1110 0.975V 1000 0.825V 0011 0.700V 1101 0.950V 0111 0.800V 0010 0.675V 1100 0.925V 0110 0.775V 0001 0.650V 1011 0.900V 0101 0.750V 0000 0.625V 1010 0.875V

Table 435. Reference Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
				Voltages shown can be lowered 20% lower by setting bit 18.
4	**ADJDAC**	RW	0	**Adjust DAC reference value –** Default is the band gap voltage. These bits adjust the value of the reference used by the d/a converters, and can be used to allow acceptable analog performance at low supply voltages.
3:0	**DACVBGVAL**	RW	0000	**DAC Vbg value** 1111 1.300V 1001 1.150V 0011 1.000V 1110 1.275V 1000 1.125V 0010 0.975V 1101 1.250V 0111 1.100V 0001 0.950V 1100 1.225V 0110 1.075V 0000 0.925V 1011 1.200V 0101 1.050V 1010 1.175V 0100 1.025V Voltages shown can be lowered 20% lower by setting bit 18.

Table 435. Reference Control Register Description (Continued)

26.3.7. MIXER TEST REGISTER

HW_MIX_TEST X:$FA1C

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											TMPPWD	TMP_CFG						DAC_MORE_AMP_I	DAC_DISABLE_RTZ			DAC_CHOP_CLK	

Table 436. HW_MIX_TEST

BITS	LABEL	RW	RESET	DEFINITION
23:12	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12	**TMPPWD**	RW	1	Set to one to power down the temperature sensor current source. This current source is only available on LRADC2.

11:8	**TMP_CFG**	RW	0000	Each bit turns on a separate current source and the source currents are additive.
				`0001 = 20uA`
				`0010 = 40uA`
				`0100 = 80uA`
				`1000 = 160uA`
				`1111 = 300uA`
				NOTE: set **HW_USBPHYPWD_PWDIBIAS** to zero and set **HW_REF_CTRL_LWREF** to zero for proper operation.

7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5	**DAC_MORE_AMP_I**	RW	0	Set to one to increase the current in the DAC I amplifier
4	**DAC_DISABLE_RTZ**	RW	0	Set to one to disable DAC RTZ mode. Set to zero for normal operation with RTZ enabled.

Table 437. Mixer Test Register Description

BITS	LABEL	RW	RESET	DEFINITION
3:2	**RSRVD**	R	00	**Reserved –** Must be written with 0.
1:0	**DAC_CHOP_CLK**	RW	00	Enable a clock that may improve signal-to-noise performance by chopping the DAC opamp input. 00 - chop clock disabled 01 - 384KHz chop clock 10 - 192KHz chop clock 11 - 96KHz chop clock

Table 437. Mixer Test Register Description

27. HEADPHONE DRIVER

The headphone driver is designed to directly drive low impedance (16 ohm) headphones. It has built in pop-suppression circuits. It can automatically detect a headphone short situation and report it to the DSP via maskable or non-maskable interrupts.

The STMP35xx includes a common mode amplifier which can present a headphone common node, eliminating the need for large expensive DC blocking capacitors in the headphone circuit.

27.1. Headphone Control Register

The organization of the Headphone Control Register is shown below.

HW_HPCTRL X:$FA15

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SHORT_CM	SHORT_LR	CHOP_CLK			SHORT_LVLADJ			SHORTMODE_CM		SHORTMODE_LR			CAPLESS	CLASSAB	HPHPWD		POP2	POP1	POP0	TESTI1DWNP	TESTI1UP	TESTIALLUP	TESTIALLDWN

Table 438. HW_DCDC1_CTRL0

BITS	LABEL	RW	RESET	DEFINITION
23	**SHORT_CM**	R	0	Status of common mode amplifier short detection. 0 = no short detected 1 = short present To clear this interrupt, set **HW_HPCTRL_SHORTMODE_CM** to 00. This has the effect of resetting the latch and holding it in reset. If **HW_HPCTRL_SHORTMODE_CM** is changed back to 10 or 11 and the short still exists then interrupts will continue on IRQA, see Figure 107. “Stereo Headphone Common Short Detection & Powerdown Circuit” on page 306.
22	**SHORT_LR**	R	0	Status of right/left differential amplifier short detection. 0 = no short detected 1 = short present To clear this interrupt, set **HW_HPCTRL_SHORTMODE_LR** to 00. This has the effect of resetting the latch and holding it in reset. If **HW_HPCTRL_SHORTMODE_LR** is changed back to 10 or 11 and the short still exists then interrupts will continue on IRQA.
21:20	**CHOP_CLK**	RW	00	Enable a chop clock to improve signal-to-noise performance by chopping the head phone amplifier input. 00 = chop clock disabled 01 = 48KHz chop clock 10 = 96KHz chop clock 11 = 192KHz chop clock
19	**RSRVD**	R	0	**Reserved –** Must be written with 0.

Table 439. Headphone Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
18:16	**SHORT_LVLADJ**	RW	000	Adjusts headphone current short detect trip point 000 = Nominal 001 = Nominal * 0.75 010 = Nominal * 0.5 011 = Nominal * 0.25 100 = Nominal * 1.25 101 = Nominal * 1.5 110 = Nominal * 1.75 111 = Nominal * 2 Increasing trip point current makes the detection less sensitive, while decreasing the trip point current makes it more sensitive.
15:14	**SHORTMODE_CM**	RW	00	Headphone Common Amplifier Short Control Mode 00 - SW view direct short, reset latched short, HW pwdn on direct short 01 - SW view latched short HW pwdn on latched short 10 - sw view direct short signal, HW pwdn on direct short 11 - sw view direct short signal, HW pwdn disabled The state of the common mode short detector, or its latched version can be seen in **HW_HPCTRL_SHORT_CM**.
13:12	**SHORTMODE_LR**	RW	$00	Headphone L/R Short Control Mode 00 - short function held in reset 01 - sw view latched short signal, HW pwd on latched signal 10 - sw view latched short signal, HW pwd disabled 11 - sw view direct short signal, HW pwd disabled The state of the left/right mode short detector, or its latched version can be seen in **HW_HPCTRL_SHORT_LR**.
11	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10	**CAPLESS**	RW	0	**Capless –** Set to one to enable the direct drive or “cap-less” headphone mode. Set to zero for conventional operation.
9	**CLASSAB**	RW	0	**ClassAB –** ClassA mode is intended only for powerup anti-pop. ClassAB mode should be set before a signal is applied. ClassAB mode should be used to drive both a low impedance (e.g. headphone) or high impedance (e.g. line-in to another device) load.
8	**HPHPWD**	RW	1	**Headphone Powerdown** 0 Headphone amplifier powered up 1 Headphone amplifier powered down (reset value)
7	**RSRVD**	R	0	**Reserved –** Must be written with 0.
6	**POP2**	RW	0	pop2
5	**POP1**	RW	0	pop1
4	**POP0**	RW	0	pop0

hphpwd [bit 10]	classab [bit 7]	pop2 [bit 6]	pop1 [bit 5]	pop0 [bit 4]	ramp result	expected output pop
0	0	0	0	0	1*84uA PMOS source current	1.3mV
0	0	0	0	1	2*84uA PMOS source current	1.3mV
0	0	0	1	0	3*84uA PMOS source current	1.3mV
0	0	1	1	1	8*84uA PMOS source current	1.3mV
1	x	0	0	0	11.0k pull-down resistor	1.3mV
1	x	0	0	1	5.6k pull-down resistor	depends on timing
1	x	0	1	0	3.7k pull-down resistor	depends on timing
1	x	0	1	1	2.8k pull-down resistor	depends on timing
1	x	1	0	0	1.9k pull-down resistor	depends on timing
1	x	1	0	1	1.2k pull-down resistor	depends on timing
1	x	1	1	0	0.8k pull-down resistor	depends on timing
1	x	1	1	1	0.4k pull-down resistor	depends on timing

3	**TESTI1DWN**	RW	0	**test i1down**

Table 439. Headphone Control Register Description (Continued)

BITS	LABEL	RW	RESET	DEFINITION
2	**TESTI1UP**	RW	0	**test i1up** **I1up** **I1dn** **Change to 1st stage current** 0 0 nominal 0 1 -50% 1 0 +33.3% 1 1 -16.7%
1 0	**TESTIALLUP** **TESTIALLDWN**	RW RW	0 0	**test iall up** **test iall down** **Iallup** **Ialldown** **Change to 1st stage current** 0 0 nominal 0 1 -25% 1 0 +100% 1 1 +20%

Table 439. Headphone Control Register Description (Continued)

27.2. Headphone Driver

The STM35xx supports a conventional stereo headphone drive as shown in Figure 105.

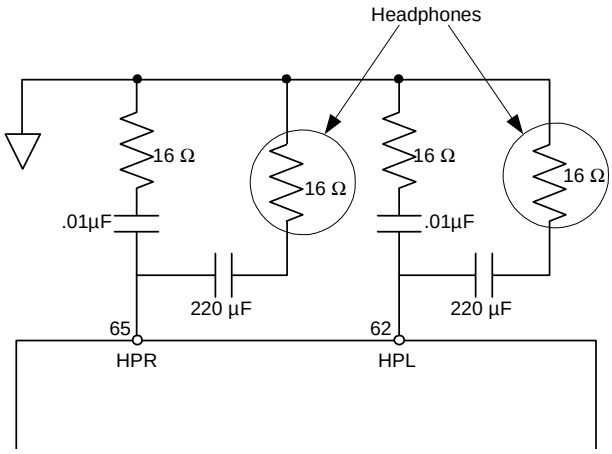


Figure 105. Conventional Stereo Headphone Application Circuit

In addition, the chip can generate an optional headphone common node circuit for the headphones which eliminates the need for the large and expensive DC blocking capacitors. It also improves the anti-pop performance. These benefits are obtained at a slight increase in power consumptions, i.e. at 30 mV rms output, the resultant increase in power consumption is approximately 2.7 mW.

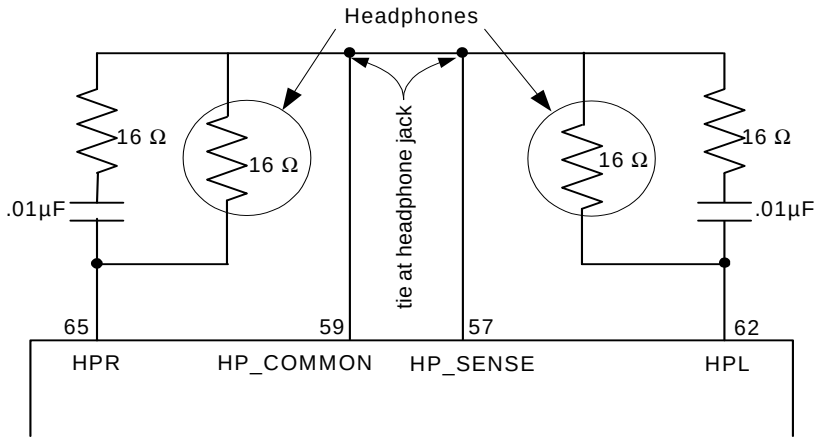


Figure 106. Stereo Headphone Application Circuit with Common Node

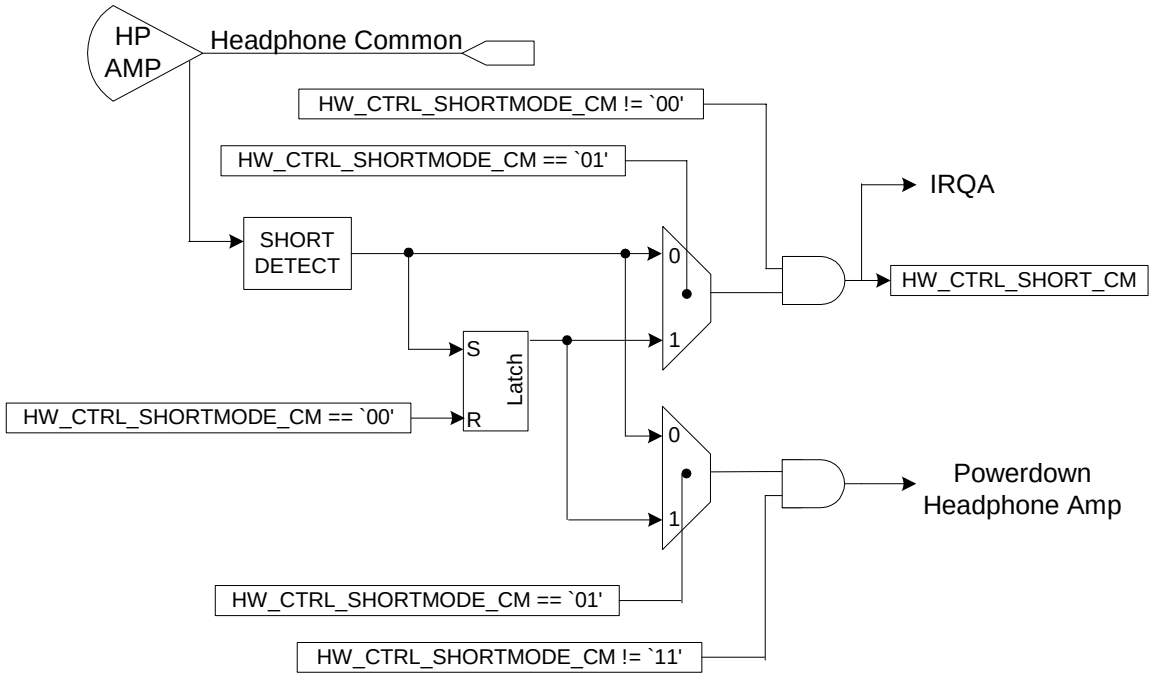


Figure 107. Stereo Headphone Common Short Detection & Powerdown Circuit

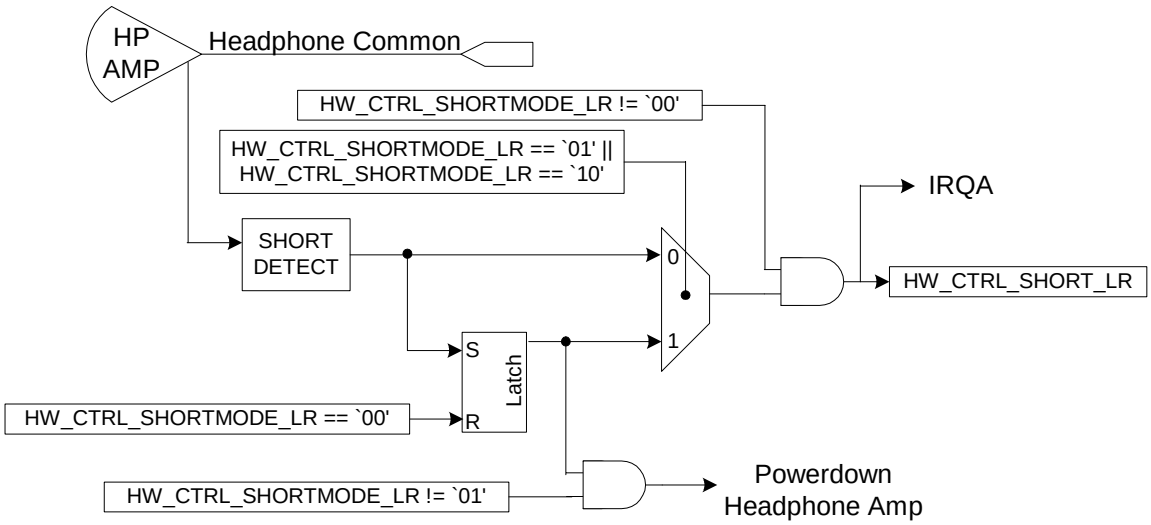


Figure 108. Stereo Headphone L/R Short Detection & Powerdown Circuit

28. LOW RESOLUTION ADC

The low resolution ADC (LRADC) block is used for voltage measurement. Three channels are available. One channel is dedicated to measure the voltage on the BATT pin. The other channels, LRADC1 and LRADC2, measure the voltage on the two application dependent LRADC pins. The battery channel can be used to sense the amount of battery life remaining. The auxiliary channels can be used for a variety of different uses, including a resistor based wired remote control, temperature sensing, etc. The LRADC is accurate to 8 bits of resolution, and samples at a divided rate from the 24.0MHz crystal clock

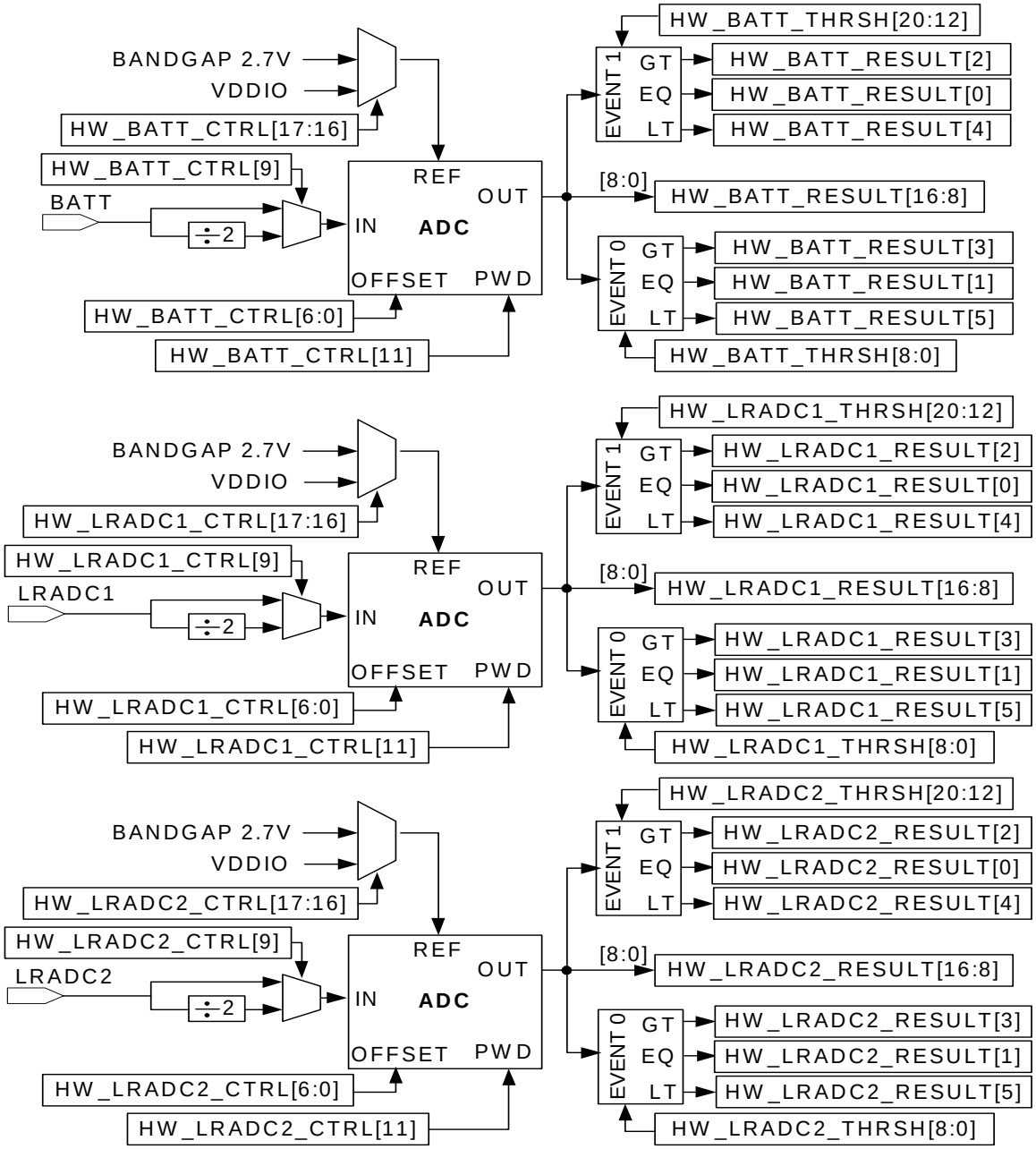


Figure 109. Low Resolution ADC

As shown in Figure 109, the LRADC module consists of three essentially identical analog to digital converters. These converters run in the crystal clock domain for sample interval precision. Programmable registers for each of the three converters are declared as **HW_BATT_***, **HW_LRADC1_*** and **HW_LRADC2_***. In the discussion below, the battery LRADC (**HW_BATT_***) is used to describe the general operation of all three converters.

Consider the battery LRADC channel, the converter can take its voltage reference either from the bandgap or I/O voltage rail (VddIO). Thus high voltage LiIon batteries can be monitored with the appropriate settings of **HW_BATT_CTRL_REF_VAL**. Once an eight bit A/D conversion is made, a seven bit digital offset can be added to it to slide the eight bit value up and down within a nine bit value. Thus the converter presents an eight bit precision result in a nine bit range. The offset can be used to calibrate an LRADC instance if desired.

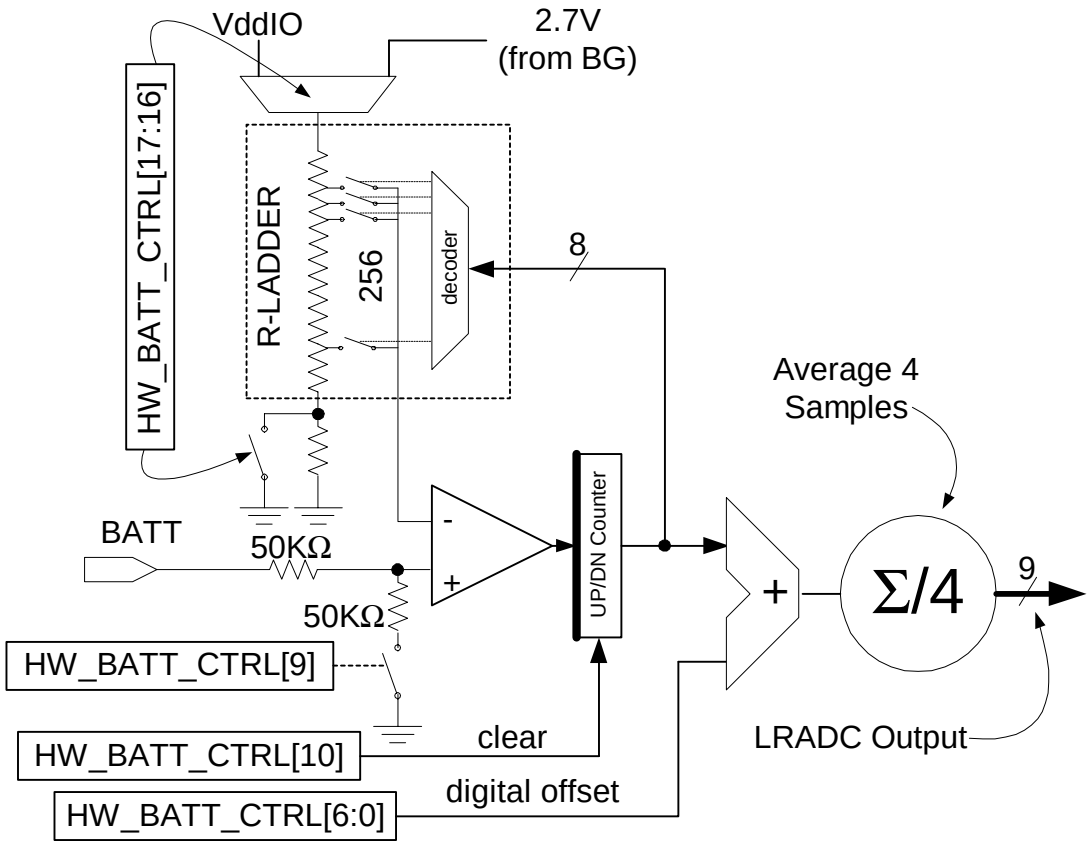


Figure 110. Low Resolution ADC Detail

The nine bit result is compared against two independent threshold values, Event Detector 0 and Event Detector 1. Each event detector reports boolean less than, greater than and equal bits which can be seen directly in the result register. These event booleans can be used to trigger DSP interrupts as discussed below. The current value of the LRADC conversion can be read in **HW_BATT_RESULT_DATA_OUT.**

WARNING, the conversion happens in the crystal clock domain while the result register is read by the DSP in the dclk domain. These clock domains are asynchronous when the DSP dclk is derived from a PLL. The clock domain crossing can lead to uncertainty in the correct reading of individual bits when the conversion value

changes. As a result, software should read this register repeatedly until two successive readings are identical.

Figure 110 shows additional detail of the LRADC itself. Notice that the bias resistor at the bottom of the resistive ladder can be bypassed by **HW_BATT_CTRL_REF_VAL**. This has the effect of shifting the R-Ladder taps down in voltage and making their step sizes larger. When the bias resistor is enabled, one can place a calibration offset into **HW_BATT_CTRL_INPUT_OFFSET** to account for the voltage dropped across the bias resistor. Also notice that the input voltage can be divided by two by setting **HW_BATT_CTRL_INPUT_DIV2** to one. This has the side effect of changing the input resistance from essentially infinite to 100KΩ. The following table summarizes the input ranges and step size resolution for various settings of **REF_VAL** and **INPUT_DIV2**.

REF_VAL	INPUT_DIV2	INPUT RANGE	RESOLUTION	PROG. OFFSET	BATTERY
11	1	2*VDDIO - 0V	VDDIO/128	0x00	
11	0	VDDIO - 0V	VDDIO/256	0x00	
10	1	5.12V - 0V	0.020V	0x00	Quad NiMH (4.8V - 3.6V) LiIon (4.4V - 2.6V)
10	0	2.56V - 0V	0.010V	0x00	
01	1	5.2V - 1.256V	This mode is not recommended		
01	0	2.6V - 0.628V	This mode is not recommended		
00	1	5.4V - 1.304V	0.016V	0x51	Double Cell (3.4V - 1.5V)
00	0	2.7V - 0.652V	0.008V	0x51	Single Cell (1.7V - 0.8V)

Table 440. LRADC Input Ranges and Resolutions

Note: 1. Do not exceed Maximum Pin voltage on any input pin, see Section 3. “CHARACTERISTICS/SPECIFICATIONS” on page 21.

***WARNING**: The pad ESD protection limits maximum voltage on all LRADC inputs. The BATT LRADC is specifically designed to handle higher voltages, but LRADC1 and LRADC2 inputs are limited to 3.3V.*

Two other controls of interest are **HW_BATT_CTRL_CLEAR** which holds the up/down counter at zero and **HW_BATT_CTRL_PWD** which powers down the comparator and the resistive ladder.

A linear convergence A/D architecture was selected for cost and power reasons while maintaining sufficient slew rate performance for brownout detection. This allows the DSP enough time to perform a graceful shutdown. To reduce the linear convergence tracking noise, four successive samples are averaged in the hardware to produce the final A/D value that is used by the event detectors.

Figure 111 shows the generation of an LRADC interrupt from event detector 0 of the battery LRADC. The interrupt request flip flop can be seen by DSP software in **HW_BATT_RESULT_IRQ_EVENT0** (**HW_BATT_RESULT**[20]). This bit is “sticky” in that once it is set it remains set until cleared by software. The polarity of the event trigger comparison is controlled by **HW_BATT_CTRL_POLARITY_EVENT0**. The event 0 interrupt is enabled by **HW_BATT_CTRL_IRQ_EN_EVENT0**. Notice that the equality detector is not available as an interrupt source.

Because of the four sample averaging, there is a slight hysteresis effect in the event detectors,

While the above discussion has focused on the Battery LRADC, the operation of LRADC1 and LRADC2 are essentially the same.

WARNING: The interrupt requests (IRQ) to the DSP is generated somewhat differently in the low resolution ADC, as shown in Figure 111. The IRQ comes from the unlatched outputs of the event detectors, not from the latched status bit. The status bit is only a sticky sample of a signal that could come and go many times, i.e. it is possible for the DSP interrupt to be asserted and then removed before the DSP can determine the actual interrupt source. It is also possible for the DSP to be interrupted many times even after the sticky bit is set.

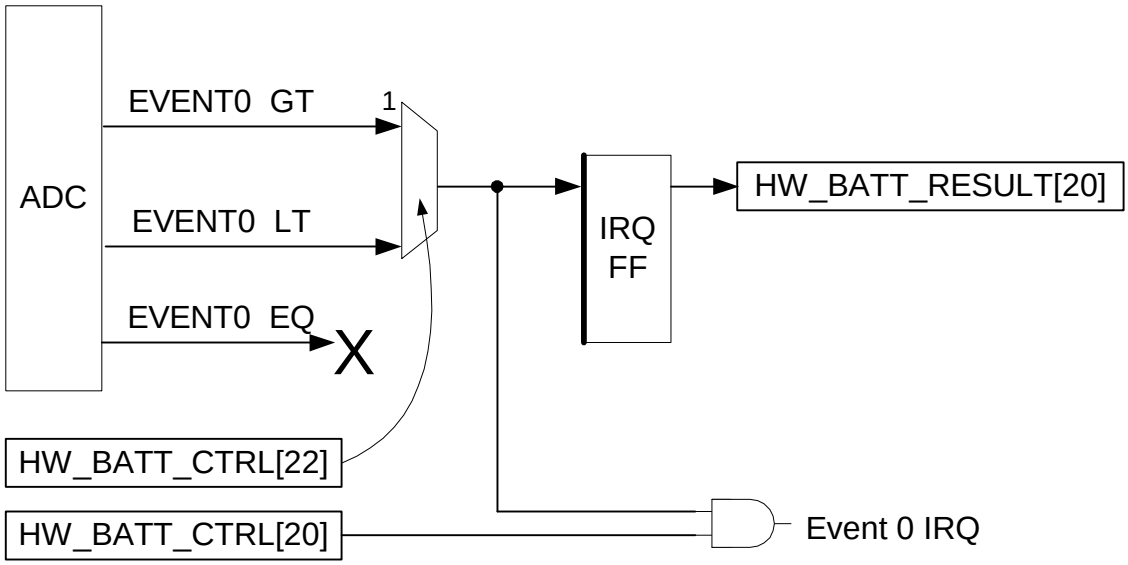


Figure 111. Low Resolution ADC Interrupt Generation

28.1. Low Resolution ADC Programmable Registers

The following programmable registers are available to DSP software for controlling and using the low resolution analog to digital converters.

28.1.1. LRADC Battery Control Register

This register controls the overall operation of the battery LRADC.

HW_BATT_CTRL X:$FA20

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
POLARITY_EVENT1	POLARITY_EVENT0	IRQ_EN_EVENT1	IRQ_EN_EVENT0			REF_VAL				CLK_DIV		PWD	CLEAR	INPUT_DIV2	HALF_CMP_PWR		INPUT_OFFSET						

Table 441. HW_BATT_CTRL

BITS	LABEL	RW	RESET	DEFINITION
23	**POLARITY_EVENT1**	RW	0	Set to one to trigger event 1 interrupts when filtered A/D value is greater than the event 1 threshold. Set to zero for interrupt when the filtered A/D value is less than the event 1 threshold.
22	**POLARITY_EVENT0**	RW	0	Set to one to trigger event 0 interrupts when filtered A/D value is greater than the event 0 threshold. Set to zero for interrupt when the filtered A/D value is less than the event 0 threshold.
21	**IRQ_EN_EVENT1**	RW	0	Set to one to enable an interrupt from the event 1 comparator.
20	**IRQ_EN_EVENT0**	RW	0	Set to one to enable an interrupt from the event 0 comparator.
21:18	**RSRVD**	R	00	**Reserved –** Must be written with 0.
17:16	**REF_VAL**	RW	00$	Reference voltage selection, see Table 440. “LRADC Input Ranges and Resolutions” on page 310. These are the primary mode control bits. They affect the range and resolution of the converter.
15:14	**RSRVD**	R	00	**Reserved –** Must be written with 0.
13:12	**CLK_DIV**	RW	00	Clock divider selection. 00 - divide by 4 01 - divide by 8 10 - divide by 16 11 - divide by 32
11	**PWD**	RW	1	Set this bit to one to power down the comparator, the R-Ladder and stop the internal clocks to minimize power consumption. Set to zero for normal operation.
10	**CLEAR**	RW	1	Set this bit to one to force the up/down counter to all zeroes. Set to zero for normal operation.
9	**INPUT_DIV2**	RW	0	Set to one to divide the input voltage by two before applying it to the comparator. The division by two changes the input resistance from essentially infinite to 100KΩ.
8	**HALF_CMP_PWR**	RW	0	Set to one to reduce the analog current for low power operation. This is accomplished at the expense of conversion performance. This bit should only be set to one if static measurements are needed.

Table 442. LRADC Battery Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
7	**RSRVD**	R	0	**Reserved –** Must be written with 0.
6:0	**INPUT_OFFSET**	RW	$00	These bits are added to the final converter output to add a digital offset to the converted value. This may be useful for LRADC calibration and to compensate for additional bias resistor when it is switched into the R-Ladder.

Table 442. LRADC Battery Control Register Description (Continued)

28.1.2. LRADC Battery Threshold Register

This register sets the thresholds used by the digital comparators in the battery LRADC.

HW_BATT_THRSH X:$FA21

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			THRESHOLD_EVENT1												THRESHOLD_EVENT0								

Table 443. HW_BATT_THRSH

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	000	**Reserved –** Must be written with 0.
20:12	**THRESHOLD_EVENT1**	RW	$000	This nine bit field defines the set point for event detector one's threshold comparator. The nine bit post offset conversion value is compared to this threshold.
11:9	**RSRVD**	R	000	**Reserved –** Must be written with 0.
8:0	**THRESHOLD_EVENT0**	RW	$000	This nine bit field defines the set point for event detector zero's threshold comparator. The nine bit post offset conversion value is compared to this threshold.

Table 444. LRADC Battery Threshold Register Description

28.1.3. LRADC Battery Result Register

This register controls the interrupts reported by the battery LRADC and provides read only views of the LRADC data value and the results of the event detector threshold comparators.

HW_BATT_RESULT X:$FA22

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	IRQ_EVENT1	IRQ_EVENT0					DATA_OUT											LT_EVENT0	LT_EVENT1	GT_ENVET0	GT_EVENT1	EQ_EVENT0	EQ_EVENT1

Table 445. HW_BATT_RESULT

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	00	**Reserved –** Must be written with 0.
21	**IRQ_EVENT1**	RW	0	Interrupt request status for event detector one. This sticky bit is set whenever event detector one's comparator trips the set point. See **HW_BATT_CTRL_POLARITY_EVENT1** and **HW_BATT_THRSH_THRESHOLD1**. Software clears this bit by writing a one to it.
20	**IRQ_EVENT0**	RW	0	Interrupt request status for event detector zero. This sticky bit is set whenever event detector zero's comparator trips the set point. See **HW_BATT_CTRL_POLARITY_EVENT0** and **HW_BATT_THRSH_THRESHOLD0**. Software clears this bit by writing a one to it.
19:17	**RSRVD**	R	00	**Reserved –** Must be written with 0.
16:8	**DATA_OUT**	R	$000	The current nine bit value of the BATT LRADC is read here. ***WARNING:*** due to clock domain crossings, software must repeatedly read this value until two successive values are read which match bit for bit.
7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5	**LT_EVENT0**	R	0	This read only bit indicates the current state of the event detector 0 Less Than result.
4	**LT_EVENT1**	R	0	This read only bit indicates the current state of the event detector 1Less Than result.
3	**GT_EVENT0**	R	0	This read only bit indicates the current state of the event detector 0 Greater Than result.
2	**GT_EVENT1**	R	0	This read only bit indicates the current state of the event detector 1 Greater Than result.
1	**EQ_EVENT0**	R	0	The "excluded muddle".
0	**EQ_EVENT1**	R	0	The "excluded muddle".

Table 446. LRADC Battery Result Register Description

28.1.4. LRADC1 Control Register

This register controls the overall operation of LRADC 1.

***WARNING**: The pad ESD protection limits maximum voltage on all LRADC inputs. The BATT LRADC is specifically designed to handle higher voltages, but LRADC1 and LRADC2 inputs are limited to 3.3V.*

HW_LRADC1_CTRL X:$FA23

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
POLARITY_EVENT1	POLARITY_EVENT0	IRQ_EN_EVENT1	IRQ_EN_EVENT0			REF_VAL				CLK_DIV		PWD	CLEAR	INPUT_DIV2	HALF_CMP_PWR		INPUT_OFFSET						

Table 447. HW_LRADC1_CTRL

BITS	LABEL	RW	RESET	DEFINITION
23	**POLARITY_EVENT1**	RW	0	Set to one to trigger event 1 interrupts when filtered A/D value is greater than the event 1 threshold. Set to zero for interrupt when the filtered A/D value is less than the event 1 threshold.
22	**POLARITY_EVENT0**	RW	0	Set to one to trigger event 0 interrupts when filtered A/D value is greater than the event 0 threshold. Set to zero for interrupt when the filtered A/D value is less than the event 0 threshold.
21	**IRQ_EN_EVENT1**	RW	0	Set to one to enable an interrupt from the event 1 comparator.
20	**IRQ_EN_EVENT0**	RW	0	Set to one to enable an interrupt from the event 0 comparator.
21:18	**RSRVD**	R	00	**Reserved –** Must be written with 0.
17:16	**REF_VAL**	RW	00$	Reference voltage selection, see Table 440. “LRADC Input Ranges and Resolutions” on page 310. These are the primary mode control bits. They affect the range and resolution of the converter.
15:14	**RSRVD**	R	00	**Reserved –** Must be written with 0.
13:12	**CLK_DIV**	RW	00	Clock divider selection. 00 - divide by 4 01 - divide by 8 10 - divide by 16 11 - divide by 32
11	**PWD**	RW	1	Set this bit to one to power down the comparator, the R-Ladder and stop the internal clocks to minimize power consumption. Set to zero for normal operation.
10	**CLEAR**	RW	1	Set this bit to one to force the up/down counter to all zeroes. Set to zero for normal operation.

Table 448. LRADC 1 Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
9	**INPUT_DIV2**	RW	0	Set to one to divide the input voltage by two before applying it to the comparator. The division by two changes the input resistance from essentially infinite to 100KΩ.
8	**HALF_CMP_PWR**	RW	0	Set to one to reduce the analog current for low power operation. This is accomplished at the expense of conversion performance. This bit should only be set to one if static measurements are needed.
7	**RSRVD**	R	0	**Reserved –** Must be written with 0.
6:0	**INPUT_OFFSET**	RW	$00	These bits are added to the final converter output to add a digital offset to the converted value. This may be useful for LRADC calibration and to compensate for additional bias resistor when it is switched into the R-Ladder.

Table 448. LRADC 1 Control Register Description (Continued)

28.1.5. LRADC1 Threshold Register

This register sets the thresholds used by the digital comparators in LRADC 1.

HW_LRADC1_THRSHX:$FA24

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			THRESHOLD_EVENT1												THRESHOLD_EVENT0								

Table 449. HW_LRADC1_THRSH

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	000	**Reserved –** Must be written with 0.
20:12	**THRESHOLD_EVENT1**	RW	$000	This nine bit field defines the set point for event detector one's threshold comparator. The nine bit post offset conversion value is compared to this threshold.
11:9	**RSRVD**	R	000	**Reserved –** Must be written with 0.
8:0	**THRESHOLD_EVENT0**	RW	$000	This nine bit field defines the set point for event detector zero's threshold comparator. The nine bit post offset conversion value is compared to this threshold.

Table 450. LRADC 1 Threshold Register Description

28.1.6. LRADC1 Result Register

This register controls the interrupts reported by LRADC 1 and provides read only views of the LRADC data value and the results of the event detector threshold comparators.

HW_LRADC1_RESULTX:$FA25

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		IRQ_EVENT1	IRQ_EVENT0				DATA_OUT											LT_EVENT0	LT_EVENT1	GT_ENVET0	GT_EVENT1	EQ_EVENT0	EQ_EVENT1

Table 451. HW_LRADC1_RESULT

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	00	**Reserved –** Must be written with 0.
21	**IRQ_EVENT1**	RW	0	Interrupt request status for event detector one. This sticky bit is set whenever event detector one's comparator trips the set point. See **HW_LRADC1_CTRL_POLARITY_EVENT1** and **HW_LRADC1_THRSH_THRESHOLD1**. Software clears this bit by writing a one to it.
20	**IRQ_EVENT0**	RW	0	Interrupt request status for event detector zero. This sticky bit is set whenever event detector zero's comparator trips the set point. See **HW_LRADC1_CTRL_POLARITY_EVENT0** and **HW_LRADC1_THRSH_THRESHOLD0**. Software clears this bit by writing a one to it.
19:17	**RSRVD**	R	00	**Reserved –** Must be written with 0.
16:8	**DATA_OUT**	R	$000	The current nine bit value of the LRADC is read here. ***WARNING:*** due to clock domain crossings, software must repeatedly read this value until two successive values are read which match bit for bit.
7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5	**LT_EVENT0**	R	0	This read only bit indicates the current state of the event detector 0 Less Than result.
4	**LT_EVENT1**	R	0	This read only bit indicates the current state of the event detector 1Less Than result.
3	**GT_EVENT0**	R	0	This read only bit indicates the current state of the event detector 0 Greater Than result.
2	**GT_EVENT1**	R	0	This read only bit indicates the current state of the event detector 1 Greater Than result.
1	**EQ_EVENT0**	R	0	The "excluded muddle".
0	**EQ_EVENT1**	R	0	The "excluded muddle".

Table 452. LRADC 1 Result Register Description

28.1.7. LRADC2 Control Register

This register controls the overall operation of LRADC 2.

***WARNING**: The pad ESD protection limits maximum voltage on all LRADC inputs. The BATT LRADC is specifically designed to handle higher voltages, but LRADC1 and LRADC2 inputs are limited to 3.3V.*

HW_LRADC1_CTRL X:$FA26

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
POLARITY_EVENT1	POLARITY_EVENT0	IRQ_EN_EVENT1	IRQ_EN_EVENT0			REF_VAL				CLK_DIV		PWD	CLEAR	INPUT_DIV2	HALF_CMP_PWR		INPUT_OFFSET						

Table 453. HW_LRADC2_CTRL

BITS	LABEL	RW	RESET	DEFINITION
23	**POLARITY_EVENT1**	RW	0	Set to one to trigger event 1 interrupts when filtered A/D value is greater than the event 1 threshold. Set to zero for interrupt when the filtered A/D value is less than the event 1 threshold.
22	**POLARITY_EVENT0**	RW	0	Set to one to trigger event 0 interrupts when filtered A/D value is greater than the event 0 threshold. Set to zero for interrupt when the filtered A/D value is less than the event 0 threshold.
21	**IRQ_EN_EVENT1**	RW	0	Set to one to enable an interrupt from the event 1 comparator.
20	**IRQ_EN_EVENT0**	RW	0	Set to one to enable an interrupt from the event 0 comparator.
21:18	**RSRVD**	R	00	**Reserved –** Must be written with 0.
17:16	**REF_VAL**	RW	00$	Reference voltage selection, see Table 440. “LRADC Input Ranges and Resolutions” on page 310. These are the primary mode control bits. They affect the range and resolution of the converter.
15:14	**RSRVD**	R	00	**Reserved –** Must be written with 0.
13:12	**CLK_DIV**	RW	00	Clock divider selection. 00 - divide by 4 01 - divide by 8 10 - divide by 16 11 - divide by 32
11	**PWD**	RW	1	Set this bit to one to power down the comparator, the R-Ladder and stop the internal clocks to minimize power consumption. Set to zero for normal operation.
10	**CLEAR**	RW	1	Set this bit to one to force the up/down counter to all zeroes. Set to zero for normal operation.

Table 454. LRADC 2 Control Register Description

BITS	LABEL	RW	RESET	DEFINITION
9	**INPUT_DIV2**	RW	0	Set to one to divide the input voltage by two before applying it to the comparator. The division by two changes the input resistance from essentially infinite to 100KΩ.
8	**HALF_CMP_PWR**	RW	0	Set to one to reduce the analog current for low power operation. This is accomplished at the expense of conversion performance. This bit should only be set to one if static measurements are needed.
7	**RSRVD**	R	0	**Reserved –** Must be written with 0.
6:0	**INPUT_OFFSET**	RW	$00	These bits are added to the final converter output to add a digital offset to the converted value. This may be useful for LRADC calibration and to compensate for additional bias resistor when it is switched into the R-Ladder.

Table 454. LRADC 2 Control Register Description (Continued)

28.1.8. LRADC2 Threshold Register

This register sets the thresholds used by the digital comparators in LRADC 2.

HW_LRADC2_THRSHX:$FA27

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			THRESHOLD_EVENT1												THRESHOLD_EVENT0								

Table 455. HW_LRADC2_THRSH

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	000	**Reserved –** Must be written with 0.
20:12	**THRESHOLD_EVENT1**	RW	$000	This nine bit field defines the set point for event detector one's threshold comparator. The nine bit post offset conversion value is compared to this threshold.
11:9	**RSRVD**	R	000	**Reserved –** Must be written with 0.
8:0	**THRESHOLD_EVENT0**	RW	$000	This nine bit field defines the set point for event detector zero's threshold comparator. The nine bit post offset conversion value is compared to this threshold.

Table 456. LRADC 2 Threshold Register Description

28.1.9. LRADC2 Result Register

This register controls the interrupts reported by LRADC 2 and provides read only views of the LRADC data value and the results of the event detector threshold comparators.

HW_LRADC1_RESULTX:$FA28

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		IRQ_EVENT1	IRQ_EVENT0				DATA_OUT											LT_EVENT0	LT_EVENT1	GT_ENVET0	GT_EVENT1	EQ_EVENT0	EQ_EVENT1

Table 457. HW_LRADC2_RESULT

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	00	**Reserved –** Must be written with 0.
21	**IRQ_EVENT1**	RW	0	Interrupt request status for event detector one. This sticky bit is set whenever event detector one's comparator trips the set point. See **HW_LRADC2_CTRL_POLARITY_EVENT1** and **HW_LRADC2_THRSH_THRESHOLD1**. Software clears this bit by writing a one to it.
20	**IRQ_EVENT0**	RW	0	Interrupt request status for event detector zero. This sticky bit is set whenever event detector zero's comparator trips the set point. See **HW_LRADC2_CTRL_POLARITY_EVENT0** and **HW_LRADC2_THRSH_THRESHOLD0**. Software clears this bit by writing a one to it.
19:17	**RSRVD**	R	00	**Reserved –** Must be written with 0.
16:8	**DATA_OUT**	R	$000	The current nine bit value of the LRADC is read here. ***WARNING:*** due to clock domain crossings, software must repeatedly read this value until two successive values are read which match bit for bit.
7:6	**RSRVD**	R	00	**Reserved –** Must be written with 0.
5	**LT_EVENT0**	R	0	This read only bit indicates the current state of the event detector 0 Less Than result.
4	**LT_EVENT1**	R	0	This read only bit indicates the current state of the event detector 1 Less Than result.
3	**GT_EVENT0**	R	0	This read only bit indicates the current state of the event detector 0 Greater Than result.
2	**GT_EVENT1**	R	0	This read only bit indicates the current state of the event detector 1 Greater Than result.
1	**EQ_EVENT0**	R	0	The "excluded muddle".
0	**EQ_EVENT1**	R	0	The "excluded muddle".

Table 458. LRADC 2 Result Register Description

29. BOOT MODES

29.1. General Information on Boot Modes

The on-chip ROM contains program code that is responsible for loading code from outside the chip into on-chip RAM, and transferring control to that code. The DSP determines which mode to use for booting by examining the signal level on some pins immediately after reset. Table 459 shows the pins used for determining the boot mode

100-PIN TIFFS PIN #	144-PIN FPBGA PIN #	PIN LABEL	BOOT FUNCTION
95	J1	GP0	Set to one to enable POST operation
84	D1	GP8	Specifies boot mode bit 0
94	H2	GP1	Specifies boot mode bit 1
93	H4	GP2	Specifies boot mode bit 2
92	H3	GP3	Specifies boot mode bit 3

Table 459. Boot Control Pins

NOTE: The Power on self test (POST) must be run in all applications of the STMP35xx. Therefore GP0 (pin 95/J1) must be tied high through a high impedance resistance (47KΩ) on *ALL* customer boards.

These pins can and are used for other functions besides configuring the boot mode; this is achieved by specifying the high or low state on these pins with a high value resistor (typically 47kΩ) pulling the pin either up to VddIO or down to VssIO. During boot mode, these pins are not driven by the chip, and the pull-up/down resistors define the values. Table 460 shows the different boot modes supported.

GP0 POST	GP3 MODE 3	GP2 MODE 2	GP1 MODE 1	GP8 MODE 0	PORT	BOOT MODE
0	X	X	X	X	INVALID	Not Valid for customer use.
1	0	0	0	X	RESERVED	Reserved for future use
1	0	0	1	0	NAND 1.8 Volt	NAND with Play recovery
1	0	0	1	1	NAND 1.8 Volt	NAND with PSWITCH recovery
1	0	1	0	0	RESERVED	Reserved for future use
1	0	1	0	1	SPI	Master
1	0	1	1	0	RESERVED	Reserved for future use
1	0	1	1	1	UTMI	UTMI Self Test Mode
1	1	0	0	0	USB	STMP Boot Class - Bulk out Endpoint
1	1	0	0	1	I^2C	Slave
1	1	0	1	0	NAND 3.3 Volt	NAND with Play recovery
1	1	0	1	1	NAND 3.3 Volt	NAND with PSWITCH recovery
1	1	1	0	0	I^2C	Master
1	1	1	0	1	SPI	Slave
1	1	1	1	0	Tester Loader	INTERNAL USE ONLY
1	1	1	1	1	Burn In	INTERNAL USE ONLY

Table 460. Boot Modes

A Power On Self Test (POST) is available during the boot process. This function runs a self-test and repair function on the on-chip RAMs and swaps in spare bits

where necessary/possible to repair any defects found. Note that chips shipped by SigmaTel are guaranteed to have no *un-repaired* defects in the on-chip RAMs, however, they may contain repairable errors. This means that the POST ***MUST*** be run as part of the boot process for correct operation of the chip. The POST process takes about 95msec to complete, and is designed to consume as little power as possible. The POST process does not stop on errors, and does not give any indication as to success or failure; the application loaded by the boot loader should assume that the on-chip RAM has been repaired correctly. The POST operation will overwrite any data stored in on-chip RAM, so any data in on-chip RAM will be lost during the boot process.

The USB, I^2C master & slave, NAND Flash and SPI slave boot modes are intended to be used in applications of the chip; in other words, they are user boot modes. The INTERNAL boot modes are for internal SigmaTel use only. Any changes to the above boot mode table will preserve all of the entries for user boot modes to preserve compatibility for existing designs.

During reset, and while the POST process is running, all digital pins on the chip are in tristate mode. Once an individual boot mode starts, the boot mode will enable and configure the pins required for that boot mode. It is the responsibility of the application loaded by the boot ROM to enable & configure all of the pins it requires, the application should do this without making any assumptions about how the pins were configured by the boot ROM.

There is one additional implicit boot mode, over and above the boot modes listed in the above table, that is OnCE (On Chip Emulation) boot mode. This boot mode occurs when the external debug hardware pulls the ONCE_DRn pin low during the boot process to trigger the on-chip debugger. When this happens the DSP will execute the first few instructions in the on-chip ROM before stopping and allowing the external debugger to take control. In this circumstance, the actual boot mode selected on the boot pins will have no effect. OnCE boot mode will only be used by developers, however PCB boards should support the OnCE boot mode wherever possible, to enable system debug.

29.2. Bootloader Code Format

For the user boot modes, the boot loader expects the code read from the external storage to be organized in a specific format. This format is able to load blocks of data into X, Y, P, or L memory. It is also capable of initializing a block of memory to a specified value. Loading a block of code or data into memory is achieved by first sending a two-word command header, followed by the data. The command header is shown in Table 461.

	BITS 23-20	BITS 19-16	BITS [15:0]
WORD 0	Mc[3:0]	0000	Address[15:0]
WORD 1	00000000		Length[15:0]
WORD 2-N	Data[23:0]		

Table 461. Bootloader Command Header + Data

For Word 0, the Memory Control bits, Mc[3:0], are used to specify the target memory space for the code that is being loaded, and whether this is a block memory load, or an initialization of a range of addresses to one value. Table 462 specifies the meaning of these bits in more detail.

The Address[15:0] field specifies the base address in memory where data is to be written. The Length[15:0] specifies the length of the block of data in memory to be

MC3	MC2 (P)	MC1 (X)	MC0 (Y)	DESCRIPTION
0	0	0	0	Unused
0	0	0	1	Load into Y Memory
0	0	1	0	Load into X Memory
0	0	1	1	Load into L memory (XY memory)
0	1	0	0	Load into P memory
0	1	0	1	Unused
0	1	1	0	Unused
0	1	1	1	Unused
1	0	0	0	Unused
1	0	0	1	Initialize Y memory with data value
1	0	1	0	Initialize X memory with data value
1	0	1	1	Initialize L (i.e. X & Y) memory with data value
1	1	0	0	Initialize P memory with data value
1	1	0	1	Unused
1	1	1	0	Unused
1	1	1	1	Boot load complete, exit and jump to P:$0. Must be a complete Command Header ($F00000 $000000).

Table 462. Bootloader Memory Control Bits

loaded or initialized. When loading into P, X or Y memory, Words 2 through N will contain one word of data for each P, X, or Y memory location to be loaded. When loading into L memory, Words 2 through N will contain two words of data for each L memory location to be loaded, words 2, 4, etc., will go into X memory, words 3, 5, etc., will go into Y memory. When initializing P, X, or Y memory, Word 2 will specify the value to be used in initializing memory. When initializing L memory, Words 2 through 3 will specify the value to be used in initializing memory.

This data structure allows the initialization of hardware control registers as part of the boot process, this could be achieved by using the above data structure to write directly to a memory mapped hardware register. Most frequently, this is used to write to the HW_PXCFG & HW_PYCFG registers that control the configuration of on-chip RAM.

29.3. Encryption

The chip expects the code to be loaded by the boot loader to be encrypted. This encryption serves two purposes; it makes it harder for an end user to hack the device, and it includes a check-sum to prevent corrupted code from being used to boot the device. The boot loader will fall back to USB boot mode if the code being loaded fails the check-sum test, this allows a player with corrupted code to be recovered by an end user with the use of a recovery program on their PC.

The details of the encryption algorithm are not covered in this document; please contact SigmaTel for more information if needed. The SigmaTel SDK (Software Development Kit) includes everything needed to encrypt code that is to be loaded into the chip.

29.4. Bootloader Examples

This section provides a few examples of data delivered to the DSP and how it will be copied into memory. Note that these examples show the data before it is encrypted using the encryption mentioned above.

29.4.1. Boot Example #1

The following example of data presented to the DSP would result in 8 words being loaded into Y memory.

```
$100210         ; load into Y memory starting at Y:$0210
$000008         ; load 8 words of data into Y memory
$000000         ; data word #0, written to Y:$0210
$111111         ; data word #1, written to Y:$0211
$222222         ; data word #2, written to Y:$0212
$333333         ; data word #3, written to Y:$0213
$444444         ; data word #4, written to Y:$0214
$555555         ; data word #5, written to Y:$0215
$666666         ; data word #6, written to Y:$0216
$777777         ; data word #7, written to Y:$0217
$F00000         ; end of boot, execute program
$000000         ; this word IS required
```

29.4.2. Boot Example #2

The following example of data presented to the DSP would result in 8 contiguous P memory locations being initialized to a specified value.

```
$C00400         ; initialize P memory starting at P:$0400
$000008         ; initialize 8 locations, i.e. P:$0400-$0407
$CCCCCC         ; initialize with the value $CCCCCC
$F00000         ; end of boot, execute program
$000000         ; this word IS required
```

29.4.3. Boot Example #3

The following example of data presented to the DSP would result in 11 contiguous L memory (XY memory) locations being initialized to specified values.

```
$300230     ; load L memory at address L:$0230
$00000B     ; load 11 L words
$000001     ; data word #0 upper, loaded to X:$0230
$000002     ; data word #0 lower, loaded to Y:$0230
$000004     ; data word #1 upper, loaded to X:$0231
$000008     ; data word #1 lower, loaded to Y:$0231
$000010     ; data word #2 upper, loaded to X:$0232
$000020     ; data word #2 lower, loaded to Y:$0232
$000040     ; data word #3 upper, loaded to X:$0233
$000080     ; data word #3 lower, loaded to Y:$0233
$000100     ; data word #4 upper, loaded to X:$0234
$000200     ; data word #4 lower, loaded to Y:$0234
$000400     ; data word #5 upper, loaded to X:$0235
$000800     ; data word #5 lower, loaded to Y:$0235
$001000     ; data word #6 upper, loaded to X:$0236
$002000     ; data word #6 lower, loaded to Y:$0236
$004000     ; data word #7 upper, loaded to X:$0237
$008000     ; data word #7 lower, loaded to Y:$0237
$010000     ; data word #8 upper, loaded to X:$0238
$020000     ; data word #8 lower, loaded to Y:$0238
$040000     ; data word #9 upper, loaded to X:$0239
$080000     ; data word #9 lower, loaded to Y:$0239
$100000     ; data word #10 upper, loaded to X:$023A
$200000     ; data word #10 lower, loaded to Y:$023A
$F00000     ; end of boot, execute program
$000000     ; this word IS required
```

29.5. Boot Procedure

The flow chart in Figure 112 shows the initial boot sequence and Figure 113 shows the subsequent general boot procedure.

29.5.1. USB boot mode

The USB boot mode enables the USB interface, attempts to enumerate on the USB bus, and then expects USB Bulk-Out data from the host PC containing the data to be use for booting. This boot mode is intended to be used at the end of the production line for EOL (end of line) tests, and for personalizing the players. Together with a recovery application running on a PC, this mode may also be used by an end user to recover if the code on a player somehow becomes corrupted.

One issue with using the USB Boot Class MODE is that the on-chip ROM does not have a unique ID and therefore multiple devices built using the chip may not be connected simultaneously to a single computer. To get around this limitation all software implementations provided by SigmaTel will populate the boot component with a Boot Manager program. The Boot Manager is used to decide whether to load code for a player or for USB. The Boot Manager will check to see if USB is connected and if so

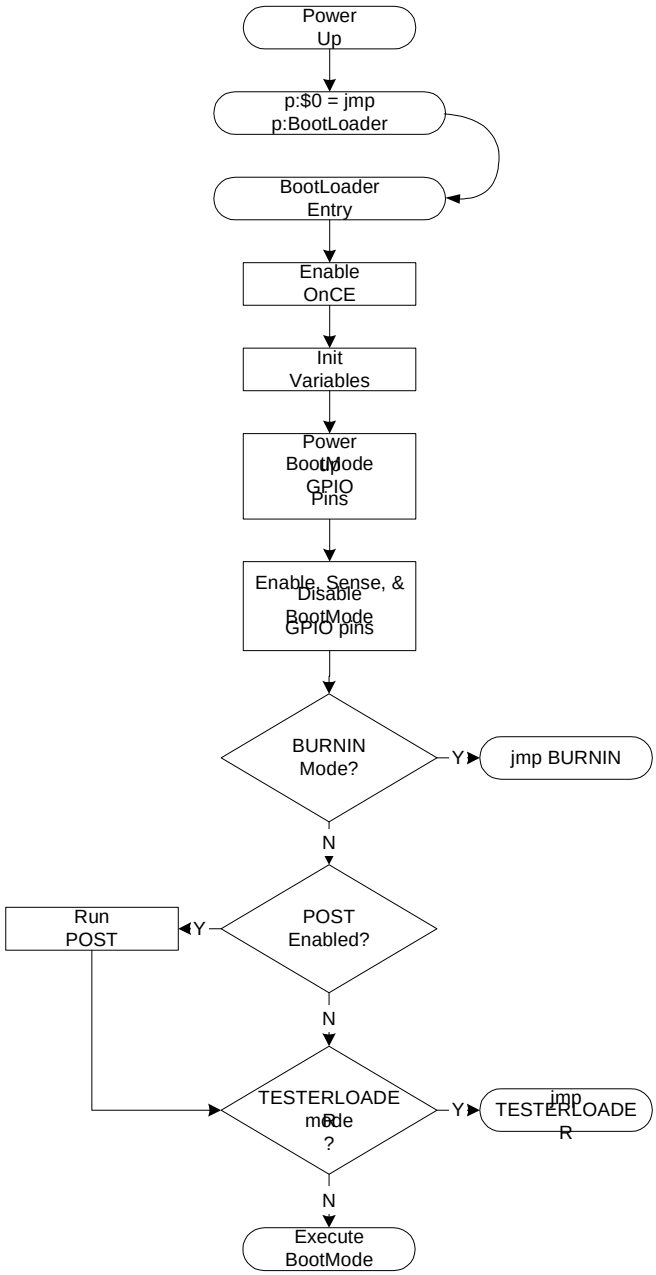


Figure 112. Initial Boot Sequence

it will load the Device Control Class (DCC) Image, and if not it will load the Player Image. The DCC Image will be specific to a given product hence it may have unique USB IDs and allow multiple devices to co-exist on a computer. In this manner the generic USB Boot Class is completely by-passed. Please refer to the chip Boot Manager document for more details. USB boot mode differs from all the other boot modes in that it leaves the PLL enabled and the DSP running at 68 MHz when done.

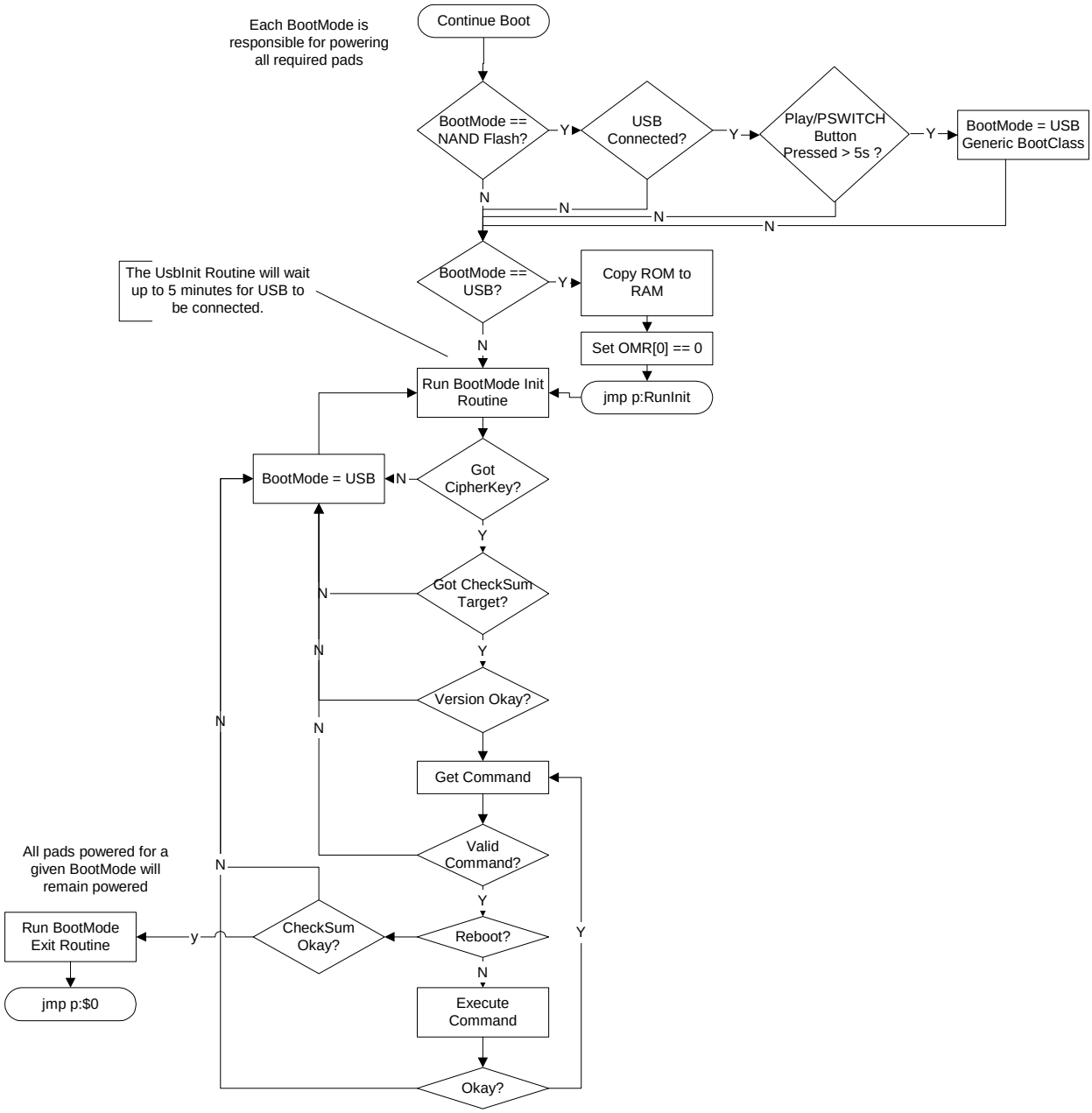


Figure 113. Boot Procedure

29.5.1.1. USB Boot Mode Pin Power

USB Boot Mode pins are analog pins and are powered at system startup up. No other digital pins are powered for this boot mode.

Note: See Table 503 for USB Interface pin placement.

29.5.2. NAND Flash Boot Mode

The NAND Flash boot mode will read boot commands from NAND Flash or SmartMedia flash devices. The Boot Mode Init routine will search up to four NAND/SmartMedia devices and boot from the first device/card that is found with valid boot code.

The NAND Flash boot mode searches for and boots from the chip Boot component. Please refer to the chip SmartMedia/NAND Blocks document for more details on components. The search will start with chip select 0 and move on to 1, 2, and 3. If an error is encountered then the device on the next chip select is searched. If desired, the entire code set may reside in the chip Boot component and the Boot Manager and other components may be left out of the system.

If all NAND/SmartMedia devices are searched and no valid boot code is found the boot mode is changed to USB, and the part attempts to boot from the USB bus. This situation will most commonly be encountered at the end of a production line where the devices being manufactured will not contain any code. This situation will also be encountered if the code on the NAND/SmartMedia device becomes corrupted for some reason, the end user will be able to reload the code onto the device by using a recovery program on a PC.

Finally, the end user can manually force a boot from USB by holding down the Play button (NAND Flash boot mode with play recovery), or power button (NAND Flash boot mode with PSWITCH recovery) for 5 seconds during the boot process. See Figure 113, Boot Procedures, for details.

29.5.2.1. NAND Flash Boot Mode Pin Power

The NAND Flash boot mode will power the following pins:

SM_D0	SM_D5	SM_ALE	SM_CE2n
SM_D1	SM_D6	SM_CLE	SM_CE3n
SM_D2	SM_D7		SM_WPn
SM_D3	SM_REn	SM_CE0n	SM_READY
SM_D4	SM_WEn	SM_CE1n	

Note: See Table 493 for SmartMedia/NAND pin placement.

29.5.3. I^2C Slave Boot Mode

The I^2C slave boot mode enables the I^2C port as a slave device, and waits for data to be written to the I^2C port using the bootloader code format documented above.

29.5.3.1. I^2C Slave Boot Mode Pin Power

The I^2C Slave boot mode will power the following pins:

I2C_SCL

I2C_SDA

Note: See Table 495 for I^2C Interface pin placement.

29.5.4. I^2C Master Boot Mode

The I^2C master boot mode enables the I^2C port as a master device, and then attempts to read data formatted using the bootloader code format documented above from an external I^2C device at I^2C address $A0. If a standard 24.576 MHz crystal is used, this boot mode will use an I^2C clock speed of ~180 kHz.

29.5.4.1. I^2C Master Boot Mode Pin Power

The I^2C Master boot mode will power the following pins:

I2C_SCL

I2C_SDA

Note: See Table 495 for I^2C Interface pin placement.

29.5.5. SPI Slave Boot Mode

The SPI slave boot mode enables the SPI port as a slave device, and waits for data to be written to the SPI port using the bootloader code format documented above.

29.5.5.1. SPI Slave Boot Mode Pin Power

The SPI Slave boot mode will power the following pins:

SPI_MOSI

SPI_MISO

SPI_SCK

SPI_SSn

Note: See Table 495 for SPI Interface pin placement.

29.5.6. TESTERLOADER Boot Mode

This boot mode provides a simple, fast interface to testers for loading test code. This mode is intended for internal SigmaTel use only, and no further documentation is provided here.

29.5.7. BURNIN Boot Mode

This boot mode provides a simple way to exercise large portions of the on-chip hardware. Note that the power consumed by the chip will be significantly higher than normal when the part is operating in BURNIN mode. This boot mode is intended for internal SigmaTel use, and no further documentation is provided here.

29.5.8. System Recovery Mode

This is not an explicit boot mode but is entered in a special case of the two NAND modes. If the chip is booted with MODE = NAND with play recovery and USB is connected, and the Play button is held for 5 seconds the player will automatically go to the USB MODE. This will cause the device to show up as a USB Boot Class device. If a host driver detects this device it may take action to reformat the SmartMedia/NAND. Please refer to Figure 113.

29.6. Memory Maps

Various memory maps are employed in the different boot modes. Figure 114 shows the memory map for the USB boot modes and Figure 115 shows the map for all other modes. Shaded areas are available for loading code.

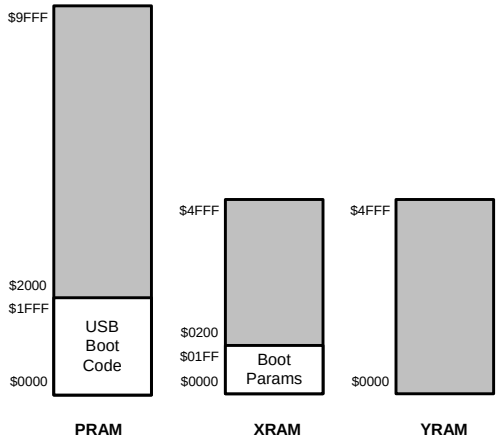


Figure 114. USB Boot Mode Memory Map

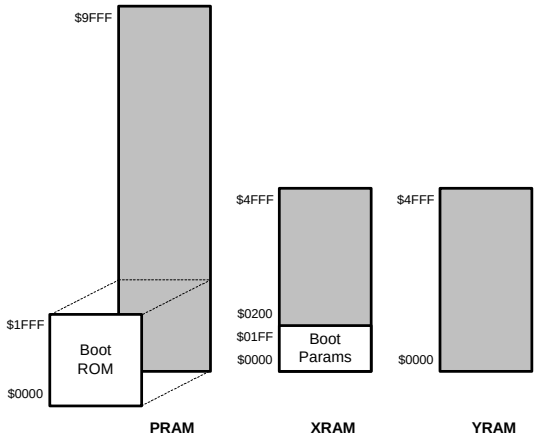


Figure 115. All Other Boot Modes Memory Map

In USB Boot Class p:$0..1 are available for writing the Reboot vector. All other interrupt vectors must be created by code at run-time.

Although the interrupt vector table is available (P:$0000-P:$007F) it is required that all vectors must be created by code at run-time.

30. DC-DC CONVERTER

The chip has 2 programmable integrated DC-DC converters that can be used to provide power for the device as well as the entire application, including external NAND flash memories, etc. The converters can be configured to operate from standard battery chemistries in the range of 0.9-3.6 volts including alkaline cells, NiMH, LiIon, etc. If multiple batteries are used, a series configuration is strongly preferred to optimize efficiency and battery life. Also, for applications that do not need integrated power management, the converters can be disabled and external supplies provided.

Board layout and external inductor/capacitor quality are critical to maximize analog performance and as well as DC-DC converter efficiency. Users should read and understand SigmaTel's printed circuit board layout application notes for guidance before beginning PCB layout. Further, users should refer to SigmaTel's reference designs for assistance in selecting the inductor and large external decoupling capacitors.

30.1. DC-DC Converter Theory of Operation

The DC-DC power conversion and charging circuitry must support a wide range of battery configurations, chemistries, and ultimately input voltages to the BATT pin, from a weak single cell alkaline battery at 0.9 volts to a fully charge LiIon battery at 4.2 volts. In addition, it must operate off of a 5 volt DC input supplied either from the USB's VBUS or from a DC power source derived from the AC mains. In all of these cases, it must supply a steady reliable 3.3 volts to the I/O rail and 1.8 volts to the core digital power rail and to the analog power rail. It is desirable for the DC-DC converter to supply a separate clean source of power for the analog rail that is isolated from the core digital 1.8 volt rail. Using separate supplies improves the peak power available to the application. In extremely cost sensitive applications, these rails can be tied together and supplied from a single capacitor.

In addition to supplying the core 1.8 volt analog and digital sources, the DC-DC converter must supply power to the other chips or devices in the application, e.g. NAND Flash power, SDRAM power, IDE power, etc. The DC-DC converter must also support operating the system at reduced rail voltages for power saving modes.

When generating 3.3 volts for the I/O rail from a single alkaline battery, the DC-DC converter must operate in a *"boost"* mode, as shown in Figure 116, below. In this case, the 0.9 volts applied to the battery pin must be *boosted* to 3.3V. For boost mode operation, the battery is connected through an inductor to the DCDC_BATT pin. This pin has both a low resistance N-FET and a low resistance P-FET. When the N-FET switches on, it pulls the inductor to ground. When it switches off, the inductor voltage goes quite positive as the field collapses. During this high voltage phase, the P-FET device can be switched on to charge the capacitance on the VDD I/O rail. The period of this oscillation determines how much high voltage is available from the inductor while the duty cycle of the P-FET device determines how much charge is transferred to VDD I/O.

The VDD I/O rail has a tapped voltage divider (R-ladder) which is fed to a comparator. The comparator references the band gap voltage (V_{BG}). The comparator effectively indicates whether the VDD I/O rail is above or below the desired voltage. If it is below the desired voltage then the P-FET should be switched on to charge the rail capacitance. If the rail droops to a lower voltage, then the N-FET on-time is increased to increase the inductor field, making more back EMF available for the next P-FET transfer. The DC-DC converter operates with a feedback loop filter to determine the desired duty cycle for the P-FET. The loop is constrained at both positive and negative limits, see 30.6.1. "DCDC1 Control Register 0" on page 347.

The **HW_DCDC_VDDIO_VOLTAGE_LEVEL** bit field allows one of 24 taps to be selected on the VDD I/O voltage divider, allowing a DC range from 2.05 V to 4.03 V. Thus one can set a variety of voltage targets for the VDD I/O rail optimizing either energy consumption or performance, as required. The loop filter output can be checked in the **HW_SPEED** register for design validation purposes.

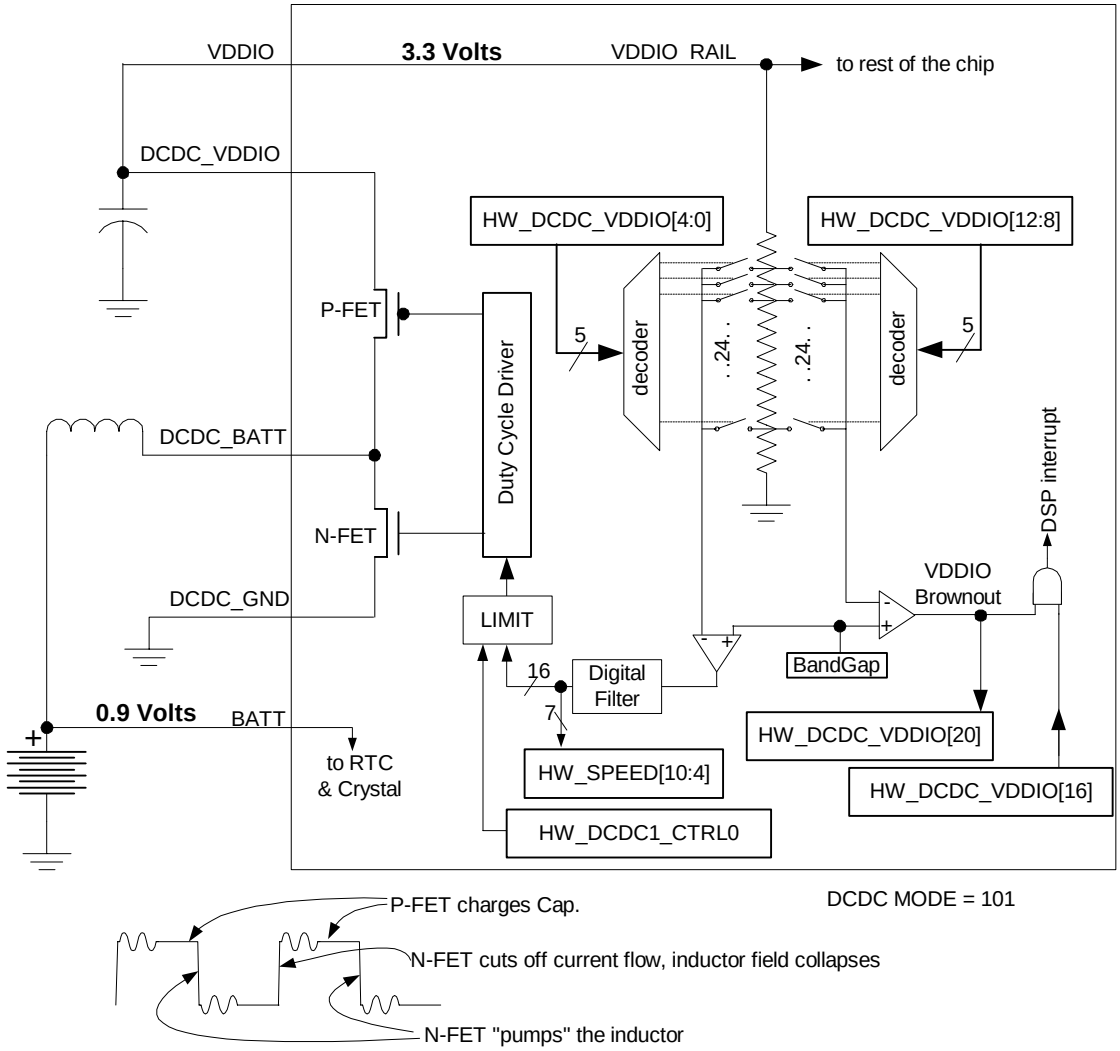


Figure 116. DC-DC Converter Boost Theory of Operation

There is a second set of taps and a switch matrix on the VDD I/O voltage divider controlled by **HW_DCDC_VDDIO_BROWNOUT_LEVEL**. This tap is connected to a second V_{BG} referenced comparator which detects under-voltage or *brown-out* conditions. The comparator results can generate a brown-out alert so that software can take appropriate action, i.e load shedding or graceful power-down as appropriate. The comparator result can be seen in **HW_DCDC_VDDIO_BROWNOUT_STATUS**. The interrupt is enabled by the **HW_DCDC_VDDIO_BROWNOUT_ENABLE** bit.

A *buck* mode is provided for dropping the appropriate voltage across the inductor to regulate a higher voltage battery input down to the desired VDD I/O rail. For a fully charged LiIon battery at 4.2 V, one must drop 0.9 volts across the inductor to keep a

steady 3.3 V on the I/O rail, see Figure 117. “DC-DC Converter Buck Theory of Operation” on page 334. In this mode, the P-FET switches on to add charge to the VDD I/O rail capacitance and to store energy in the inductor field, while the N-FET switches on to remove energy from the inductor field. Again, it is the duty cycle that controls the target voltage. The duty cycle is controlled by a feedback loop that tracks the error between the desired rail voltage and V_{BG}, similar to what we saw in the boost mode loop.

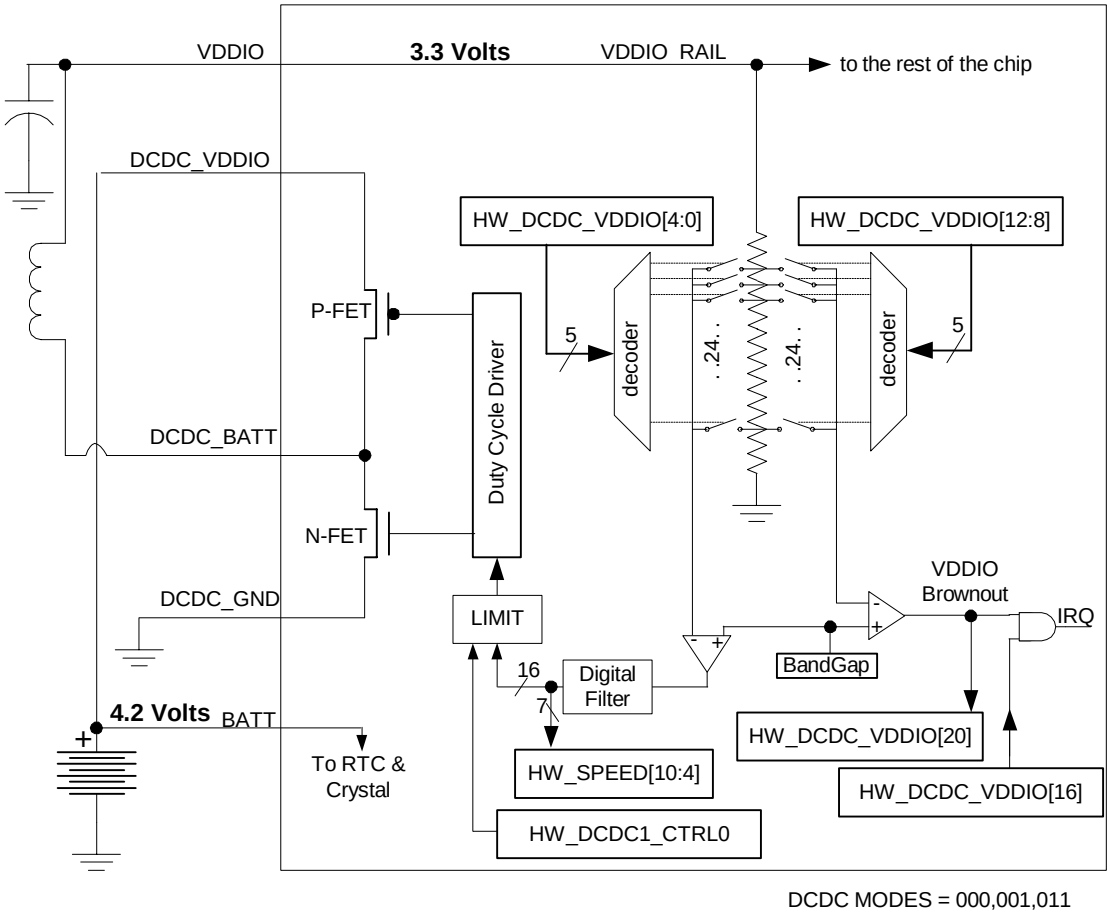


Figure 117. DC-DC Converter Buck Theory of Operation

NOTE: the DCDC converter pin names can be misleading in buck mode. For example the pin named *“VDDIO”* is actually connected to the battery while the pin named *“DCDC_BATT”* is actually connected to the VDD I/O rail.

The brownout detection circuitry works in exactly the same way as it does in boost mode. Notice that board level wiring has to change between these two modes. One must determine the desired mode before designing the application board schematic.

The above discussion focused on generating a single regulated voltage for a single VDD rail. The STMP35xx has three distinct power planes or rails: 3.3 V VDD I/O rail, 1.8 V VDD rail for the digital core and a 1.8 V VDD rail for the analog circuitry, see Table 463. “Power Pin to Power Plane Mapping” on page 335. This table shows which power pins must be attached to which power planes on the application board. The discussion below shows the various ways these power planes can be supplied for the battery chemistry and configuration desired.

POWER PLANE DESIGNATION	PIN NAME	100-PIN TQFP	144-PIN FPBGA
VddIO1	VddIO1	29	G8
VddIO2	VddIO2	96	G5
VddIO3	VddIO3	-	F7
VddIO4	VddIO4	-	E7
VddA1	VddA1	73	B2
VddA2	VddPLL	76	B1
VddA3	VddHP	64	D5
VddD1	VddD1	39	E8
VddD2	VddD2	11	H7
VddD3	VddD3	86	F6

Table 463. Power Pin to Power Plane Mapping

Figure 118, below, shows all three regulator channels of DC-DC converter #1 running in boost mode. As shown in the figure, there is one N-FET that generates the bulk high voltage, boosted across the inductor. There are three P-FET devices that can be switched on to charge the three separate voltage planes attached to the converter.

The DC-DC converter selects one P-FET per cycle to charge one of the VDD rail channels. Each rail has its own voltage divider, comparator, converter feedback loop and duty cycle control. Thus the N-FET will turn on once per cycle and exactly one of the P-FETs will be turned on. There is a PFM mode that allows all P-FETs to remain off for a cycle for low load applications.

Figure 118 also shows some additional circuitry related to the DC-DC converter. Notice that the BATT pin is connected to a linear regulator which supplies the crystal oscillator, the real time clock and the portion of the DC-DC converter that must remain on even when the STMP35xx is in a *“powered-down”* state. Thus these circuits are powered up as soon as the battery is installed and remain powered until the battery is removed or discharged. This power domain is referred to elsewhere in the data sheet as the real time clock or crystal power domain. The power switch (PSWITCH) monitoring circuitry is also powered from the crystal power domain so that it can awaken the DCDC controller. NOTE: it is not necessary for the crystal oscillator and RTC to be enabled and operational before the power switch triggers a power up event. The power switch causes the oscillator to awaken which counts out a delay and awakens the DCDC converter controller to begin to ramp the rail voltages. Thus in the lowest energy consuming powered off state, all clocks are stopped and a DC circuit is waiting for the PSWITCH transition.

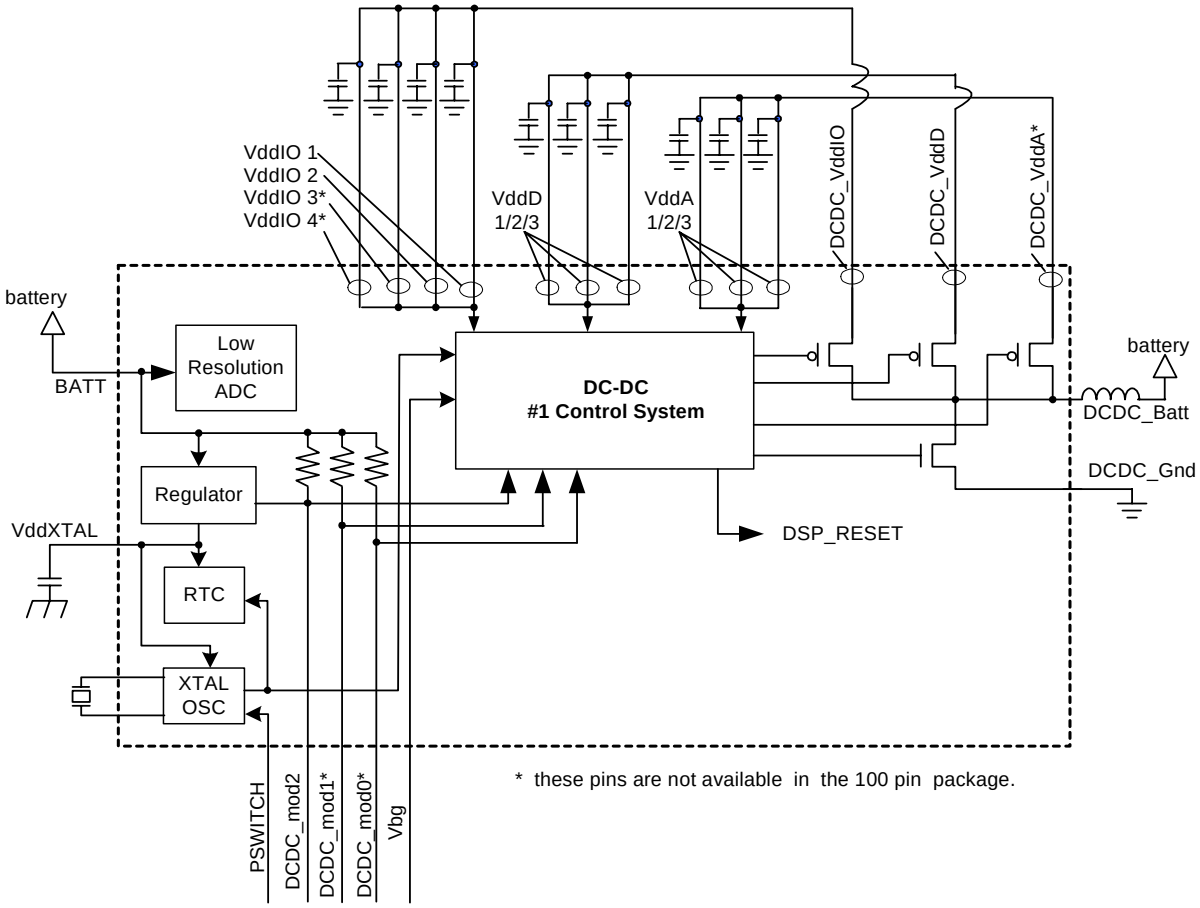


Figure 118. DC-DC #1 Converter, 3 output Channels

30.1.1. Defining Battery Configuration

The outputs of both DC-DC converters are isolated from the power rails of the chip. Therefore, the connection between the DC-DC converter outputs and the voltage rails of the chip must be done on the PCB. However, the specific PCB connections and battery connections are unique to each battery configuration as described below.

The battery configuration of the player is determined by the three mode pins DCDC_mod2, DCDC_mod1, DCDC_mod0. These pins each have an internal 100 kΩ pull-up resistor to the BATT pin (either the battery must always be connected to the BATT pin for the device to power up or the device must have power supplied to its 5 V input pin, VDD5V), so each mode pin is default high. Low value resistors should be added on the board to pull the desired mode pin low. Note that the configuration of these pins cannot be changed dynamically. In 100-pin configurations, only DCDC_mod2 is available on a pin, therefore only modes 111 and 011 can be used in this configuration. Also, the pins of DCDC Converter #2 are not available in a 100-pin configuration.

The selection of the three DC-DC mode pins determines whether each of the two DC-DC converters is operating in Buck Mode, Boost Mode, or is powered off. As the

following table shows, each converter operates in Boost Mode when its output voltages > battery voltage, and in Buck Mode when output voltage < battery voltage.

Table 464 details the decode of the three DC-DC mode select pins:

DCDC_MODE2:0	POWER SOURCE	DCDC CONVERTER #1	DCDC CONVERTER #2
111	1 Alkaline or 1 NiMH (0.9V-1.5V)	2-channel boost (1.8/3.3 V)	off
110*	reserved	reserved	reserved
101*	1 Alkaline or 1 NiMH (0.9V-1.5V)	3-channel boost (1.8/1.8/3.3 V)	off
100*	reserved	reserved	reserved
011	LiIon, (3.0-3.6V)	1-channel buck (1.8 V)	off
010*	External supplies	off	off
001*	2 Alkaline or 2 NiMH (1.8V-3.0V)	1-channel buck (1.8 V)	1-channel boost (3.3 V)
000*	LiIon (3.3V-4.2V)	1-channel buck (1.8 V)	1-channel buck (3.3 V)

* Only available in 144-pin package

Table 464. Decode of the DC-DC Mode Select Pins

As Table 464 shows, DC-DC converter #2 is only used to generate the 3.3 V rail when higher voltage battery configurations are selected. Additionally, DC-DC converter #2 has a lower resistance PMOS FET that can perform well in high current applications such as rotating media. In DCDC mode 011, the DC-DC converter only generates the 1.8 V rail, so an external regulator must be used to generate the 3.3 V I/O rail voltage.

30.1.1.1. DC-DC Converter Configuration

The two integrated DC-DC converters provide several low resistance FETs for use in power conversion as shown in the following Figure 119:

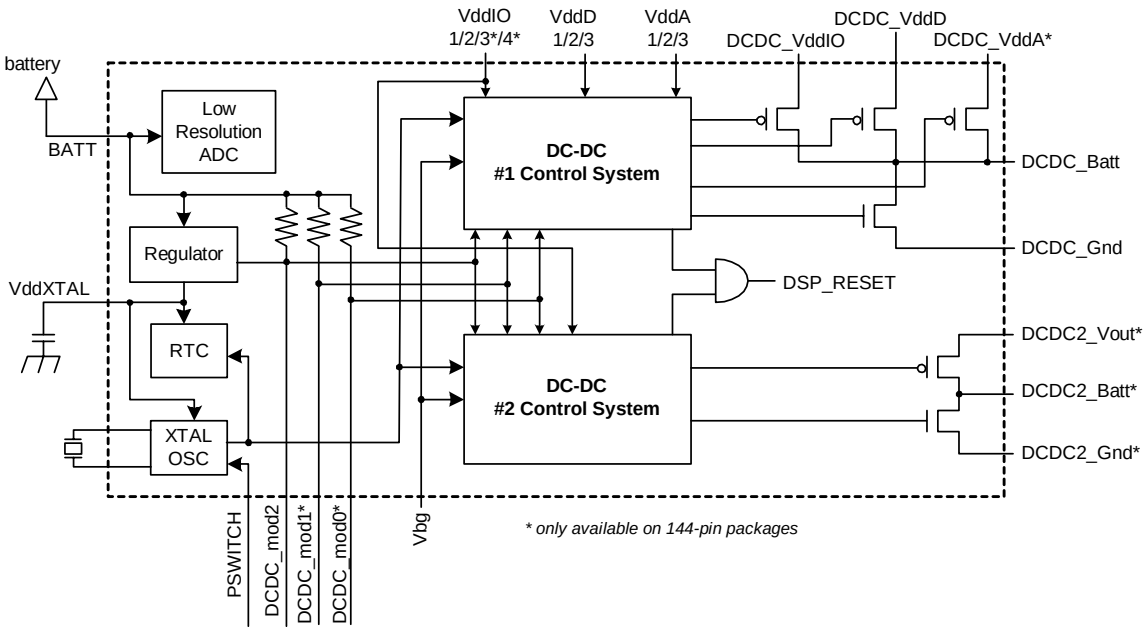


Figure 119. DC-DC Converter Control System

The configuration of the three DC-DC mode pins described in the previous section determines how these two groups of FETs are switched to provide the desired output voltage values. The FETs can be switched to either create a Buck Mode converter (battery >= output voltage) or Boost Mode converter (battery <= output voltage). These two different operating modes require different connectivity between the battery and inductor on the PCB. The various arrangements are shown in Figures 120 to 125.

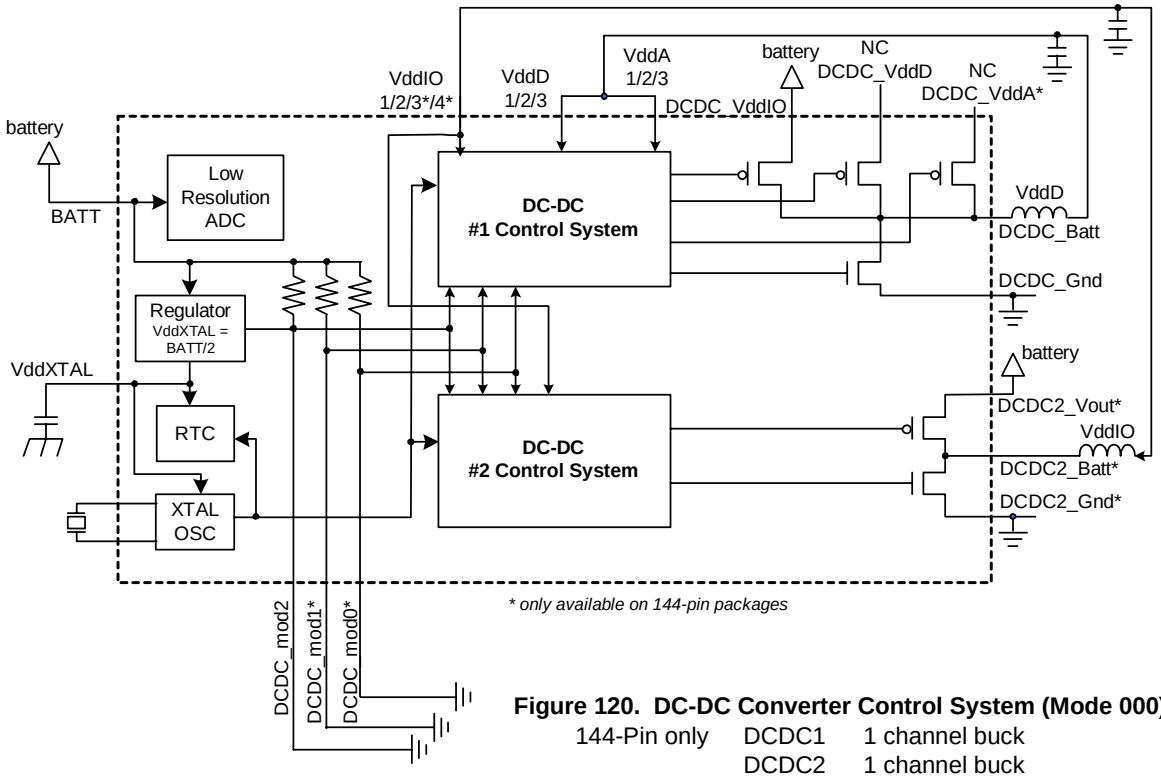


Figure 120. DC-DC Converter Control System (Mode 000)

144-Pin only	DCDC1	1 channel buck
	DCDC2	1 channel buck

30.1.1.2. Power Up Sequence

The DC-DC converters control the power up sequence of the device and holds the rest of the chip in reset until after the power up sequence is complete. This is necessary to prevent operation of the DSP and system before stable power supply voltages are present.

The Power Up sequence begins when the battery is connected to the BATT pin of the device (or a 5 V source is connected to the VDD5V pin). As shown in Figure 119, the BATT pin provides the pull-up on the three DC-DC mode select pins, as well as supplying power to the crystal oscillator and the real-time clock. This means that the crystal oscillator can be running, if desired, whenever a battery is connected to BATT pin. This feature allows the real time clock to operate when the chip is in the off state. The crystal oscillator/RTC is the only power drain on the battery in this state and consumes only a very small amount of power. During this time, the digital (VddD) and analog (VddA) supplies are held at ground, while the VddIO rail is shorted to the battery. This is the off state that continues until the PSWITCH pin is asserted high.

When the PSWITCH pin is asserted, for at least 100ms, the DC-DC converters are enabled and the device attempts to power up. Depending on the DC-DC mode

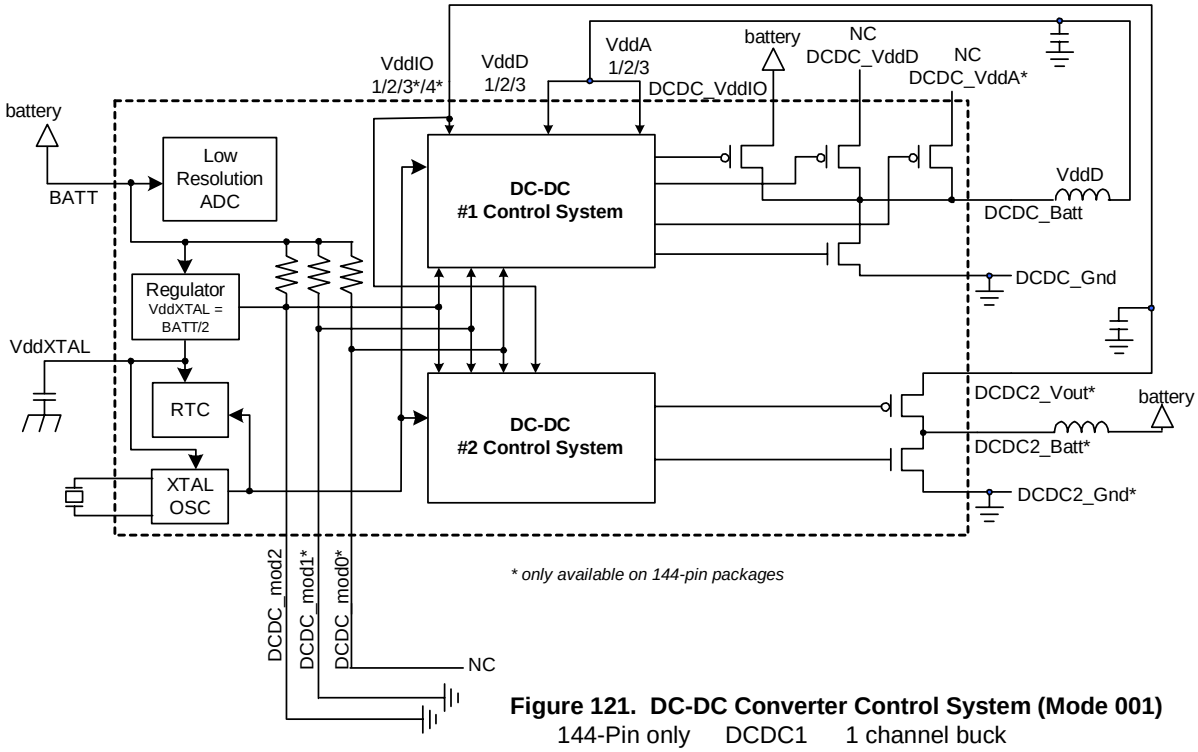


Figure 121. DC-DC Converter Control System (Mode 001)
144-Pin only DCDC1 1 channel buck
DCDC2 1 channel boost

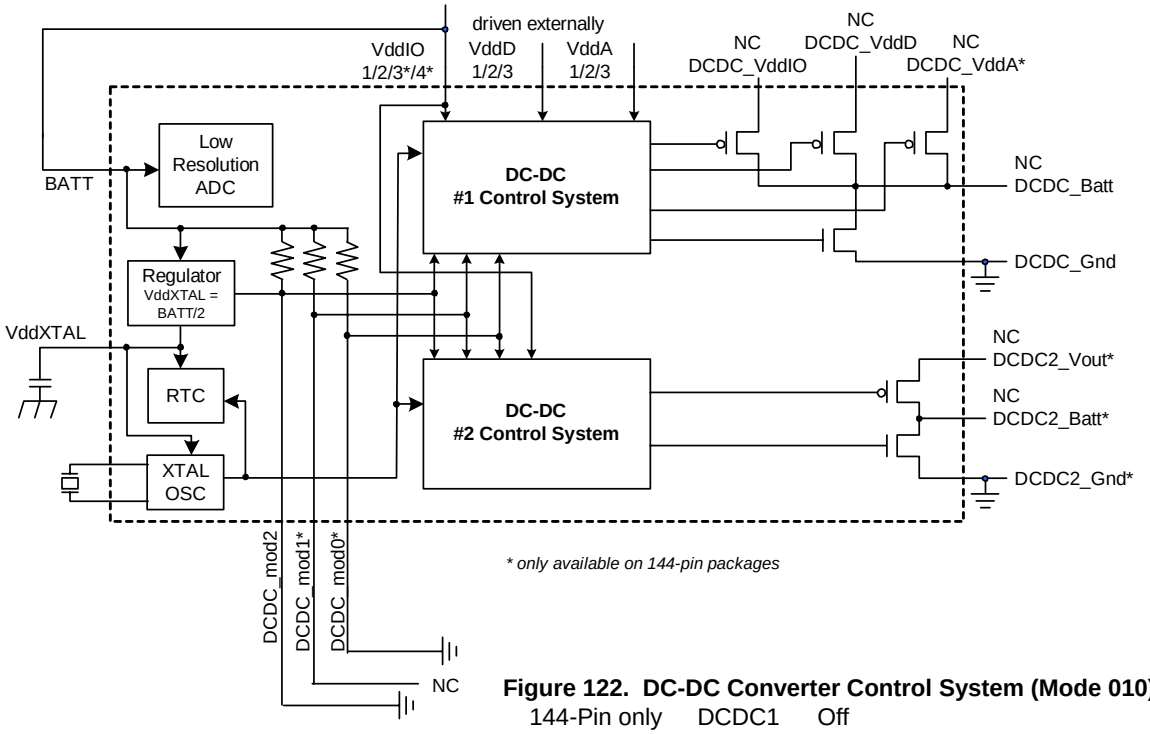


Figure 122. DC-DC Converter Control System (Mode 010)
144-Pin only DCDC1 Off
DCDC2 Off

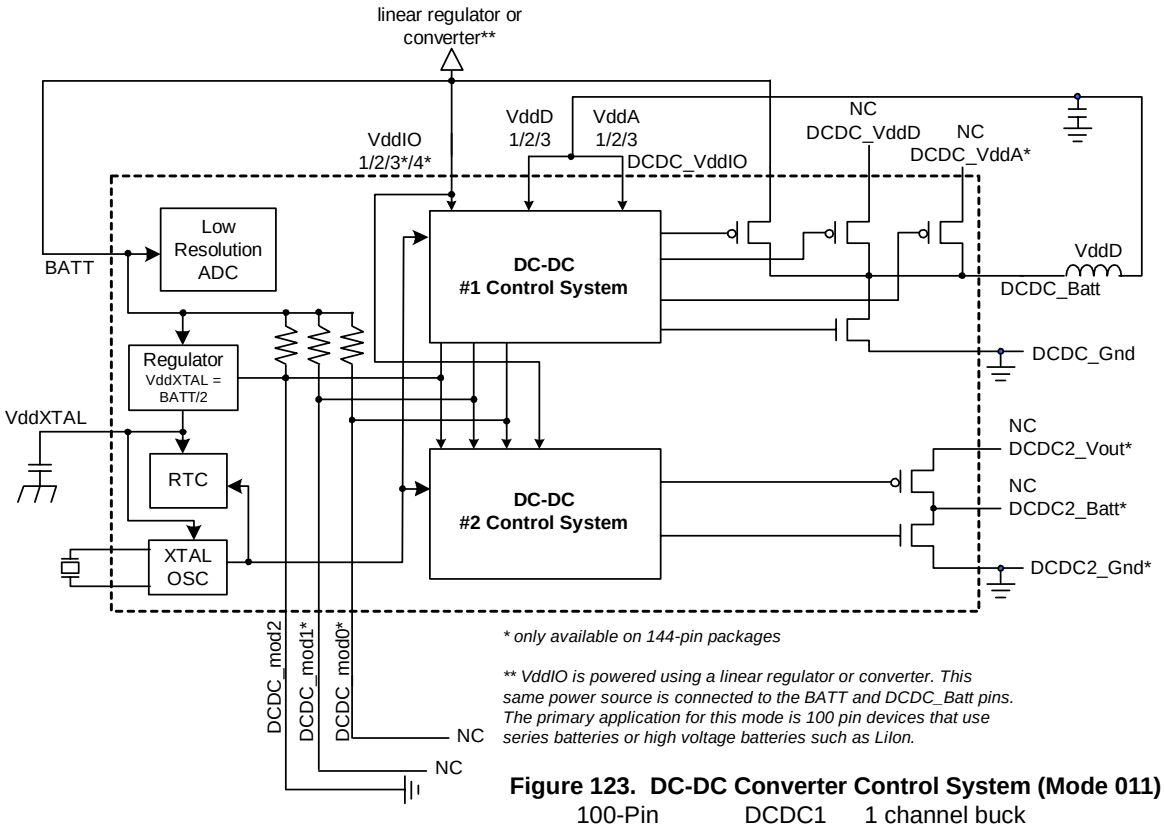


Figure 123. DC-DC Converter Control System (Mode 011)

100-Pin	DCDC1	1 channel buck
or 144-Pin	DCDC2	Off

select pins, one or both of the DC-DC converters begin switching and drive the voltage outputs toward the target values. When both of the DC-DC converters' control systems sense that the voltage rails have reached their default target values, the DSP reset is de-asserted and the DSP begins executing code. If the power supplies do not reach the target values by the time PSWITCH is de-asserted, then the player returns to the off state.

The startup time for the DC-DC converters are dependent on battery voltage, but should be less than 100 milliseconds.

The crystal oscillator and RTC/ALARM can optionally remain off to minimize the standby power drain, see **HW_DCDC_PERSIST_SLEEP_XTAL_ENABLE**. Note that the default state of the crystal oscillator is powered down.

There is an integrated 5KΩ resistor that can be switched in between the VDDXTAL pin and the PSWITCH pin. If enabled, see **HW_DCDC_PERSIST_AUTORESTART**, then the device will immediately go through a power up sequence after power down, as if the PSWITCH had been pressed.

30.1.1.3. Power Down Sequence

Power Down is also controlled by the DC-DC converters. When the DC-DC converters detect a power down event, they return the player to the off state described above that holds the internal VddD and VddA supplies at ground, and holds the VddIO rail at ground in DCDC MODE 011; otherwise VddIO is connected to the battery. A Power Down event is triggered by writing a 1 to the Clock Control Register PWDN bit (**HW_CCR**, bit [17]). A Power Down event can also be triggered by a fast falling edge on the PSWITCH pin. The Power Down via PSWITCH is an edge sensi-

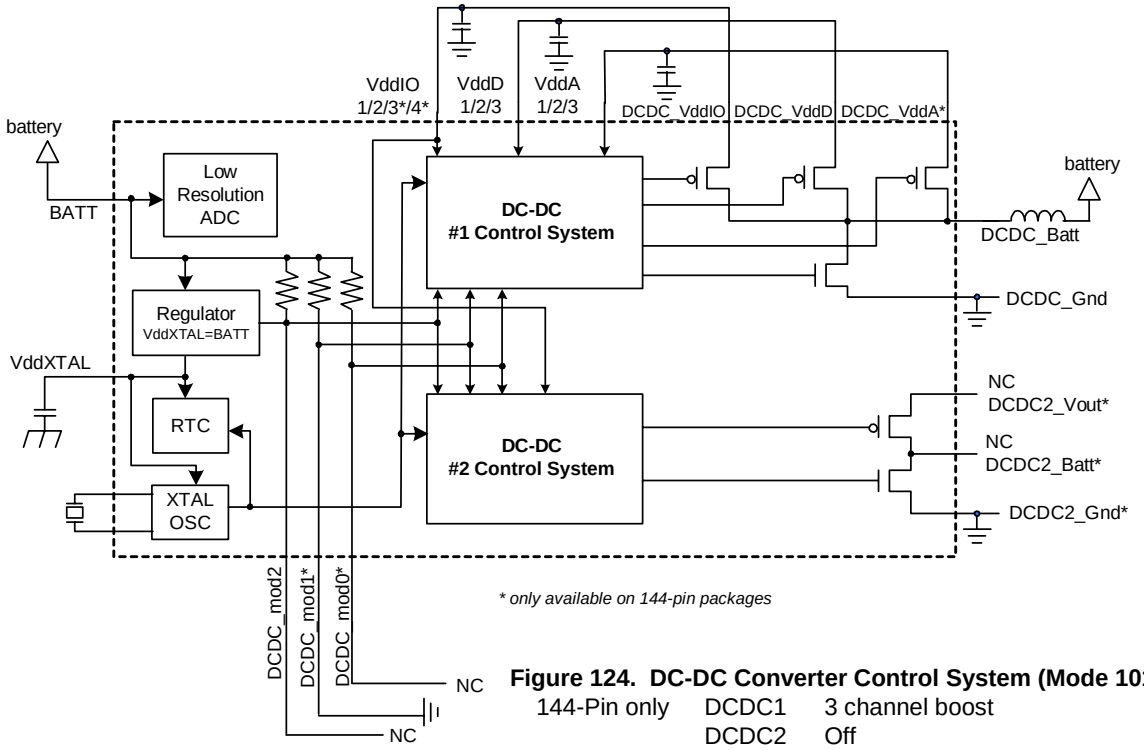


Figure 124. DC-DC Converter Control System (Mode 101)

144-Pin only	DCDC1	3 channel boost
	DCDC2	Off

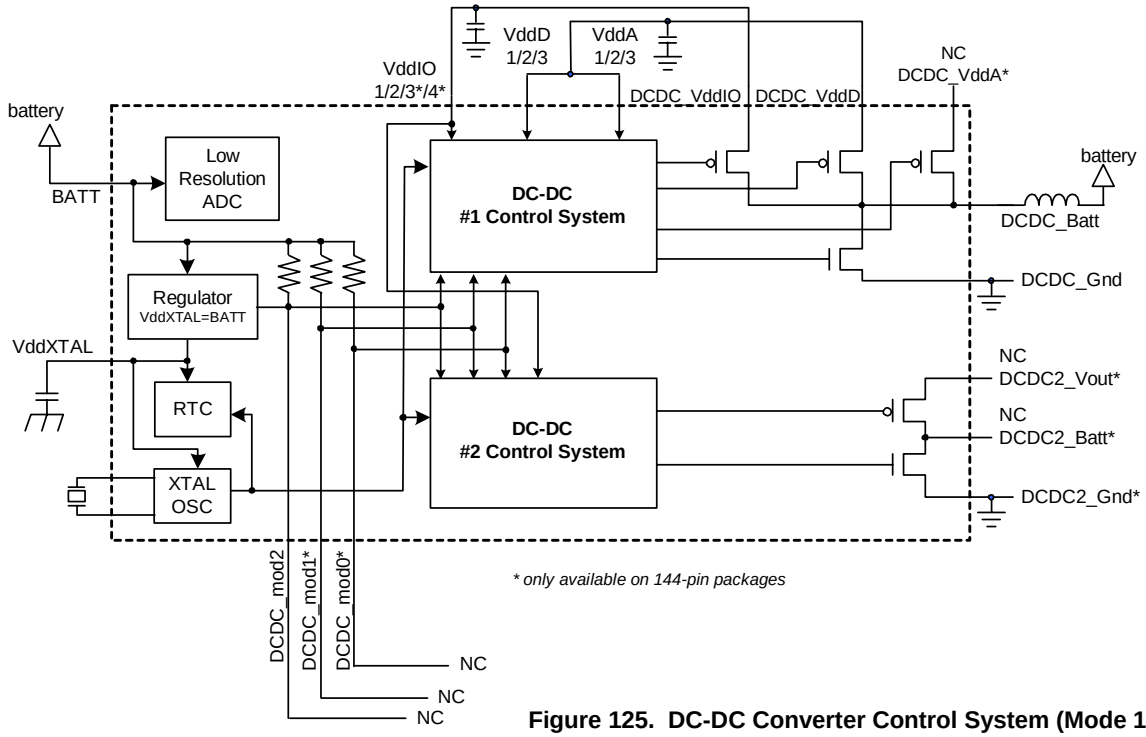


Figure 125. DC-DC Converter Control System (Mode 111)

100-Pin	DCDC1	2 channel boost
or 144-pin	DCDC2	Off

tive event and requires < 15ns falling edge to cause a Power Down to be detected. An external capacitor on PSWITCH can be used to prevent unwanted Power Down events in the final application, but this fast falling edge Power Down mechanism can also be disabled by writing a 1 to the Mixer Test Register PSRN field (**HW_MIXTBR_PSRN**).

30.1.1.4. Powered Down State

While the chip is powered down, the VddA & VddD rails are shorted to ground. The VddIO rail is either shorted to ground (DCDC MODE 011) or to the battery. The crystal oscillator and the RTC can continue to operate by drawing power from the BATT pin. The functions are implemented differently in each of the DCDC configurations.

For the DCDC Boost Modes (modes 101, & 111), the DCDC_VddD & DCDC_VddA pins are shorted to digital ground, which will short the VddA & VddD rails to ground. In the DCDC Buck Modes (modes 001 & 011), the DCDC_Batt pin is shorted to digital ground, which will also short the VddA & VddD rails to ground. In mode 000, the DCDC2_BATT pin is pulled to ground which will short VddIO to ground as well. For the DCDC external supply mode (mode 010), the DCDC_Batt pin is also shorted to digital ground, but since this pin is not connected, it will have no effect on the VddA & VddD rails.

For the DCDC Boost Modes (modes 101, & 111), the DCDC_VddIO & DCDC_Batt pins are shorted to each other, which will short the VddIO rail to the battery voltage. For the DCDC Buck Modes (modes 000, 001 & 011) the DCDC2_Batt and DCDC2_Vout pins are shorted to each other, which will also short the VddIO rail to the battery. For the DCDC external supply mode (mode 010), the DCDC2_Batt and DCDC2_Vout pins are also shorted to each other, but since these pins are not connected, it will have no effect on the VddIO rail.

For the DCDC Boost Modes (modes 101, & 111), the crystal oscillator and RTC take their power from the BATT pin, which is decoupled by a capacitor on the VddXTAL pin. For the DCDC Buck Modes (modes 000, 001 & 011) the crystal oscillator and RTC take their power from the BATT pin via a regulator which restricts the voltage to 50% of the battery voltage, which is decoupled by a capacitor on the VddXTAL pin. For the DCDC external supply mode (mode 010), the crystal oscillator and RTC also take their power from the BATT pin via a regulator which restricts the voltage to 50% of the battery voltage, which is decoupled by a capacitor on the VddXTAL pin.

30.1.1.5. Reset Sequence

A reset event can be triggered by writing the binary value 1101 to the Reset Control Register SRST field (HW_RCR, bits [7:4]). This reset only affects the digital logic, although the digital logic also includes all of the registers that control the analog portions of the chip. The DCDC converters continue to maintain the power supply rails during the reset, although the target voltages for the DCDC converters may change once the digital registers go to their reset values.

30.1.1.6. PSWITCH Recovery Mode Support

The Boot loader in the on-chip ROM contains several recovery mode options to allow new firmware to be downloaded to the device from the USB host. One method of invoking the recovery mode is to hold the PSWITCH closed for an extended period of time after power on. Currently, the ROM code looks for the PSWITCH to be held for 5 seconds after power on to force it into recovery mode. When the user first presses the PSWITCH, the rising edge seen on the PSWITCH pin triggers a

power up, as described above. Once the ROM code is executing it looks at the Boot Mode pins to determine the type of boot to perform, see Section 29. “BOOT MODES” on page 322. If one of the PSWITCH recovery boot modes is selected then the state of the PSWITCH pin is read from **HW_SPARER_PSWITH** to determine whether to enter the recovery mode.

30.2. Linear Mode Battery Charger

Several new circuits have been added to the STMP35xx to integrate battery charging for LiIon and NiMH batteries from a 5V source connected to the VDD5V pin. The source can come either from the USB’s VBUS or from a transformer. The first new circuit of interest is a five volt presence detector which trips when a voltage on the VDD5V pin exceeds the VDDIO rail by more than 0.6 V. Detection of this condition causes the activation of a linear regulator which drops the supplied 5 V down to 3.3 V. This regulator is tied to the VDD IO rail so it immediately begins to supply power to the rail. A second linear regulator is activated which drops the 3.3 V I/O rail to 1.8 V for the digital core VDD, finally a P-FET shorts the analog and digital core rails together so they both receive current from the linear regulator In addition, the presence of a 5 V source automatically stops the DC-DC converter(s) so that all chip power is supplied from the VDD5V pin.

If one sets **HW_VDD5V_POWER_CHARGE_BATT_CURRENT** to a non-zero value then the variable current shown in Figure 126 provides charging current to a LiIon battery. There is a voltage sensor monitoring the battery voltage which will automatically shut down the current source when the battery voltage reaches 4.1 V (4.2 V with **HW_VDD5V_POWER_CHARGE_LI_TYPE** set to one). One can programatically monitor the battery voltage using the Battery LRADC, see Section 28.1.1. “LRADC Battery Control Register” on page 311. LRADC2 can be used to monitor a temperature sensor for optimum battery charge profiles. There is an integrated current source for the external temperature sensor which can be enabled by setting **HW_MIX_TEST_TMP_CFG** to the desired bias current.

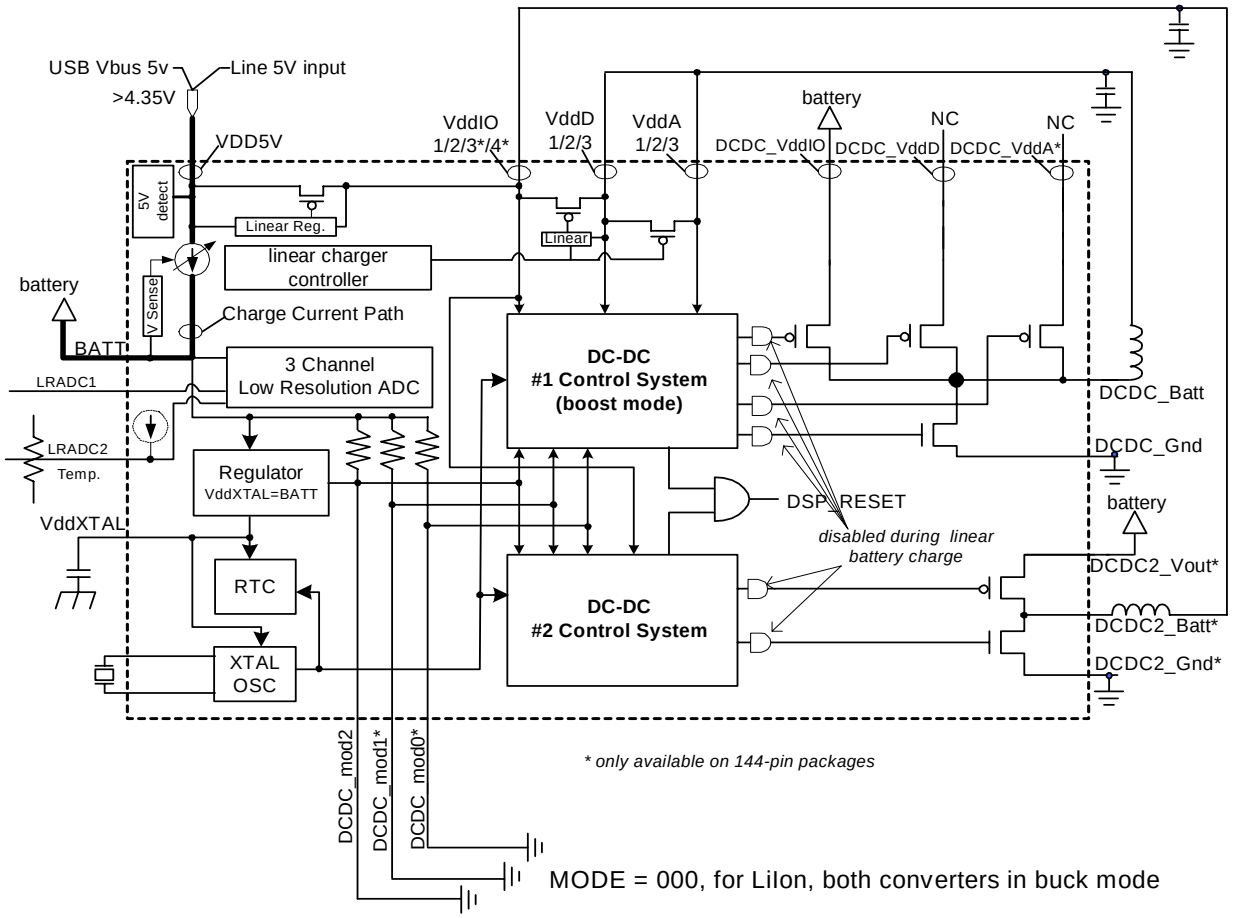


Figure 126. DC-DC Converter Linear Charger Theory of Operation

30.3. Switch Mode Battery Charger

An optional high efficiency switch mode battery charge method is available for NiMH batteries as shown in Figure 127, “DC-DC Converter Switch Mode Charger Theory of Operation” on page 345. When a five volt source is first applied to the VDD5V pin, the device converts to linear regulation as shown in Figure 126. Battery chargers are disabled unless explicitly turned on. To enable the switched mode charger, set **HW_VDD5V_POWER_CHARGE_SWCHRG_BAT** to one.

In this configuration, a current source provides charging current from the VDD5V pin directly to the VDD I/O Rail, disabling the 3.3 V linear regulator that supplies current to the VDD I/O rail in VDD5V sourced modes. DCDC converter #1 is enable to supply power to the I/O rail as well. Thus any excess current on the converter PFET flows back into the battery for charging. The digital and analog core rails are still strapped together in this mode and the VDD linear regulator is still active.

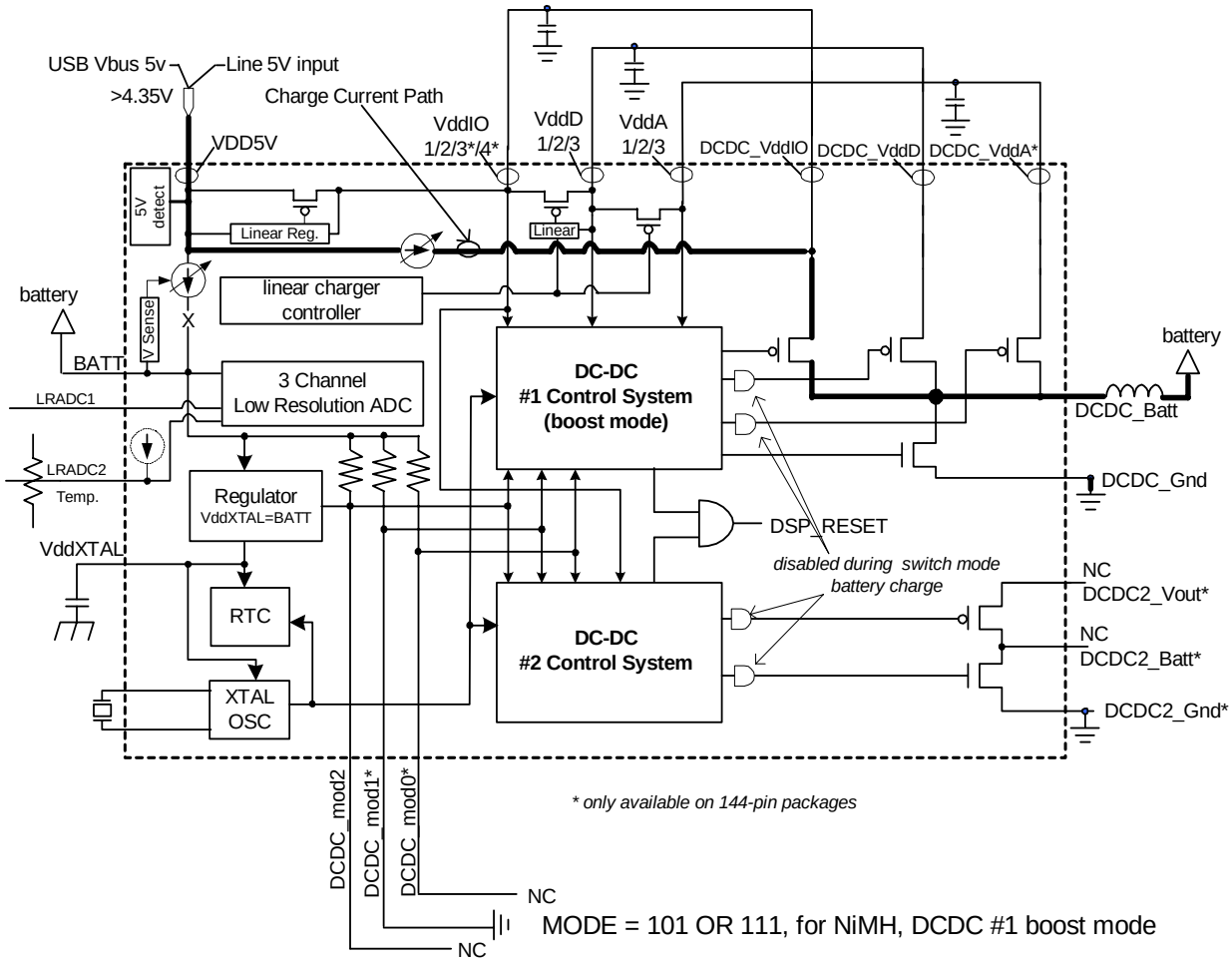


Figure 127. DC-DC Converter Switch Mode Charger Theory of Operation

30.4. Silicon Speed Sensor

The STMP35xx contains two silicon speed sensors to measure the performance characteristics of an individual die at its ambient temperature and process parametrics. Each sensor consists of a ring oscillator and a frequency counter. The ring oscillator runs on the VddD power rail. Therefore its frequency tracks the silicon performance as it changes in response to changes in the DCDC converter target voltage. The crystal oscillator is directly used as the precision time base for measuring the frequency of a ring oscillator. The ring oscillator is normally disabled. There is a seven bit counter connected to the ring oscillator which performs the frequency measurement. When a zero to one transition is detected on **HW_DCDCTBR_OSC1_START_COUNT**, then the counter is reset. The ring oscillator is next enabled for exactly one crystal clock period then disabled.

Thus the counter holds the number of cycles the ring oscillator was able to generate during one crystal clock period. The natural frequency of the ring oscillator strongly tracks the silicon process parametrics, i.e. faster silicon processes yield ring oscilla-

tors that run faster and thereby yield larger count values. The natural frequency tracks junction temperature effects on silicon speed as well. By taking a quick snapshot of the silicon sensor frequency one can lower the core digital rail voltages to a point more closely approaching the actual performance limit.

The application work load will define a minimum frequency of operation for the DSP and other devices on the DCLK. After device characterization, a calibration curve will be provided for the silicon speed sensor, see 30.7.6. “Clock Speed vs. Voltage” on page 366. Software can then translate this curve into a linear approximation represented by the following table:

DESIRED FREQUENCY	STARTING VDDD	MINIMUM COUNTER VALUE
12KHZ - 35MHZ	1.65 V (TBD)	TBD
35MHZ - 50MHZ	1.75 V (TBD)	TBD
50MHZ - 60MHZ	1.8 V (TBD)	TBD
65MHZ - 70MHZ	1.8 V (TBD)	TBD
75MHZ - 85MHZ	1.8 V (TBD)	TBD

Table 465. Silicon Speed Sensor

One heuristic for using the speed sensor is to select the desired new frequency of operation and set the VddD to the appropriate entry in the table. This value is conservative in that after characterization, there will still be sufficient margin that all die from across the process and temperature design will work at that voltage.

The software then goes into a slow adaptation loop to track the voltage to the lowest safe point for the process/temperature of each die. At 50ms intervals, the software will read the silicon speed sensor and compare it to the minimum safe value. If it is still above the safe threshold, it will implement a feedback loop filter to gradually lower the voltage to the threshold. As the temperature will change over time, the loop filter will also raise the voltage as required to stay in the safe region for the desired frequency.

30.5. Summary of Major DC-DC Features

The unique architecture of the integrated DC-DC converters on the device allows access for many control features via registers. Some of the most commonly used features are described below:

1. Digital adjustment of the voltage rails: The target values of the voltage rails can be adjusted during operation to optimize power consumption and extend battery life. A suggested clock speed vs. voltage curve to allow for the optimum settings to be chosen is included in 30.7.6 “Clock Speed vs. Voltage” on page 366.

   Note: In most DC-DC operating modes (except mode 101) the analog supply is shorted to the digital supply. In these cases analog performance may suffer if the supply voltage is reduced below 1.56 volts or so. If the supply voltage is affecting analog performance, alternate settings for the analog reference voltage and bandgap voltage may provide relief. The control for the analog reference voltages resides in the Reference Control Register (HW_REF_CTRL).

2. Power supply brownout detection: Digital comparators in the DC-DC converters provide immediate input to the DSP via an IRQB or NMI interrupt (controlled by the **HW_RCR** Reset Control Register) if the supplies drop below the brownout

level specified in the **HW_DCDC_VDDIO**, **HW_DCDC_VDDD**, **HW_DCDC_VDDA** Control Registers. This feature is generally used to alert the DSP when the power supplies are reaching dangerously low levels and allow a controlled system shutdown. These comparators may trip on brief transients below the target brownout voltage, so enough margin between the output voltage target and the brownout voltage target must be provided to eliminate false triggers. The magnitude of the difference between the output voltage target and brownout voltage target will depend on system power requirements and board layout, should typically be set to between 100 mV and 200 mV.

3. Battery brownout detection. Another comparator is present outside of the DC-DC converters to provide battery brownout information to the DSP via the Reset/Interrupt Control Register (**HW_RCR**). This feature is intended to provide indication to the DSP of a low battery voltage that requires a controlled system shutdown due to an event such as the battery falling out during application operation. The programming of the battery brownout level is via the Battery Low Resolution ADC Control Register (**HW_BATT_CTRL**).
4. DC-DC clock rate. The DC-DC converter clock rate can be lowered by 2x, 4x, or 8x from the default value of crystal frequency divided by 16 (1.536 MHz for a 24.576 MHz crystal or 1.5 MHz for a 24.0 MHz crystal). The clock rate can be lowered to reduce power consumption in the DC-DC converter if the corresponding increase in supply transients can be tolerated in the application. See bit 4, 5 in DC-DC Test Bit Register (**HW_DCDCTBR**).
5. Linear Regulator generates analog/digital supply. Another low-power option is provided by replacing the DC-DC converter #1 with a linear regulator to generate the 1.8 V supply during very low current modes (<5 mA). Enabling this regulator is described in the DCDC1 Control Register (**HW_DCDC_PWR_CHARGE**). It is possible to switch back and forth between the linear regulator and switching DC-DC converter.
6. Controlled linear charge current source supplying LiIon battery charge current from five volt source.
7. Controlled linear charge current source supplying NiMH battery charge current in a high efficiency switch mode charger.
8. Temperature sensor support for improved battery charge protocols.
9. Optional PFM mode for low load applications, see the DC-DC Test Bit Register (**HW_DCDCTBR**).
10. Silicon speed sensor to allow the supply voltage to be dynamically adjust to take advantage of silicon speed and temperature variations.

30.6. DC-DC Programmable Registers

The DC-DC control registers are used to control DC-DC low-level functionality. ***It is ill-advised to change these registers from their defaults unless SigmaTel specifically indicates otherwise.***

30.6.1. DCDC1 Control Register 0

The organization of the DCDC1 Control Register 0 is shown below.

HW_DCDC1_CTRL0 X:$FA0C

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			NLEV								PLVBO								PLVBU				

Table 466. HW_DCDC1_CTRL0

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	0	**Reserved –** Must be written with 0.
20:16	**NLEV**	RW	01001	**Negative Digital Loop Clip Level –** This value represents the negative value at which the digital loop filter clips to prevent illegal values. These five bits represent bits 18:14 of the 20-bit loop filter value. Bit 19 of the negative clip value is set to binary 1, while bits 13:0 are high. Altering the clip value should be done with caution since it may cause the sigma delta output to be non-monotonic. Default decodes to -376832. It is not recommended to increase the NLVL value when the value on the loop filter is equal to NLVL. This case occurs when the battery voltage is greater than the VddD voltage in mode 5 or mode7.
15:13	**RSRVD**	R	000	**Reserved –** Must be written with 0.
12:8	**PLVBO**	RW	01000	**Positive Digital Loop Clip Level in Boost Mode –** This value represents the positive value at which the digital loop filter clips to prevent illegal values while operating in boost converter mode. These five bits represent bits 18:14 of the 20-bit loop filter value. Bit 19 of the positive clip is set to binary 0, while bits 13:0 are low. The optimum value of this register depends on the battery voltage and acts effectively to limit the battery current when the load exceeds the capabilities of the converter. Default decodes to 131072.
7:5	**RSRVD**	R	000	**Reserved –** Must be written with 0.
4:0	**PLVBU**	RW	10111	**Positive Digital Loop Clip Level in Buck Mode –** This value represents the positive value at which the digital loop filter clips to prevent illegal values while operating in buck converter mode. These five bits represent bits 18:14 of the 20-bit loop filter value. Bit 19 of the positive clip is set to binary 0, while bits13:0 are low. The optimum value of this register depends on the battery voltage and acts effectively to limit the battery current when the load exceeds the capabilities of the converter. Default decodes to 376832.

Table 467. DCDC1 Control Register 0 Description

30.6.2. DCDC1 Control Register 1

The organization of the DCDC1 Control Register 1 is shown below.

HW_DCDC1_CTRL1 X:$FA0D

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
													FFOR			R				C			

Table 468. HW_DCDC1_CTRL1

BITS	LABEL	RW	RESET	DEFINITION
23:11	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10:8	**FFOR**	RW	000	**Feed Forward to loop filter value –** This two's complement value steps the loop filter value by the amount DDD once on a 000->DDD transition. Thus, this value must be rewritten to 000 every time before use. The feed forward feature is used to help the control loop react under heavy, but well-known, transient loads.
7:4	**R**	RW	1000	**Resistor Value –** This value represents the R value of the “resistor” in the digital loop filter in the common mode control loop in the DC-DC converter. The reset value of decimal 16384 is chosen in conjunction with the value for bits [3:0], the actual decoupling capacitor, and the clock rate to provide a stable control system. The value will rarely be altered, but conditions may arise in the actual application for which a different value is more optimum. Values range 2048 to 30720 (Only bits 14:11 are shown here, others bits are always 0). Values for NLVL, R, PLVU, and PLVD must satisfy the following restrictions: $NLVL - 4R \geq -475137$ $PLVU + 4R \leq 475136$ $PLVD + 4R \leq 475136$
3:0	**C**	RW	1000	**Capacitor 1/C Value –** This value represents 1/C value of the “capacitor” in the digital loop filter in the common mode control loop in the DC-DC converter. The reset value of decimal 32 is chosen in conjunction with the value for the R field in the DCDC1 Control Register 0 (**HW_DCDC1_CTRL0_R**), the actual external decoupling capacitor, and the clock rate to provide a stable control system. This value will rarely be altered, but conditions may arise in the actual application for which a different value is more optimum. Values can range between 8 and 120.

Table 469. DCDC1 Control Register 1 Description

30.6.3. DCDC2 Control Register 0

The organization of the DCDC2 Control Register 0 is shown below.

HW_DCDC2_CTRL0 X:$FA11

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
			NLEV								PLVBO								PLVBU				

Table 470. HW_DCDC2_CTRL0

BITS	LABEL	RW	RESET	DEFINITION
23:21	**RSRVD**	R	000	**Reserved –** Must be written with 0.
20:16	**NLEV**	RW	01001	**Negative Digital Loop Clip Level –** This value represents the negative value at which the digital loop filter clips to prevent illegal values. These five bits represent bits 18:14 of the 20-bit loop filter value. Bit 19 of the negative clip value is set to binary 1, while bits 13:0 are high. Altering the clip value should be done with caution since it may cause the sigma delta output to be non-monotonic. Default decodes to -376832.
15:13	**RSRVD**	R	000	**Reserved –** Must be written with 0.
12:8	**PLVBO**	RW	01000	**Positive Digital Loop Clip Level in Boost Mode –** This value represents the positive value at which the digital loop filter clips to prevent illegal values while operating in boost converter mode. These five bits represent bits 18:14 of the 20-bit loop filter value. Bit 19 of the positive clip is set to binary 0, while bits 13:0 are low. The optimum value of this register depends on the battery voltage and acts effectively to limit the battery current when the load exceeds the capabilities of the converter. Default decodes to 131072.
7:5	**RSRVD**	R	000	**Reserved –** Must be written with 0.
4:0	**PLVBU**	RW	10111	**Positive Digital Loop Clip Level in Buck Mode –** This value represents the positive value at which the digital loop filter clips to prevent illegal values while operating in buck converter mode. These five bits represent bits 18:14 of the 20-bit loop filter value. Bit 19 of the positive clip is set to binary 0, while bits 13:0 are low. The optimum value of this register depends on the battery voltage and acts effectively to limit the battery current when the load exceeds the capabilities of the converter. Default decodes to 376832.

Table 471. DCDC2 Control Register 0 Description

30.6.4. DCDC2 Control Register 1

The organization of the DCDC2 Control Register 1 is shown below.

HW_DCDC2_CTRL1 X:$FA12

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
													FFOR			R				C			

Table 472. HW_DCDC2_CTRL1

BITS	LABEL	RW	RESET	DEFINITION
23:11	**RSRVD**	R	0	**Reserved –** Must be written with 0.
10:8	**FFOR**	RW	000	**Feed Forward to loop filter value –** This two's complement value steps the loop filter value by the amount DDD once on a 000->DDD transition. Thus, this value must be rewritten to 000 every time before use. The feed forward feature is used to help the control loop react under heavy, but well-known, transient loads.
7:4	**R**	RW	1000	**Resistor Value –** This value represents the R value of the “resistor” in the digital loop filter in the common mode control loop in the DC-DC converter. The reset value of decimal 4096 is chosen in conjunction with the value for bits [21:15], the actual decoupling capacitor, and the clock rate to provide a stable control system. The value will rarely be altered, but conditions may arise in the actual application for which a different value is more optimum. Values range 256 to 32767 (Only bits 14:8 are shown here, others bits are always 0). Values for NLVL, R, PLVU, and PLVD must satisfy the following restrictions: $$NLVL - 4R \geq -475137$$ $$PLVU + 4R \leq 475136$$ $$PLVD + 4R \leq 475136$$
3:0	**C**	RW	1000	**Capacitor 1/C Value –** This value represents 1/C value of the “capacitor” in the digital loop filter in the common mode control loop in the DC-DC converter. The reset value of decimal 64 is chosen in conjunction with the value for **HW_DCDC2_CTRL1_R**, the actual external decoupling capacitor, and the clock rate to provide a stable control system. This value will rarely be altered, but conditions may arise in the actual application for which a different value is more optimum. Values can range between 8 and 120.

Table 473. DCDC2 Control Register 1 Description

30.6.5. DC-DC VddIO Control Register

The organization of the DC-DC VddIO Control Register is shown below.

HW_DCDC_VDDIO X:$FA0E

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		VDDIO_OK	BROWNOUT_STATUS				BROWNOUT_ENABLE				BROWNOUT_LEVEL								VOLTAGE_LEVEL				

Table 474. HW_DCDC_VDDIO

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**VDDIO_OK**	R	-	**Convergence State** 0 converter still converging to new setting 1 converged on new voltage setting
20	**BROWNOUT_STATUS**	R	-	**VddIO brownout detect status** 0 No brownout detected on VddIO rail 1 Brown out detected on VddIO rail
19:17	**RSRVD**	R	00	**Reserved –** Must be written with 0.
16	**BROWNOUT_ENABLE**	RW	0	**VddIO brownout detect enable** 0 VddIO brownout detect disabled 1 VddIO brownout detect enabled
15:13	**RSRVD**	R	00	**Reserved –** Must be written with 0.
12:8	**BROWNOUT_LEVEL**	RW	01100	**On-chip VddIO brownout detect level** 11111 4.03 V 10111 3.52 V 01111 3.01 V 00111 2.50 V 11110 3.96 V 10110 3.45 V 01110 2.94 V 00110 2.43 V 11101 3.90 V 10101 3.39 V 01101 2.88 V 00101 2.37 V 11100 3.84 V 10100 3.33 V **01100 2.82 V** 00100 2.30 V 11011 3.78 V 10011 3.26 V 01011 2.75 V 00011 2.24 V 11010 3.71 V 10010 3.20 V 01010 2.69 V 00010 2.18 V 11001 3.65 V 10001 3.14 V 01001 2.62 V 00001 2.11 V 11000 3.58 V 10000 3.07 V 01000 2.56 V 00000 2.05 V
7:5	**RSRVD**	R	000	**Reserved –** Must be written with 0.
4:0	**VOLTAGE_LEVEL**	RW	10000	**On-chip VddIO voltage level –** The voltage level settings for this field are the same as for the **BROWNOUT_LEVEL** field above. Default Voltage is 3.07 Volts.

*Note: **BROWNOUT_LEVEL** and **VOLTAGE_LEVEL** represent nominal on-chip voltages. Due to random offsets, board layout, and ohmic drops, the measured voltages on the board will not always measure exactly as the above table decodes.*

Table 475. DC-DC VddIO Control Register Description

30.6.6. DC-DC VddD Control Register

The organization of the DC-DC VddD Control Register is shown below.

HW_DCDC_VDDD X:$FA0F

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		VDDD_OK	BROWNOUT_STATUS				BROWNOUT_ENABLE				BROWNOUT_LEVEL								VOLTAGE_LEVEL				

Table 476. HW_DCDC_VDDD

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**VDDD_OK**	R	-	**Convergence State** 0 converter still converging to new setting 1 converged on new voltage setting
20	**BROWNOUT_STATUS**	R	-	**VddD brownout detect status** 0 No brownout detected on VddD rail 1 Brown out detected on VddD rail
19:17	**RSRVD**	R	00	**Reserved –** Must be written with 0.
16	**BROWNOUT_ENABLE**	RW	0	**VddD brownout detect enable** 0 VddD brownout detect disabled 1 VddD brownout detect enabled
15:13	**RSRVD**	R	00	**Reserved –** Must be written with 0.
12:8	**BROWNOUT_LEVEL**	RW	10000	**On-chip VddD brownout detect level** 11111 2.02 V 10111 1.76 V 01111 1.50 V 00XXX RSRVD 11110 1.98 V 10110 1.73 V 01110 1.47 V 11101 1.95 V 10101 1.70 V 01101 1.44 V 11100 1.92 V 10100 1.66 V 01100 1.40 V 11011 1.89 V 10011 1.63 V 01011 1.37 V 11010 1.85 V 10010 1.60 V 01010 1.34 V 11001 1.82 V 10001 1.57 V 01001 1.31 V 11000 1.79 V **10000 1.54 V** 01000 1.28 V
7:5	**RSRVD**	R	000	**Reserved –** Must be written with 0.
4:0	**VOLTAGE_LEVEL**	RW	10110	**On-chip VddD voltage level –** The voltage level settings for this field are the same as for the **BROWNOUT_LEVEL** field above. Default Voltage is 1.73 Volts.

*Note: **BROWNOUT_LEVEL** and **VOLTAGE_LEVEL** represent nominal on-chip voltages. Due to random offsets, board layout, and ohmic drops, the measured voltages on the board will not always measure exactly as the above table decodes.*

Table 477. DC-DC VddD Control Register Description

30.6.7. DC-DC VddA/Battery Brownout Enable Control Register

The organization of the DC-DC VddA Control Register is shown below.

HW_DCDC_VDDA X:$FA10

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		VDDA_OK	BROWNOUT_STATUS				BROWNOUT_ENABLE				BROWNOUT_LEVEL								VOLTAGE_LEVEL				

Table 478. HW_DCDC_VDDA

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**VDDA_OK**	R	-	**Convergence State** 0 converter still converging to new setting 1 converged on new voltage setting
20	**BROWNOUT_STATUS**	R	-	**VddA brownout detect status** 0 No brownout detected on VddA rail 1 Brown out detected on VddA rail
19:17	**RSRVD**	R	00	**Reserved –** Must be written with 0.
16	**BROWNOUT_ENABLE**	RW	0	**VddA brownout detect enable** 0 VddA brownout detect disabled 1 VddA brownout detect enabled
15:13	**RSRVD**	R	00	**Reserved –** Must be written with 0.
12:8	**BROWNOUT_LEVEL**	RW	10000	**On-chip VddA brownout detect level** 11111 2.02 V 10111 1.76 V 01111 1.50 V 00XXX RSRVD 11110 1.98 V 10110 1.73 V 01110 1.47 V 11101 1.95 V 10101 1.70 V 01101 1.44 V 11100 1.92 V 10100 1.66 V 01100 1.40 V 11011 1.89 V 10011 1.63 V 01011 1.37 V 11010 1.85 V 10010 1.60 V 01010 1.34 V 11001 1.82 V 10001 1.57 V 01001 1.31 V 11000 1.79 V **10000 1.54 V** 01000 1.28 V
7:5	**RSRVD**	R	000	**Reserved –** Must be written with 0.
4:0	**VOLTAGE_LEVEL**	RW	10110	**On-chip VddA voltage level –** The voltage level settings for this field are the same as for the **BROWNOUT_LEVEL** field above. Default Voltage is 1.73 Volts.

*Note: **BROWNOUT_LEVEL** and **VOLTAGE_LEVEL** represent nominal on-chip voltages. Due to random offsets, board layout, and ohmic drops, the measured voltages on the board will not always measure exactly as the above table decodes.*

Table 479. DC-DC VddA Control Register Description

30.6.8. Silicon Speed Sensor Register

The organization of the Silicon Speed Sensor Register is shown below.

HW_SISPEED X:$FA13

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	RINGOSC2									OSC2_START_COUNT	OSC2_PWRUP		RINGOSC1									OSC1_START_COUNT	OSC1_PWRUP

Table 480. HW_SISPEED

BITS	LABEL	RW	RESET	DEFINITION
23	**RSRVD**	R	0	**Reserved –** Must be written with 0.
22:16	**RINGOSC2**	R		If **OSC2_PWRUP** is set to one then this read only bit field returns the frequency count value from ring oscillator 2. If **OSC2_PWRUP** is set to zero then it returns the upper bits of DCDC2 converter loop filter Duty Cycle value. **Frequency Count –** The crystal oscillator is used as a time base to measure the frequency of ring oscillator 2. The measured frequency is returned here. **Duty Cycle –** Digital loop filter value representing the duty cycle of charging /discharging the inductor. 100000 DC-DC duty cycle at minimum (small charge time/large discharge time) ..... 011111 DC-DC duty cycle at maximum (large charge time/ small discharge time) This value will be limited according to the **NLEV** and **PLBO**/**PLVBU** values set for this DC-DC Converter.
15:14	**RSRVD**	R	00	**Reserved –** Must be written with 0.
13	**OSC2_START_COUNT**	RW	0	Execute ring oscillator counting on low to high transition of this bit and change meaning of **HW_SISPEED_RINGOSC2**. Note that **OSC2_PWRUP** must be set to one to power on the oscillator before counting.
12	**OSC2_PWRUP**	RW	0	Set to one to power up ring oscillator 2 circuitry. Set to zero for low power mode.
11	**RSRVD**	R	0000	**Reserved –** Must be written with 0.

Table 481. Silicon Speed Sensor Register Description

BITS	LABEL	RW	RESET	DEFINITION
10:4	**RINGOSC1**	R		If **OSC1_PWRUP** is set to one then this read only bit field returns the frequency count value from ring oscillator 1. If **OSC1_PWRUP** is set to zero then it returns the upper bits of DCDC1 converter loop filter Duty Cycle value. **Frequency Count –** The crystal oscillator is used as a time base to measure the frequency of ring oscillator 1. The measured frequency is returned here. **Duty Cycle –** Digital loop filter value representing the duty cycle of charging /discharging the inductor. 100000 DC-DC duty cycle at minimum (small charge time/large discharge time) ..... 011111 DC-DC duty cycle at maximum (large charge time/ small discharge time) This value will be limited according to the **NLEV** and **PLBO**/**PLVBU** values set for this DC-DC Converter.
3:2	**RSRVD**	R	00	**Reserved –** Must be written with 0.
1	**OSC1_START_COUNT**	RW	0	Execute ring oscillator counting on low to high transition of this bit and change meaning of **HW_SISPEED_RINGOSC1**. Note that **OSC1_PWRUP** must be set to one to power on the oscillator before counting.
0	**OSC1_PWRUP**	RW	0	Set to one to power up ring oscillator 1 circuitry. Set to zero for low power mode.

Table 481. Silicon Speed Sensor Register Description (Continued)

30.6.9. DC-DC Test Bit Register

The organization of the DC-DC Test Bit Register is shown below.

HW_DCDCTBR X:$FA14

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DCDC2_STOPCLK	DCDC2_PFM	DCDC2_VDD5V_ACTIVE	DCDC2_CLK2X	DCDC2_CLK4X	DCDC2_DIS_5BIT	DCDC1_HALF_FETS	DCDC1_NEW_SCHEME	DCDC_ANA_BGR_BIAS	DCDC_MORE_CAP	DCDC_ANA_LESSI	PWRUP_VDDIO_BRNOUT	DCDC1_STOPCLK	DCDC1_PFM	DCDC1_VDD5V_ACTIVE	DCDC1_NOZERO	DCDC1_CLK2X	DCDC1_CLK4X	DCDC1_BAT_ADJ	DCDC1_DIS_5BIT	DCDC1_ADJ_TN			

Table 482. HW_DCDCTBR

BITS	LABEL	RW	RESET	DEFINITION
23	**DCDC2_STOPCLK**	RW	0	Set to one to disable the clock to the DCDC2 converter. Only use this when power comes from the VDD5V pin to reach minimum system power or DCDC2 is unused.
22	**DCDC2_PFM**	RW	0	Set to one to enable a pulse skip mode (pulse frequency modulation in DCDC #2) that is useful in very lightly loaded conditions to minimize system power.
21	**DCDC2_VDD5V_ACTIVE**	RW	0	Set to one to force DCDC #2 to remain active, even with 5V attached to the device. This mode may help during the 5V unplug transition back to battery power supplied by the DCDC converter. When 5V is not present and the mode is NOT 101 or 11, this bit prevents pass through behavior when the battery voltage is approximately equal to the output voltage.
20	**DCDC2_CLK2X**	RW	0	Set to one to slow down the DCDC #2 clock frequency by a factor of 1/2. Can be combined with **DCDC2CLK4X**.
19	**DCDC2_CLK4X**	RW	0	Set to one to slow down the DCDC #2 clock frequency by a factor of 1/4. Can be combined with **DCDC2CLK2X**.
18	**DCDC2_DIS_5BIT**	RW	0	Set to one to reduce the quantization of DC-DC switching times from 5 bits to 3 bits for DCDC #2.
17	**DCDC1_HALF_FETS**	RW	0	Set to one to reduce the size of the FETs by half (disable half of the FETS) for DCDC converter #1. This mode may be useful to minimize total system power in lightly loaded applications.
16	**DCDC1_NEW_SCHEME**	RW	0	Set to one to use the new switching scheme for switching FETS in DCDC converter #1 buck mode.
15	**DCDC_ANA_BGR_BIAS**	RW	0	Set to one to switch the DCDC analog bias current from the selfbias current to a bandgap derived current.
14	**DCDC_MORE_CAP**	RW	0	Set to one to add additional capacitance to common mode sense node.
13	**DCDC_ANA_LESSI**	RW	0	Set to one to reduce the bias current to the DCDC analog.

Table 483. DC-DC Test Bit Register Description

BITS	LABEL	RW	RESET	DEFINITION
12	**PWRUP_VDDIO_BRNOUT**	RW	0	Set to one to enable the VDDIO brownout comparator. NOTE: the VDDD brownout comparator is on by default.
11	**DCDC1_STOPCLK**	RW	0	Set to one to disable the clock to the DCDC1 converter. Only use this when power comes from the 5V pin to reach minimum system power.
10	**DCDC1_PFM**	RW	0	Set to one to enable a pulse skip mode (pulse frequency modulation for DCDC #1) that is useful in very lightly loaded conditions to minimize system power.
9	**DCDC1_VDD5V_ACTIVE**	RW	0	Set to one to force DCDC #1 to remain active, even with 5V attached to the device. This mode may help during the 5V unplug transition back to battery power supplied by the DCDC converter. When 5V is not present and the mode is NOT 101 or 111, this bit prevents pass through behavior when the battery voltage is approximately equal to the output voltage. When the chip is configured for mode 101 or 111, then setting this bit to one allows VDD to be less than the battery voltage by using a PFM mode switching algorithm.
8	**DCDC1_NOZERO**	RW	0	Set to one to prevent DCDC converter #1 from a duty cycle of zero.
7	**DCDC1_CLK2X**	RW	0	Set to one to slow down the DCDC #1 clock frequency by a factor of 1/2. Can be combined with **DCDC1CLK4X**.
6	**DCDC1_CLK4X**	RW	0	Set to one to slow down the DCDC #1 clock frequency by a factor of 1/4. Can be combined with **DCDC1CLK2X**.
5	**DCDC1_BAT_ADJ**	RW	0	Set to one to use battery LRADC information to scale VDDIO duty cycle in DCDC #1. This is only used in mode 101 or 111.
4	**DCDC1_DIS_5BIT**	RW	0	Set to one to reduce the quantization of DC-DC switching times from 5 bits to 3 bits for DCDC #1.
3:0	**DCDC1_ADJ_TN**	RW	0000	Adjust duty cycle of DCDC #1 VDDIO output in boost mode. This would be based on information supplied by the battery LRADC.

Table 483. DC-DC Test Bit Register Description (Continued)

30.6.10. DC-DC Persistent Bit Register

The organization of the DC-DC Persistent Register is shown below. This register has a shadow copy that is in the DCLK clock and DSP power domain. It accessible by the DSP. There is a master copy of this register in the VDDXTAL clock and power domain. The persistent bits control various function states during power down modes of operation. Writing the shadow copy from the DSP does not automatically update the master copy. One must preload the shadow bits with desired values and then toggle the **UPDATE** bit to copy the values from the shadow version to the master version.

HW_DCDC_PERSIST X:$FA1B

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
											AUTO_RESTART_STAT	DELAY_5V_AUTO_RESTART_STAT	UDPATE	AUTO_RESTART	DELAY_5V_AUTO_RESTART	LOW_BATT_TYPE	LOW_BATTERY_ENABLE	SLEEP_XTAL_ENABLE	XTAL_TRIM1	XTAL_TRIM0	XTAL_BIAS_DOWN1	XTAL_BIAS_DOWN0	XTAL_TRIM_ENABLE

Table 484. HW_DCDC_PERSIST

BITS	LABEL	RW	RESET	DEFINITION
23:13	**RSRVD**	R	0	**Reserved –** Must be written with 0.
12	**AUTO_RESTART_STAT**	R	0	This read only bit shows the state of the **AUTO_RESTART** bit as it was last copied (by toggling **UPDATE**) into the master register bit in the XTAL clock and power domain. NOTE: Immediately after power up, **AUTO_RESTART_STAT** reflects the persistent state, whereas **AUTO_RESTART** will have just been reset to zero.
11	**DELAY_5V_AUTO_RESTART_STAT**	R	0	This read only bit shows the state of the **DELAY_5V_AUTO_RESTART** bit as it was last copied (by toggling **UPDATE**) into the master register bit in the XTAL clock and power domain. NOTE: Immediately after power up, **DELAY_5V_AUTO_RESTART_STAT** reflects the persistent state, whereas **DELAY_5V_AUTO_RESTART** will have just been reset to zero.
10	**UPDATE**	RW	0	A zero to one transition on this bit causes the shadow values of the **HW_DCDC_PERSIT** bits to be copied to the master register in the VDDXTAL clock and power domain.

Table 485. DC-DC Persistent Bit Register Description

BITS	LABEL	RW	RESET	DEFINITION
9	**AUTO_RESTART**	RW	0	Set this bit to one to enable a 5KΩ resistor between the VDDXTAL pin and the PSWITCH pin. When the DCDC converter powers down the chip either because the power down bit was set in **HW_CCR** or because the 5V source was removed, it waits for the power switch to be pressed. Enabling the 5KΩ resistor over-rides the power switch and makes it behave as if it had just been pressed.
8	**DELAY_5V_AUTO_RESTART**	RW	0	The DCDC converter uses two primary methods to trigger a DSP restart. The first waits for the power switch to be pressed (or the 5KΩ resistor switched in by **AUTO_RESTART**). The second trigger is generated when a voltage greater than VDDIO appears on the 5V pin. Customers should not set this bit to one without explicit advice from SigmaTel.
7	**LOW_BATT_TYPE**	RW	0	Set to one to select LiIon specific threshold for low battery detection. Set to zero for other battery chemistries.
6	**LOW_BATTERY_ENABLE**	RW	0	Set to one to enable the low battery detect function. The crystal oscillator and RTC can operate on the VDDXTAL power domain even while the rest of the chip is powered down. Setting this bit to one enables a brownout detector for the VDDXTAL domain so that the RTC and Persistent bits will not be scrambled in an under voltage event.
5	**SLEEP_XTAL_ENABLE**	RW	0	Set to one to allow crystal oscillator to remain on during power down state. NOTE: default turns the crystal off. It must be turned on to use the Real Time Clock and Alarm. **NOTE: HW_RTC_PERSIST0_XTAL_PDN** must also be set to zero to enable crystal oscillator operation during power down states.
4	**XTAL_TRIM1**	RW	0	Set to one to add four pF capacitance to XTAL0/XTALI pins.
3	**XTAL_TRIM0**	RW	0	Set to one to add four more pF capacitance to XTAL0/XTALI pins. Be sure to turn on **XTAL_TRIM1** first.
2	**XTAL_BIAS_DOWN1**	RW	0	Set to one to shrink the crystal current 30%.
1	**XTAL_BIAS_DOWN0**	RW	0	Set to one to shrink the crystal current another 30%. Be sure to turn on **XTAL_BIAS_DOWN1** first.
0	**XTAL_TRM_ENABLE**	RW	0	Set to one to enable a resistor between the crystal capacitors. This allows the caps. added by **XTAL_TRIM0** and **XTAL_TRIM1** to start at the correct DC level to avoid a clock glitch when adding the caps.

Table 485. DC-DC Persistent Bit Register Description (Continued)

30.6.11. Power Charger Register

This register controls the 5 Volt power source mode. In this mode, the power charger can take power from the 5 Volt input from the USB cable or from a wall transformer for normal operation of the chip. The STMP35xx can optionally supply energy to recharge the battery in 5 Volt power mode.

HW_VDD5V_PWR_CHARGE X:$FA1D

23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
		BATT_INFO	VDD5V_PRESENT			SWCHRG_BAT	DRV_BAT	TEST_USBREGS	DCANA_LP	PWDN_ON_IOBRNOUT	DISABLE_ILIMIT	LI_TYPE	\	RES	PWD				BATT_CURRENT				

Table 486. HW_VDD5V_PWR_CHARGE

BITS	LABEL	RW	RESET	DEFINITION
23:22	**RSRVD**	R	0	**Reserved –** Must be written with 0.
21	**BATT_INFO**	R	-	This read only test is intended for chip validation and software debug but should not be used in software charging algorithms. This bit is set to one when the linear charger is supplying current to the battery. It is set to zero when the charging current falls below approximately 0.5 mA.
20	**VDD5V_PRESENT**	R	-	This read-only status bit is set to one when the 5V input pin voltage is greater than the VDDIO rail voltage.
19:18	**RSRVD**	R	00	**Reserved –** Must be written with 0.
17	**SWCHRG_BAT**	RW	0	Set to one to enable the DCDC converter to supply energy to the NiMH battery using its switching mode for higher efficiency charging.
16	**DRV_BAT**	RW	0	Set this SigmaTel test mode bit to one to connect the VDDD rail to the BATT pin.
15	**TEST_USBREGS**	RW	0	Set to one to enable the 5V sourced linear regulators. Set to zero for normal operation, wherein the 5V sensor controls the linear regulators. NOTE: this is a SigmaTel only test function.
14	**DCANA_LP**	RW	0	Set to one to power down unused circuitry in the DCDC converter. This is useful for minimizing current in USB standby mode.
13	**PWDN_ON_IOBRNOUT**	RW	0	Reset the system when a brownout event is detected on the VDDIO rail. This will commonly occur when the system which has been drawing current from the USB 5V bus is suddenly unplugged. Set to one to enable system reset on brownout detection. Set to one to allow full use of the 100mA provided from the USB cable. ***WARNING***: setting this bit to one enables the linear regulators and shuts down the DCDC converter, even if VDD5V is not connected to a power source. Always check for 5 V presence on VDD5V before setting this bit to one.

Table 487. VDD5V Power Charger Register Description

BITS	LABEL	RW	RESET	DEFINITION
12	**DISABLE_ILIMIT**	RW	0	Set to one to disable In-rush current limiter on the linear charger. Set to zero to enable the current limiter. This circuit is present to improve/meet the USB 2.0 in-rush current specification.
11	**LI_TYPE**	RW	0	Linear Charger LiIon protocol: Set to zero for a 4.2 volt LiIon battery. Set it to one for a 4.1 volt LiIon battery.
10	**RSRVD**	R	000	**Reserved –** Must be written with 0.
9	**RES**	RW	0	When this bit is set to zero the charging current is derived using the accurate off-chip resistor shared with the integrated USB 2.0 PHY. When set to one, an on-chip bias is used.
8	**PWD**	RW	01	Battery charger powerdown mode, turn off all battery charge current except those needed to avoid over voltage when 5V is sensed.
7:5	**RSRVD**	R	000	**Reserved –** Must be written with 0.
4:0	**BATT_CURRENT**	RW	00000	These bits control the magnitude of the battery charge current. For the linear regulator, these values are: 00000 = OFF 00001 = 10 mA 00010 = 25 mA 00011 = 35 mA 00100 = 50 mA 01000 = 100 mA 10000 = 200 mA 11111 = 385 mA Currents are additive for other combinations. When switched mode charger is used (NiMH) (**HW_VDD5V_PWR_CHARGE_SWCHRG_BAT = 1**) then the battery current = (charge current from table above) * 0.8 * (charger efficiency) * VDDIO/Vbattery. **NOTE**: set **HW_USBPHYPWD_PWDIBIAS** to zero and set **HW_REF_CTRL_LWREF** to zero for proper operation.

Table 487. VDD5V Power Charger Register Description (Continued)

Charger current limitations due to power dissipation, electromigration limits and or temperature limits will be determined during chip characterization.

30.7. DC-DC Converter Efficiency

NOTE: Data in this section will be updated after chip validation and characterization are complete.

The following table shows the typical on resistances of the power FETs in both DC-DC converters. These are nominal values that include package parasitic resistance in the TQFP package. The fpBGA package has significantly more parasitic resistance than the TQFP, so resistance values may be approximately 100-200 mΩ higher in the fpBGA.

DC-DC #1	NFET	0.5 Ω
DC-DC #1	PFET connected to VddA	0.7 Ω
DC-DC #1	PFET connected to VddD	0.7 Ω
DC-DC #1	PFET connected to VddIO	1.1 Ω
DC-DC #2	NFET	0.5 Ω
DC-DC #2	PFET	0.7 Ω

Table 488. Typical Resistance of Power Fets on DC-DC Converters

The following graphs show typical efficiency plots vs. the battery voltage for the DC-DC converters configured in different modes. These graphs show measurements made with typical devices at room temperature with specific static loads. Therefore, these graphs should not be interpreted as specifications, but as estimates of typical efficiencies under nominal conditions.

30.7.1. DCDC1 Mode111 Efficiency

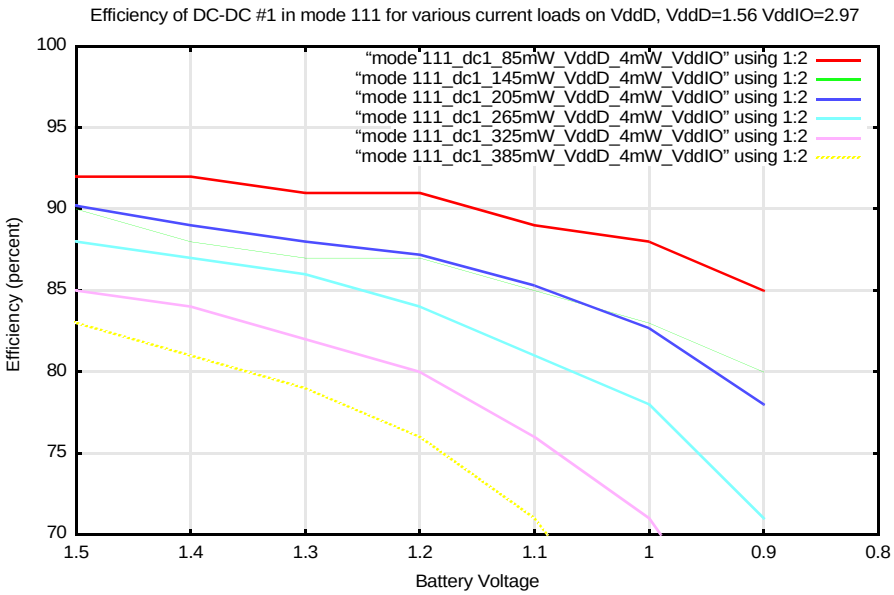


Figure 128. Efficiency of DC-DC #1 in mode 111 with VddIO Constant

These two graphs show the efficiency of the DC-DC converter (#1 in mode 111) versus battery voltage for several values of output power. The test conditions for the two graphs are identical except for the load the power is delivered to. Because the power FET that connects the inductor to the DCDC_VDDIO output has a higher

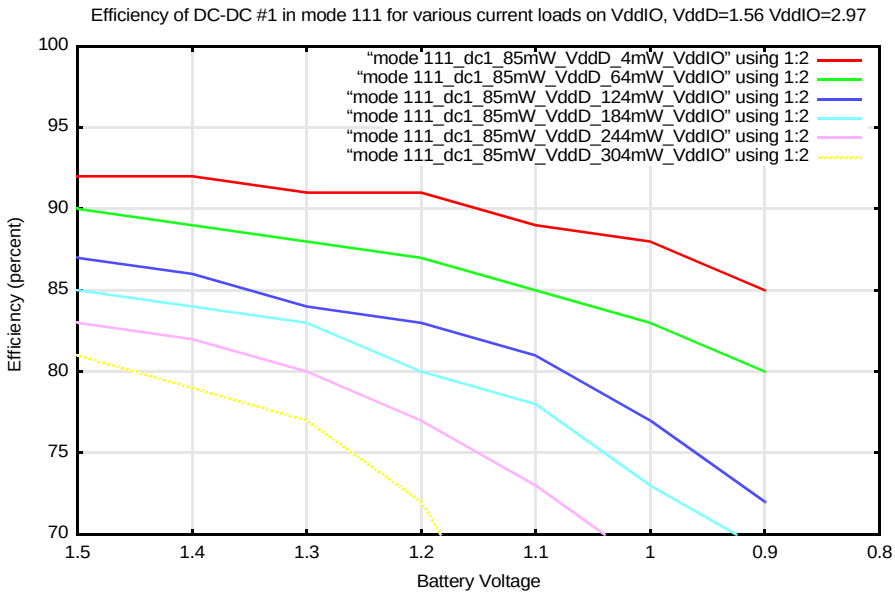


Figure 129. Efficiency of DC-DC #1 in mode 111 with VddD Constant

resistance than the FET that connects the inductor to the DCDC_VDDD output, the efficiency of the DC-DC converter is improved when the same power is delivered to the low voltage output, VddD, relative to the high voltage output, VddIO.

It should be noted that the boost configuration of DC-DC #1 (mode 101 and mode 111) could be the least efficient configurations for two reasons. First, Boost Mode power conversion only transfers power to the load during one phase of the converters charge/discharge cycle. Therefore, conservation of charge requires an inductor current that is higher than the load current, so I^2R losses in the power FETs that switch the inductor are increased relative to a Buck Mode configuration where current is transferred to the load during the entire charge/discharge cycle. Secondly, since mode 101 and mode 111 generate multiple outputs using one external inductor, the inductor current must increase to support multiple load currents. Thus, I^2R losses are also increased when using the multiple output configurations relative to the configurations that have one inductor per output voltage. Therefore, mode 101 and mode 111 are primarily intended for lower power applications that use a single battery and single inductor to achieve an ultra small form factor.

30.7.2. DCDC2 Mode001 Efficiency

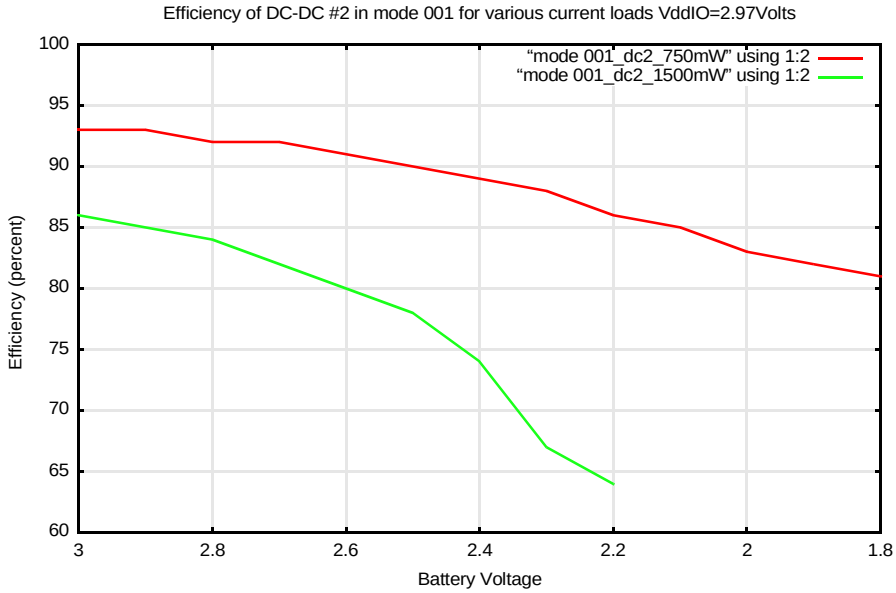


Figure 130. Efficiency of DC-DC #2 in mode 001 for various current loads

This graph shows the efficiency of DC-DC #2 for two values of output power. Since DC-DC #2 has a PFET with less on-resistance than the comparable DC-DC #1 PFET, and this configuration is a single load converter, this mode is preferred for higher power VddIO loads than mode 111 or mode 101.

30.7.3. Max Power Out in Boost Mode

For a Boost Mode converter, the power delivered to the load can be estimated using the following equation:

$$P_{OUT} = \frac{V_{OUT}(V_{BAT}(1-D) - V_{OUT}(1-D)^2)}{R_P + R_L + D(R_N - R_P)}$$

In this equation "D" represents the duty cycle of DC-DC converter, Vout represents the output voltage of the converter, Rn and Rp represent the on resistance of the power fets appropriate for the particular DC-DC converter configuration, and Rl represents the on-resistance of the DC-DC inductor. For multiple output modes 111 and 101, this equation is a reasonable approximation if it is assumed that all the power is transferred to one of the loads. This equation leads to the following expression for the duty cycle that gives to the maximum power transferred to the load:

$$D_{MAX} = \frac{R_P - R_N\sqrt{1 + \frac{V_{BAT}}{V_{OUT}} \times \frac{(R_P - R_N)}{R_N}}}{R_P - R_N}$$

Substituting Dmax into the equation for Pout gives a maximum power that can be theoretically delivered to the load. Note that this equation was derived using only V=Ldi/dt and conservation of charge, and thus represents a theoretical maximum that neglects many significant sources of loss including frequency dependent core losses, bond wire ringing, switch non-overlap times, etc.

From an application point of view, it is probably not desirable to operate the converter at the maximum power output, since I^2R losses will also be maximized.

Therefore, it is recommended to constrain system power requirements to loads that can be supplied with efficiency ≥ 70% as shown in the graphs in Figures 128 and 130.

30.7.4. DCDC1 Mode011/Mode001/Mode000 Efficiency

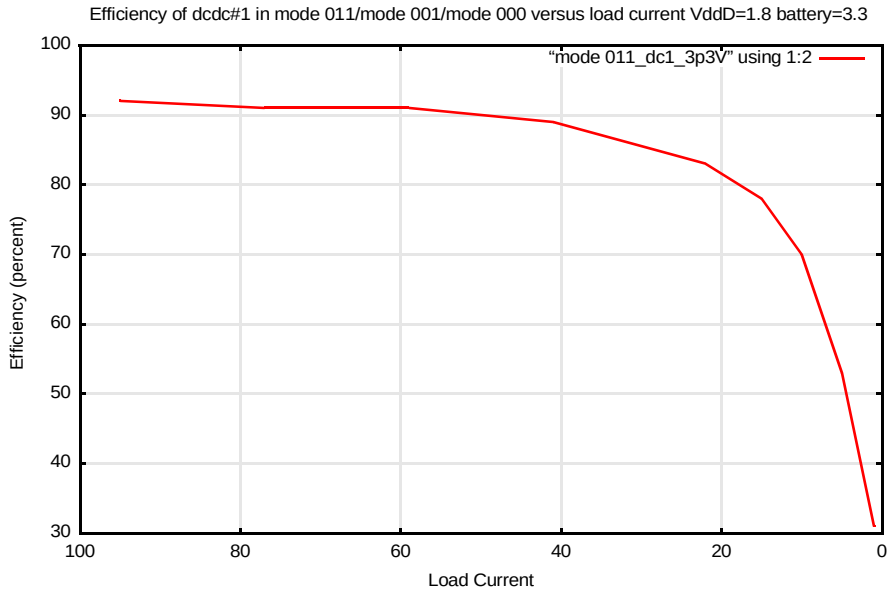


Figure 131. Efficiency of DC-DC #1 in mode 011/mode 001/mod000 vs. load current

This graph shows the efficiency for DC-DC #1 configured as a single channel buck converter. Since DC-DC #1 is configured identically in modes 011, 001, 000, this graph is applicable in all three modes.

30.7.5. Max Power Out in Buck Mode

Unlike the boost case, the maximum output power from a Buck Mode configuration is limited simply by voltage difference between the load and the battery divided by the Pfet on resistance.

$$P_{OUT} = \frac{V_{BAT} - V_{OUT}}{R_P + R_L}$$

Thus a Buck Mode configuration can better supply large currents to the load than Boost Mode configurations.

30.7.6. Clock Speed vs. Voltage

To be added after characterization.

30.8. System Brownout

The chip also contains circuitry to sense when the power requirements of the system are more than the power the DC-DC converters can provide. The entrance into the state of power required > power available is called a brownout event. Detection of a brownout event is important to provide a controlled shutdown and acceptable system behavior in the event of battery falling out, weak battery, short circuit, etc. Typically, these brownout events are low frequency events (<100 kHz) due to the large capacitors on the battery and the supply rails in the application, and the soft-

ware can shutdown the application in a controlled manner. Figure 132 shows the circuitry available to detect a brownout event:

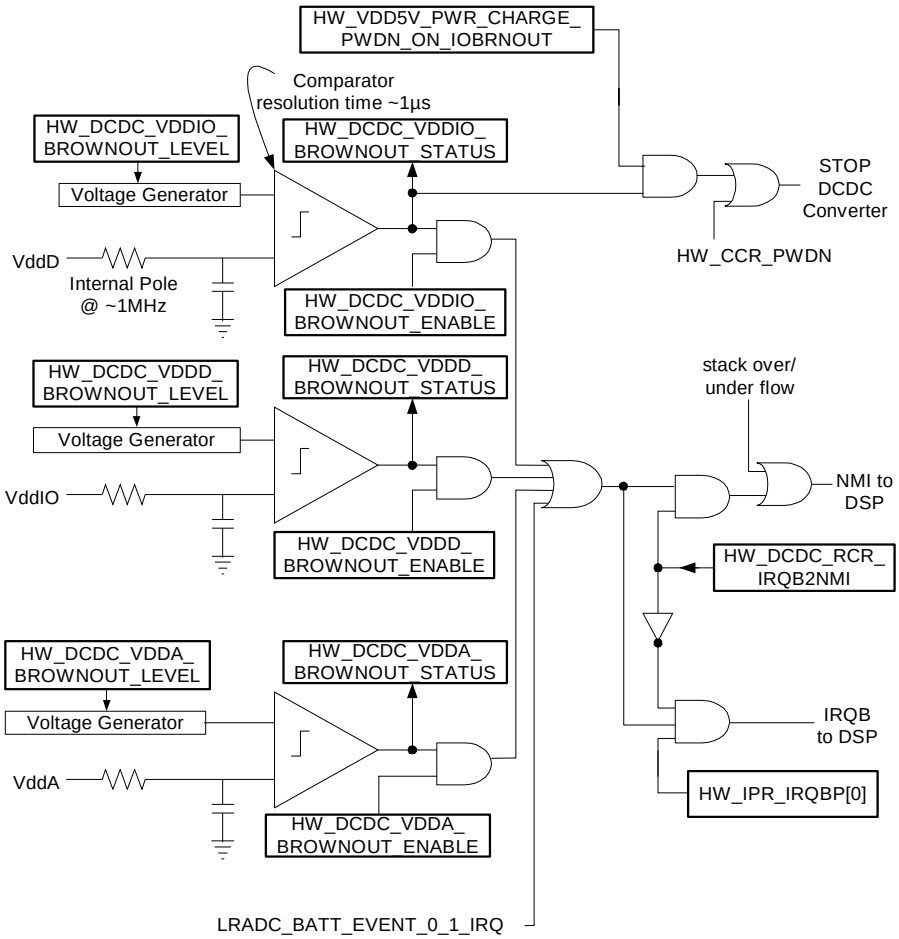
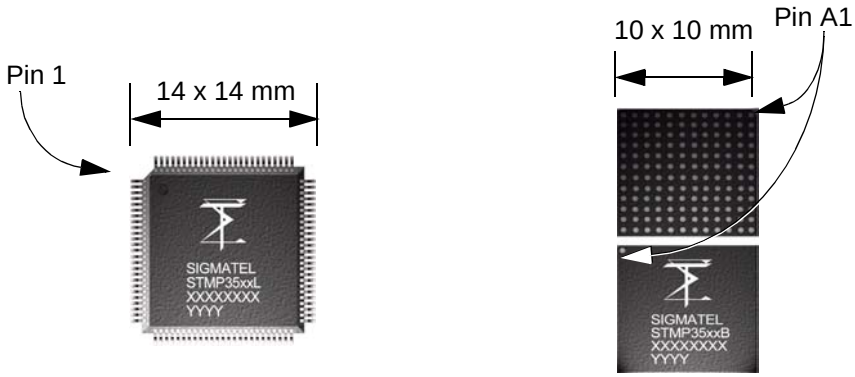


Figure 132. Brownout Event Detect Available Circuitry

As shown in Figure 132, there are three brownout sources that can be used to interrupt the DSP via a NMI or IRQB. All brownouts are detected with comparators which detect when a crucial system voltages (battery, VddD, VddIO) falls below a software defined target voltage. These brownout detection comparators have a resolution time of approximately 1us. This finite resolution time combined with the internal low-pass filter pole at 1 MHz, limit the frequency range for which a brownout event will be detected. In the audio frequency range, the brownout events will be detected when any instantaneous system voltage falls below the target voltage as defined in the software control of the Voltage Generator blocks. In the 100 kHz frequency range, the comparator resolution time will act to reduce brownout sensitivity, and the instantaneous system voltage will need to fall below the software control value to detect a brownout event. As the frequency continues to increase, this effect continues to worsen and by 500kHz, a brownout event cannot be detected. Thus, it is necessary to decouple the supplies well on the board to filter any high frequency transients that could cause an undetected brownout event.

31. PIN DESCRIPTION



100 TQFP 144 fpBGA

For additional package measurements, please see 32. “PACKAGE DRAWINGS” on page 380.

Figure 133. Chip Package Photos

31.1. Pin Placement and Definitions

100 TQFP	144 fpBGA	PIN NAME	MODULE	TYPE	PIN SET	DESCRIPTION
1	M2	GP14	GPIO	I/O	DIO3, DIO18	GP0B14
		SPI_MOSI	SPI	I/O		SPI Master Output/Slave Input
2	L2	GP13	GPIO	I/O	DIO3, DIO18	GP0B13
		SPI_MISO	SPI	I/O		SPI Master Input/Slave Output
3	K4	GP12	GPIO	I/O	D, DIO3, DIO18	GP0B12
		SPI_SCK	SPI	I/O		SPI Serial Clock
4	M3	GP16	GPIO	I/O	DIO3	GP0B16
		I2C_SCL	I^2C	I/O		I^2C Serial Clock
5	L3	GP17	GPIO	I/O	DIO3	GP0B17
		I2C_SDA	I^2C	I/O		I^2C Serial Data
6	H6	TESTMODE	SYSTEM	I	DIO3	Test Mode Pin
7	M4	CF_CE1n	EMC-CF	O	DIO3	CompactFlash Chip Enable 1
		GP44	GPIO	I/O		GP1B20
8	J5	CF_IORDn	EMC-CF	O	DIO3	CompactFlash I/O Read Data Strobe
		GP52	GPIO	I/O		GP2B4
9	L4	CF_IOWRn	EMC-CF	O	DIO3	CompactFlash I/O Write Data Strobe
		GP51	GPIO	I/O		GP2B3
10	G6	VssD2	POWER	P		Digital Core Ground 2
11	H7	VddD2	POWER	P		Digital Core Power 2
12	M5	CF_A0	EMC-CF	O	DIO3	CompactFlash Address 0
		GP32	GPIO	I/O		GP1B8
		RAM_A0	SDRAM	O		SDRAM Address 0
NA	K5	CF_A22	EMC-CF	O	DIO3	CompactFlash Address 22
		GP69	GPIO	I/O		GP2B21
		RAM_A22	SDRAM	O		SDRAM Address 22
		UTMI_RXVALID	UTMI	I		UTMI receive valid
13	L5	CF_A1	EMC-CF	O	DIO3	CompactFlash Address 1
		GP33	GPIO	I/O		GP1B9
		RAM_A1	SDRAM	O		SDRAM Address 1
NA	M6	CF_A21	EMC-CF	O	DIO3	CompactFlash Address 21
		GP68	GPIO	I/O		GP2B20
		RAM_A21	SDRAM	O		SDRAM Address 21
		UTMI_RXACTIVE	UTMI	I		UTMI
14	K6	CF_A2	EMC-CF	O	DIO3	CompactFlash Address 2
		GP34	GPIO	I/O		GP1B10 (play switch recovery if used)
		RAM_A2	SDRAM	O		SDRAM Address 2

Table 489. Pin Definition Table

100 TQFP	144 fpBGA	PIN NAME	MODULE	TYPE	PIN SET	DESCRIPTION
NA	J6	CF_A20	EMC-CF	O	DIO3	CompactFlash Address 20
		GP67	GPIO	I/O		GP2B19
		RAM_A20	SDRAM	O		SDRAM Address 20
		UTMI_RXVALIDH	UTMI	I		UTMI
15	L6	CF_A3	EMC-CF	O	DIO3	CompactFlash Address 3
		GP35	GPIO	I/O		GP1B11
		RAM_A3	SDRAM	O		SDRAM Address 3
NA	L7	CF_A19	EMC-CF	O	DIO3	CompactFlash Address 19
		GP66	GPIO	I/O		GP2B18
		RAM_A19	SDRAM	O		SDRAM Address 19
		UTMI_RXERR	UTMI	I		UTMI receive error
16	K7	CF_A4	EMC-CF	O	DIO3, DIO18	CompactFlash Address 4
		SM_CE3n	EMC-SM	O		SmartMedia/NAND Chip Enable 3
		GP36	GPIO	I/O		GP1B12
		RAM_A4	SDRAM	O		SDRAM Address 4
NA	F7	VddIO3	POWER	P		Digital I/O Power 3
NA	G7	VssIO3	POWER	P		Digital I/O Ground 3
NA	M7	CF_A18	EMC-CF	O	DIO3	CompactFlash Address 18
		GP65	GPIO	I/O		GP2B17
		RAM_A18	SDRAM	O		SDRAM Address 18
		UTMI_DATA0	UTMI	I/O		UTMI data bus
17	J7	CF_A5	EMC-CF	O	DIO3, DIO18	CompactFlash Address 5
		SM_CE2n	EMC-SM	O		SmartMedia/NAND Chip Enable 2
		GP37	GPIO	I/O		GP1B13
		RAM_A5	SDRAM	O		SDRAM Address 5
NA	L8	CF_A17	EMC-CF	O	DIO3	CompactFlash Address 17
		GP64	GPIO	I/O		GP2B16
		RAM_A17	SDRAM	O		SDRAM Address 17
		UTMI_DATA1	UTMI	I/O		UTMI data bus
18	K8	CF_A6	EMC-CF	O	DIO3, DIO18	CompactFlash Address 6
		SM_CE0n	EMC-SM	O		SmartMedia/NAND Chip Enable 0
		GP38	GPIO	I/O		GP1B14
		RAM_A6	SDRAM	O		SDRAM Address 6
NA	M8	CF_A16	EMC-CF	O	DIO3	CompactFlash Address 16
		GP63	GPIO	I/O		GP2B15
		RAM_A16	SDRAM	O		SDRAM Address 16
		UTMI_DATA2	UTMI	I/O		UTMI Data Bus
19	J8	CF_A7	EMC-CF	O	DIO3	CompactFlash Address 7
		SM_SEn	EMC-SM	O		SmartMedia/NAND Spare Area Enable
		GP39	GPIO	I/O		GP1B15
		RAM_A7	SDRAM	O		SDRAM Address 7
NA	M9	CF_A15	EMC-CF	O	DIO3	CompactFlash Address 15
		GP62	GPIO	I/O		GP2B14
		RAM_A15	SDRAM	O		SDRAM Address 15
		UTMI_DATA3	UTMI	I/O		UTMI Data Bus
20	L9	CF_A8	EMC-CF	O	DIO3, DIO18	CompactFlash Address 8
		SM_CLE	EMC-SM	O		SmartMedia/NAND Command Latch Enable
		GP40	GPIO	I/O		GP1B16
		RAM_A8	SDRAM	O		SDRAM Address 8
NA	K9	CF_A14	EMC-CF	O	DIO3	CompactFlash Address 14
		GP61	GPIO	I/O		GP2B13
		RAM_A14	SDRAM	O		SDRAM Address 14
		UTMI_DATA4	UTMI	I/O		UTMI Data Bus
21	L10	CF_A9	EMC-CF	O	DIO3, DIO18	CompactFlash Address 9
		SM_ALE	EMC-SM	O		SmartMedia/NAND Address Latch Enable
		GP41	GPIO	I/O		GP1B17
		RAM_A9	SDRAM	O		SDRAM Address 9
22	M10	CF_A10	EMC-CF	O	DIO3	CompactFlash Address 10
		GP42	GPIO	I/O		GP1B18
		RAM_A10	SDRAM	O		SDRAM Address 10
23	M11	CF_OEn	EMC-CF	O	DIO3, DIO18	CompactFlash Output Enable Strobe
		SM_REn	EMC-SM	O		SmartMedia/NAND Read Enable Strobe
		GP53	GPIO	I/O		GP2B5

Table 489. Pin Definition Table (Continued)

100 TQFP	144 fpBGA	PIN NAME	MODULE	TYPE	PIN SET	DESCRIPTION
24	K10	CF_CE0n	EMC-CF	O	DIO3, DIO18	CompactFlash Chip Enable 0
		SM_CE1n	EMC-SM	O		SmartMedia/NAND Chip Enable 1
		GP45	GPIO	I/O		GP1B21
25	M12	CF_WAITn	EMC-CF	I	DIO3, DIO18	CompactFlash Wait (not)
		SM_READY	EMC-SM	I		SmartMedia/NAND Ready/Busy
		GP56	GPIO	I/O		GP2B8
NA	L12	CF_A11	EMC-CF	O	DIO3	CompactFlash Address 11
		GP80	GPIO	I/O		GP3B8
		RAM_A11	SDRAM	O		SDRAM Address 11
		UTMI_DATA5	UTMI	I/O		UTMI Data Bus
26	L11	CF_WEn	EMC-CF	O	DIO3, DIO18	CompactFlash Write Enable Strobe
		SM_WEn	EMC-SM	O		SmartMedia/NAND Write Enable Strobe
		GP54	GPIO	I/O		GP2B6
NA	K11	CF_A12	EMC-CF	O	DIO3	CompactFlash Address 12
		GP81	GPIO	I/O		GP3B9
		RAM_A12	SDRAM	O		SDRAM Address 12
		UTMI_DATA6	UTMI	I/O		UTMI Data Bus
27	K12	SM_WPn	EMC-SM	O	DIO3, DIO18	SmartMedia/NAND Write Protect
		GP55	GPIO	I/O		GP2B7
NA	J9	CF_A13	EMC-CF	O	DIO3	CompactFlash Address 13
		GP82	GPIO	I/O		GP3B10
		RAM_A13	SDRAM	O		SDRAM Address 13
		UTMI_DATA7	UTMI	I/O		UTMI Data Bus
28	H8	VssIO1	POWER	P		Digital I/O Ground 1
29	G8	VddIO1	POWER	P		Digital I/O Power 1
30	J10	CF_D0	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 0
		SM_D0	EMC-SM	I/O		SmartMedia/NAND I/O 0
		GP24	GPIO	I/O		GP1B0
		RAM_D0	SDRAM	I/O		SDRAM Data 0
NA	J12	CF_D15	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 15
		GP79	GPIO	I/O		GP3B7
		RAM_D15	SDRAM	I/O		SDRAM Data 15
		UTMI_DATA8	UTMI	I/O		UTMI Data Bus
31	H9	CF_D1	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 1
		SM_D1	EMC-SM	I/O		SmartMedia/NAND I/O 1
		GP25	GPIO	I/O		GP1B1
		RAM_D1	SDRAM	I/O		SDRAM Data 1
NA	J11	CF_D14	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 14
		GP78	GPIO	I/O		GP3B6
		RAM_D14	SDRAM	I/O		SDRAM Data 14
		UTMI_DATA9	UTMI	I/O		UTMI Data Bus
32	H12	CF_D2	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 2
		SM_D2	EMC-SM	I/O		SmartMedia/NAND I/O 2
		GP26	GPIO	I/O		GP1B2
		RAM_D2	SDRAM	I/O		SDRAM Data 2
NA	H11	CF_D13	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 13
		GP77	GPIO	I/O		GP3B5
		RAM_D13	SDRAM	I/O		SDRAM Data 13
		UTMI_DATA10	UTMI	I/O		UTMI Data Bus
33	H10	CF_D3	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 3
		SM_D3	EMC-SM	I/O		SmartMedia/NAND I/O 3
		GP27	GPIO	I/O		GP1B3
		RAM_D3	SDRAM	I/O		SDRAM Data 3
NA	G11	CF_D12	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 12
		GP76	GPIO	I/O		GP3B4
		RAM_D12	SDRAM	I/O		SDRAM Data 12
		UTMI_DATA11	UTMI	I/O		UTMI Data Bus
34	G10	CF_D4	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 4
		SM_D4	EMC-SM	I/O		SmartMedia/NAND I/O 4
		GP28	GPIO	I/O		GP1B4
		RAM_D4	SDRAM	I/O		SDRAM Data 4

Table 489. Pin Definition Table (Continued)

100 TQFP	144 fpBGA	PIN NAME	MODULE	TYPE	PIN SET	DESCRIPTION
NA	G12	CF_D11	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 11
		GP75	GPIO	I/O		GP3B3
		RAM_D11	SDRAM	I/O		SDRAM Data 11
		UTMI_DATA12	UTMI	I/O		UTMI Data Bus
35	G9	CF_D5	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 5
		SM_D5	EMC-SM	I/O		SmartMedia/NAND I/O 5
		GP29	GPIO	I/O		GP1B5
		RAM_D5	SDRAM	I/O		SDRAM Data 5
NA	F9	CF_D10	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 10
		GP74	GPIO	I/O		GP3B2
		RAM_D10	SDRAM	I/O		SDRAM Data 10
		UTMI_DATA13	UTMI	I/O		UTMI Data Bus
36	F11	CF_D6	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 6
		SM_D6	EMC-SM	I/O		SmartMedia/NAND I/O 6
		GP30	GPIO	I/O		GP1B6
		RAM_D6	SDRAM	I/O		SDRAM Data 6
NA	F12	CF_D9	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 9
		GP73	GPIO	I/O		GP3B1
		RAM_D9	SDRAM	I/O		SDRAM Data 9
		UTMI_DATA14	UTMI	I/O		UTMI Data Bus
37	F10	CF_D7	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 7
		SM_D7	EMC-SM	I/O		SmartMedia/NAND I/O 7
		GP31	GPIO	I/O		GP1B7
		RAM_D7	SDRAM	I/O		SDRAM Data 7
38	F8	VssD1	POWER	P		Digital Core Ground 1
39	E8	VddD1	POWER	P		Digital Core Power 1
NA	E9	CF_D8	EMC-CF	I/O	DIO3, DIO18	CompactFlash Data 8
		GP72	GPIO	I/O		GP3B0
		RAM_D8	SDRAM	I/O		SDRAM Data 8
40	E12	CF_CDn	EMC-CF	I	DIO3	CompactFlash Card Detect
		GP46	GPIO	I/O		GP1B22
41	E10	CF_READY	EMC-CF	I	DIO3	CompactFlash Ready
		GP47	GPIO	I/O		GP1B23
42	E11	CF_WPn	EMC-CF	I	DIO3	CompactFlash Write Protect
		GP48	GPIO	I/O		GP2B0
43	C11	CF_RESETn	EMC-CF	I	DIO3	CompactFlash Reset
		GP50	GPIO	I/O		GP2B2
44	D11	CF_REGn	EMC-CF	O	DIO3	CompactFlash Register Select
		GP43	GPIO	I/O		GP1B19
45	D10	CF_BVD1	EMC-CF	I	DIO3	CompactFlash Bad Voltage Detect
		GP49	GPIO	I/O		GP2B1
46	B11	ONCE_DSI	SYSTEM	I	DIO3	Debug Data In
47	D12	DCDC_VddIO	DCDC	P		DCDC VddIO
48	C12	DCDC_VddD	DCDC	P		DCDC VddD
49	B12	DCDC_Batt	DCDC	P		DCDC Battery
50	A12	DCDC_Gnd	DCDC	P		DCDC Ground
NA	A11	DCDC_VddA	DCDC	P		DCDC VddA
NA	E7	VddIO4	POWER	P		Digital I/O Power 4
NA	E6	VssIO4	POWER	P		Digital I/O Ground 4
NA	B10	DCDC2_Gnd	DCDC	P		DCDC2 Ground
NA	A10	DCDC2_Batt	DCDC	P		DCDC2 Battery
NA	A9	DCDC2_Vout	DCDC	P		DCDC2 Vout
51	C10	ONCE_DSK	SYSTEM	O	DIO3	Debug Clock
52	D9	ONCE_DSO	SYSTEM	O	DIO3	Debug Data Out
53	C9	ONCE_DRN	SYSTEM	I	DIO3	Debug Reset
NA	B9	GP83	GPIO	I/O	DIO3	GP3B11
		RAM_DQM0	SDRAM	O		SDRAM DQM0
		UTMI_DATA15	UTMI	I/O		UTMI Data Bus
54	D8	DCDC_mod2	DCDC	P		DCDC mode pin 2
55	A7	MIC	CODEC	A	AIO	Microphone Input
56	D7	BATT	POWER	P		Battery Input

Table 489. Pin Definition Table (Continued)

100 TQFP	144 fpBGA	PIN NAME	MODULE	TYPE	PIN SET	DESCRIPTION
57	B8	LINE1L	CODEC	A	AIO	Line-in 1 Left
		HP_SENSE	CODEC	A		Direct coupled headphone sense
58	A8	LRADC1	SYSTEM	A	AIO	Low Resolution ADC channel 2 Input
		MIC_BIAS	CODEC	A		Microphone bias
59	C8	LINE1R	CODEC	A	AIO	Line-in 1 Right
		HP_COMMON	CODEC	A		Direct Coupled Headphone common amp.
60	C7	LRADC2	SYSTEM	A	AIO	Low Resolution ADC channel 3 input
		TEMP_SENSE	SYSTEM	A		Temperature sensor bias current
		MIC_BIAS	CODEC	A		Alternate Microphone Bias
61	B7	VDD5V	POWER	P		USB/External 5V power source
62	B6	HPL	CODEC	A	AIO	Headphone/Line-out Left
63	C6	VssHP	POWER	P		Headphone Ground
NA	D6	DCDC_mod1	DCDC	P		DCDC mode pin 1
64	D5	VddHP	POWER	P	AIO	Headphone Power
65	A6	HPR	CODEC	A	AIO	Headphone/Line-out Right
NA	D4	LINE2L	CODEC	A	AIO	Line-in 2 Left (AKA: FM-in Left)
NA	C5	LINE2R	CODEC	A	AIO	Line-in 2 Right (AKA: FM-in Right)
66	A5	Vbg	CODEC	A	AIO	Bandgap Decoupling Capacitor
67	B4	Vag	CODEC	A	AIO	Analog Ground Decoupling Capacitor
68	A4	ADCL	CODEC	A	AIO	ADC Left Filter Capacitor
69	B3	ADCR	CODEC	A	AIO	ADC Right Filter Capacitor
70	C4	REF_RES	USB	A		USB Precision Resistor
71	C3	REFp	CODEC	A	AIO	ADC Positive Reference Capacitor
NA	A3	DCDC_mod0	DCDC	P		DCDC mode pin 0
72	B5	VssA1	POWER	P		Analog Ground 1
73	B2	VddA1	POWER	P		Analog Power 1
74	A2	XTALO	SYSTEM	A		Crystal Out
75	A1	XTALI	SYSTEM	A		Crystal In
76	B1	VddPLL	POWER	P	AIO	PLL Power
77	E4	VddXTAL	SYSTEM	A		Power for XTAL oscillator (generated on-chip)
78	E5	VssPLL	POWER	P		PLL Ground
79	D3	FRESET	SYSTEM	A		Fast Reset, Fast Falling Edge (<15ns) Powers Down Device
		PSWITCH	SYSTEM	A		Power Switch
80	C2	USB_DP	USB	A	USBIO	USB Positive Data Line
81	C1	USB_DM	USB	A	USBIO	USB Negative Data Line
82	E3	GP11	GPIO	I/O	DIO3	GP0B11
83	D2	GP9	GPIO	I/O		GP0B9
		PWM2	PWM	O		Pulse Width Modulator #2
NA	E2	GP90	GPIO	I/O	DIO3	GP3B18
		RAM_CLK	SDRAM	O		SDRAM Clock
		UTMI_TXREADY	UTMI	O		UTMI transmit data ready
84	D1	GP8	GPIO	I/O	DIO3	GP0B8
85	E1	GP10	GPIO	I/O	DIO3	GP0B10
86	F6	VddD3	POWER	P		Digital Core Power 3
87	F5	VssD3	POWER	P		Digital Core Ground 3
88	F4	GP7	GPIO	I/O	DIO3	GP0B7
		I2S_DataO2	I^2S	O		I^2S Data Out 2
		PWM3	PWM	O		Pulse Width Modulator #3
89	F3	GP6	GPIO	I/O	DIO3	GP0B6
		I2S_DataO1	I^2S	O		I^2S Data Out 1
NA	F2	GP84	GPIO	I/O	DIO3	GP3B12
		RAM_DQM1	SDRAM	O		SDRAM DQM1
		UTMI_TXVALID	UTMI	I		UTMI transmit valid low byte
90	G4	GP5	GPIO	I/O	DIO3	GP0B5
		I2S_DataO0	I^2S	O		I^2S Data Out 0
NA	F1	GP89	GPIO	I/O	DIO3	GP3B17
		RAM_CKE	SDRAM	O		SDRAM CKE
		UTMI_TXVALIDH	UTMI	O		UTMI transmit valid high byte

Table 489. Pin Definition Table (Continued)

100 TQFP	144 fpBGA	PIN NAME	MODULE	TYPE	PIN SET	DESCRIPTION
91	G3	GP4	GPIO	I/O	DIO3	GP0B4
		I2S_BCLK	I^2S	I		I^2S Bit Clock
NA	G2	GP88	GPIO	I/O	DIO3	GP3B16
		RAM_CSn	SDRAM	O		SDRAM CSn
		UTMI_OPMODE0	UTMI	O		UTMI
92	H3	GP3	GPIO	I/O	DIO3	GP0B3
		I2S_LRCLK	I^2S	I		I^2S Word Clock
NA	G1	GP85	GPIO	I/O	DIO3	GP3B13
		RAM_RASn	SDRAM	O		SDRAM RASn
		UTMI_OPMODE1	UTMI	O		UTMI
93	H4	GP2	GPIO	I/O	DIO3	GP0B2
		I2S_DataI2	I^2S	I		I^2S Data In 2
NA	J3	GP86	GPIO	I/O	DIO3	GP3B14
		RAM_CASn	SDRAM	O		SDRAM CASn
		UTMI_XVER_SEL	UTMI	O		UTMI
94	H2	GP1	GPIO	I/O	DIO3	GP0B1
		I2S_DataI1	I^2S	I		I^2S Data In 1
NA	H1	GP87	GPIO	I/O	DIO3	GP3B15
		RAM_WEn	SDRAM	O		SDRAM WEn
		UTMI_LINESTATE0	UTMI	I		UTMI
95	J1	GP0	GPIO	I/O	DIO3	GP0B0
		I2S_DataI0	I^2S	I		I^2S Data In 0
NA	J2	GP92	GPIO	I/O	DIO3	GP3B20
		I2S_DataI0	I^2S	I		I^2S Data In 0 (Alternate pin)
		UTMI_LINESTATE1	UTMI	I		UTMI
96	G5	VddIO2	POWER	P		Digital I/O Power 2
97	H5	VssIO2	POWER	P		Digital I/O Ground 2
98	J4	GP19	GPIO	I/O	DIO3	GP0B19
		TIO1	TIMER	I/O		Timer 1 Pin
		PWM1	PWM	O		Pulse Width Modulator #1
NA	K3	GP93	GPIO	I/O	DIO3	GP3B21
		I2S_BCLK	I^2S	I		I^2S Bit Clock (Alternate pin)
		UTMI_CLK	UTMI	O		UTMI
99	K2	GP18	GPIO	I/O	DIO3	GP0B18
		TIO0	TIMER	I/O		Timer 0 Pin
		PWM0	PWM	O		Pulse Width Modulator #0
NA	K1	GP91	GPIO	I/O	DIO3	GP3B19
		I2S_WCLK	I^2S	I		I^2S Word Clock (Alternate pin)
		UTMI_RESET	UTMI	O		UTMI
100	L1	GP15	GPIO	I/O	DIO3	GP0B15 (Do not use as GPIO if using SPI)
		SPI_SSn	SPI	I		SPI Slave Select
NA	M1	CF_A23	EMC-CF	O	DIO3	CompactFlash Address 23
		GP70	GPIO	I/O		GP2B22
		RAM_A23	SDRAM	O		SDRAM Address 23

Table 489. Pin Definition Table (Continued)

31.1.1. Analog Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
68	A4	ADCL	A	ADC Left Filter Capacitor
69	B3	ADCR	A	ADC Right Filter Capacitor
62	B6	HPL	A	Headphone/Line-out Left
65	A6	HPR	A	Headphone/Line-out Right
57	B8	LINE1L	A	Line-in 1 Left
		HP_SENSE	A	Direct coupled headphone sense
59	C8	LINE1R	A	Line-in 1 Right
		HP_COMMON	A	Direct coupled headphone common amplifier
NA	D4	LINE2L	A	Line-in 2 Left (AKA: FM-in Left)
NA	C5	LINE2R	A	Line-in 2 Right (AKA: FM-in Right)
55	A7	MIC	A	Microphone Input
70	C4	REF_RES	A	USB Precision Resistor
71	C3	REFp	A	ADC Positive Reference Capacitor
67	B4	Vag	A	Analog Ground Decoupling Capacitor
66	A5	Vbg	A	Bandgap Decoupling Capacitor

Table 490. Analog Pins

31.1.2. DCDC Converter Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
49	B12	DCDC_Batt	P	DCDC Battery
50	A12	DCDC_Gnd	P	DCDC Ground
NA	A3	DCDC_mod0	P	DCDC mode pin 0
NA	D6	DCDC_mod1	P	DCDC mode pin 1
54	D8	DCDC_mod2	P	DCDC mode pin 2
NA	A11	DCDC_VddA	P	DCDC VddA
48	C12	DCDC_VddD	P	DCDC VddD
47	D12	DCDC_VddIO	P	DCDC VddIO
NA	A10	DCDC2_Batt	P	DCDC2 Battery
NA	B10	DCDC2_Gnd	P	DCDC2 Ground
NA	A9	DCDC2_Vout	P	DCDC2 Vout

Table 491. DCDC Converter Pins

31.1.3. External Memory Interface (CompactFlash) Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
12	M5	CF_A0	O	CompactFlash Address 0
13	L5	CF_A1	O	CompactFlash Address 1
14	K6	CF_A2	O	CompactFlash Address 2
15	L6	CF_A3	O	CompactFlash Address 3
16	K7	CF_A4	O	CompactFlash Address 4
17	J7	CF_A5	O	CompactFlash Address 5
18	K8	CF_A6	O	CompactFlash Address 6
19	J8	CF_A7	O	CompactFlash Address 7
20	L9	CF_A8	O	CompactFlash Address 8
21	L10	CF_A9	O	CompactFlash Address 9
22	M10	CF_A10	O	CompactFlash Address 10
NA	L12	CF_A11	O	CompactFlash Address 11
NA	K11	CF_A12	O	CompactFlash Address 12
NA	J9	CF_A13	O	CompactFlash Address 13
NA	K9	CF_A14	O	CompactFlash Address 14
NA	M9	CF_A15	O	CompactFlash Address 15
NA	M8	CF_A16	O	CompactFlash Address 16
NA	L8	CF_A17	O	CompactFlash Address 17
NA	M7	CF_A18	O	CompactFlash Address 18
NA	L7	CF_A19	O	CompactFlash Address 19
NA	J6	CF_A20	O	CompactFlash Address 20
NA	M6	CF_A21	O	CompactFlash Address 21
NA	K5	CF_A22	O	CompactFlash Address 22
NA	M1	CF_A23	O	CompactFlash Address 23
45	D10	CF_BVD1	I	CompactFlash Bad Voltage Detect

Table 492. External Memory Interface (CompactFlash)

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
40	E12	CF_CDn	I	CompactFlash Card Detect
24	K10	CF_CE0n	O	CompactFlash Chip Enable 0
7	M4	CF_CE1n	O	CompactFlash Chip Enable 1
30	J10	CF_D0	I/O	CompactFlash Data 0
31	H9	CF_D1	I/O	CompactFlash Data 1
32	H12	CF_D2	I/O	CompactFlash Data 2
33	H10	CF_D3	I/O	CompactFlash Data 3
34	G10	CF_D4	I/O	CompactFlash Data 4
35	G9	CF_D5	I/O	CompactFlash Data 5
36	F11	CF_D6	I/O	CompactFlash Data 6
37	F10	CF_D7	I/O	CompactFlash Data 7
NA	E9	CF_D8	I/O	CompactFlash Data 8
NA	F12	CF_D9	I/O	CompactFlash Data 9
NA	F9	CF_D10	I/O	CompactFlash Data 10
NA	G12	CF_D11	I/O	CompactFlash Data 11
NA	G11	CF_D12	I/O	CompactFlash Data 12
NA	H11	CF_D13	I/O	CompactFlash Data 13
NA	J11	CF_D14	I/O	CompactFlash Data 14
NA	J12	CF_D15	I/O	CompactFlash Data 15
8	J5	CF_IORDn	O	CompactFlash I/O Read Data Strobe
9	L4	CF_IOWRn	O	CompactFlash I/O Write Data Strobe
23	M11	CF_OEn	O	CompactFlash Output Enable Strobe
41	E10	CF_READY	I	CompactFlash Ready
44	D11	CF_REGn	O	CompactFlash Register Select
43	C11	CF_RESETn	I	CompactFlash Reset
25	M12	CF_WAITn	I	CompactFlash Wait (not)
26	L11	CF_WEn	O	CompactFlash Write Enable Strobe
42	E11	CF_WPn	I	CompactFlash Write Protect

Table 492. External Memory Interface (CompactFlash) (Continued)

31.1.4. External Memory Interface (SmartMedia/NAND) Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
21	L10	SM_ALE	O	SmartMedia/NAND Address Latch Enable
18	K8	SM_CE0n	O	SmartMedia/NAND Chip Enable 0
24	K10	SM_CE1n	O	SmartMedia/NAND Chip Enable 1
17	J7	SM_CE2n	O	SmartMedia/NAND Chip Enable 2
16	K7	SM_CE3n	O	SmartMedia/NAND Chip Enable 3
20	L9	SM_CLE	O	SmartMedia/NAND Command Latch Enable
30	J10	SM_D0	I/O	SmartMedia/NAND I/O 0
31	H9	SM_D1	I/O	SmartMedia/NAND I/O 1
32	H12	SM_D2	I/O	SmartMedia/NAND I/O 2
33	H10	SM_D3	I/O	SmartMedia/NAND I/O 3
34	G10	SM_D4	I/O	SmartMedia/NAND I/O 4
35	G9	SM_D5	I/O	SmartMedia/NAND I/O 5
36	F11	SM_D6	I/O	SmartMedia/NAND I/O 6
37	F10	SM_D7	I/O	SmartMedia/NAND I/O 7
25	M12	SM_READY	I	SmartMedia/NAND Ready/Busy
23	M11	SM_REn	O	SmartMedia/NAND Read Enable Strobe
19	J8	SM_SEn	O	SmartMedia/NAND Spare Area Enable
26	L11	SM_WEn	O	SmartMedia/NAND Write Enable Strobe
27	K12	SM_WPn	O	SmartMedia/NAND Write Protect

Table 493. External Memory Interface (SmartMedia/NAND) Pins

31.1.5. General Purpose Input/Output Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
95	J1	GP0	I/O	GP0B0
94	H2	GP1	I/O	GP0B1
93	H4	GP2	I/O	GP0B2
92	H3	GP3	I/O	GP0B3
91	G3	GP4	I/O	GP0B4
90	G4	GP5	I/O	GP0B5

Table 494. General Purpose Input/Output Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
89	F3	GP6	I/O	GP0B6
88	F4	GP7	I/O	GP0B7
84	D1	GP8	I/O	GP0B8
83	D2	GP9	I/O	GP0B9
85	E1	GP10	I/O	GP0B10
82	E3	GP11	I/O	GP0B11
3	K4	GP12	I/O	GP0B12
2	L2	GP13	I/O	GP0B13
1	M2	GP14	I/O	GP0B14
100	L1	GP15	I/O	GP0B15 (Do not use as GPIO if using SPI)
4	M3	GP16	I/O	GP0B16
5	L3	GP17	I/O	GP0B17
99	K2	GP18	I/O	GP0B18
98	J4	GP19	I/O	GP0B19
30	J10	GP24	I/O	GP1B0
31	H9	GP25	I/O	GP1B1
32	H12	GP26	I/O	GP1B2
33	H10	GP27	I/O	GP1B3
34	G10	GP28	I/O	GP1B4
35	G9	GP29	I/O	GP1B5
36	F11	GP30	I/O	GP1B6
37	F10	GP31	I/O	GP1B7
12	M5	GP32	I/O	GP1B8
13	L5	GP33	I/O	GP1B9
14	K6	GP34	I/O	GP1B10
15	L6	GP35	I/O	GP1B11
16	K7	GP36	I/O	GP1B12
17	J7	GP37	I/O	GP1B13
18	K8	GP38	I/O	GP1B14
19	J8	GP39	I/O	GP1B15
20	L9	GP40	I/O	GP1B16
21	L10	GP41	I/O	GP1B17
22	M10	GP42	I/O	GP1B18
44	D11	GP43	I/O	GP1B19
7	M4	GP44	I/O	GP1B20
24	K10	GP45	I/O	GP1B21
40	E12	GP46	I/O	GP1B22
41	E10	GP47	I/O	GP1B23
42	E11	GP48	I/O	GP2B0
45	D10	GP49	I/O	GP2B1
43	C11	GP50	I/O	GP2B2
9	L4	GP51	I/O	GP2B3
8	J5	GP52	I/O	GP2B4
23	M11	GP53	I/O	GP2B5
26	L11	GP54	I/O	GP2B6
27	K12	GP55	I/O	GP2B7
25	M12	GP56	I/O	GP2B8
NA	K9	GP61	I/O	GP2B13
NA	M9	GP62	I/O	GP2B14
NA	M8	GP63	I/O	GP2B15
NA	L8	GP64	I/O	GP2B16
NA	M7	GP65	I/O	GP2B17
NA	L7	GP66	I/O	GP2B18
NA	J6	GP67	I/O	GP2B19
NA	M6	GP68	I/O	GP2B20
NA	K5	GP69	I/O	GP2B21
NA	M1	GP70	I/O	GP2B22
NA	E9	GP72	I/O	GP3B0
NA	F12	GP73	I/O	GP3B1
NA	F9	GP74	I/O	GP3B2
NA	G12	GP75	I/O	GP3B3
NA	G11	GP76	I/O	GP3B4
NA	H11	GP77	I/O	GP3B5
NA	J11	GP78	I/O	GP3B6
NA	J12	GP79	I/O	GP3B7

Table 494. General Purpose Input/Output Pins (Continued)

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
NA	L12	GP80	I/O	GP3B8
NA	K11	GP81	I/O	GP3B9
NA	J9	GP82	I/O	GP3B10
NA	B9	GP83	I/O	GP3B11
NA	F2	GP84	I/O	GP3B12
NA	G1	GP85	I/O	GP3B13
NA	J3	GP86	I/O	GP3B14
NA	H1	GP87	I/O	GP3B15
NA	G2	GP88	I/O	GP3B16
NA	F1	GP89	I/O	GP3B17
NA	E2	GP90	I/O	GP3B18
NA	K1	GP91	I/O	GP3B19
NA	J2	GP92	I/O	GP3B20
NA	K3	GP93	I/O	GP3B21

Table 494. General Purpose Input/Output Pins (Continued)

31.1.6. I^2C Interface Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
4	M3	I2C_SCL	I/O	I^2C Serial Clock
5	L3	I2C_SDA	I/O	I^2C Serial Data

Table 495. I^2C Pins

31.1.7. I^2S Interface Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
91	G3	I2S_BCLK	I	I^2S Bit Clock
NA	K3	I2S_BCLK	I	I^2S Bit Clock (Alternate pin)
95	J1	I2S_DataI0	I	I^2S Data In 0
NA	J2	I2S_DataI0	I	I^2S Data In 0 (Alternate pin)
94	H2	I2S_DataI1	I	I^2S Data In 1
93	H4	I2S_DataI2	I	I^2S Data In 2
90	G4	I2S_DataO0	O	I^2S Data Out 0
89	F3	I2S_DataO1	O	I^2S Data Out 1
88	F4	I2S_DataO2	O	I^2S Data Out 2
92	H3	I2S_WCLK	I	I^2S Word Clock
NA	K1	I2S_WCLK	I	I^2S Word Clock (Alternate pin)

Table 496. I^2S Pins

31.1.8. Power Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
56	D7	BATT	P	Battery Input
73	B2	VddA1	P	Analog Power 1
61	B7	VddA2	P	Analog Power 2
39	E8	VddD1	P	Digital Core Power 1
11	H7	VddD2	P	Digital Core Power 2
86	F6	VddD3	P	Digital Core Power 3
64	D5	VddHP	P	Headphone Power
29	G8	VddIO1	P	Digital I/O Power 1
96	G5	VddIO2	P	Digital I/O Power 2
NA	F7	VddIO3	P	Digital I/O Power 3
NA	E7	VddIO4	P	Digital I/O Power 4
76	B1	VddPLL	P	PLL Power
72	B5	VssA1	P	Analog Ground 1
38	F8	VssD1	P	Digital Core Ground 1
10	G6	VssD2	P	Digital Core Ground 2
87	F5	VssD3	P	Digital Core Ground 3
63	C6	VssHP	P	Headphone Ground
28	H8	VssIO1	P	Digital I/O Ground 1
97	H5	VssIO2	P	Digital I/O Ground 2
NA	G7	VssIO3	P	Digital I/O Ground 3
NA	E6	VssIO4	P	Digital I/O Ground 4

Table 497. Power Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
78	E5	VssPLL	P	PLL Ground

Table 497. Power Pins (Continued)

31.1.9. SDRAM Interface Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
12	M5	RAM_A0	O	SDRAM Address 0
13	L5	RAM_A1	O	SDRAM Address 1
14	K6	RAM_A2	O	SDRAM Address 2
15	L6	RAM_A3	O	SDRAM Address 3
16	K7	RAM_A4	O	SDRAM Address 4
17	J7	RAM_A5	O	SDRAM Address 5
18	K8	RAM_A6	O	SDRAM Address 6
19	J8	RAM_A7	O	SDRAM Address 7
20	L9	RAM_A8	O	SDRAM Address 8
21	L10	RAM_A9	O	SDRAM Address 9
22	M10	RAM_A10	O	SDRAM Address 10
NA	L12	RAM_A11	O	SDRAM Address 11
NA	K11	RAM_A12	O	SDRAM Address 12
NA	J9	RAM_A13	O	SDRAM Address 13
NA	K9	RAM_A14	O	SDRAM Address 14
NA	M9	RAM_A15	O	SDRAM Address 15
NA	M8	RAM_A16	O	SDRAM Address 16
NA	L8	RAM_A17	O	SDRAM Address 17
NA	M7	RAM_A18	O	SDRAM Address 18
NA	L7	RAM_A19	O	SDRAM Address 19
NA	J6	RAM_A20	O	SDRAM Address 20
NA	M6	RAM_A21	O	SDRAM Address 21
NA	K5	RAM_A22	O	SDRAM Address 22
NA	M1	RAM_A23	O	SDRAM Address 23
NA	J3	RAM_CASn	O	SDRAM CASn
NA	F1	RAM_CKE	O	SDRAM CKE
NA	E2	RAM_CLK	O	SDRAM Clock
NA	G2	RAM_CSn	O	SDRAM CSn
30	J10	RAM_D0	I/O	SDRAM Data 0
31	H9	RAM_D1	I/O	SDRAM Data 1
32	H12	RAM_D2	I/O	SDRAM Data 2
33	H10	RAM_D3	I/O	SDRAM Data 3
34	G10	RAM_D4	I/O	SDRAM Data 4
35	G9	RAM_D5	I/O	SDRAM Data 5
36	F11	RAM_D6	I/O	SDRAM Data 6
37	F10	RAM_D7	I/O	SDRAM Data 7
NA	E9	RAM_D8	I/O	SDRAM Data 8
NA	F12	RAM_D9	I/O	SDRAM Data 9
NA	F9	RAM_D10	I/O	SDRAM Data 10
NA	G12	RAM_D11	I/O	SDRAM Data 11
NA	G11	RAM_D12	I/O	SDRAM Data 12
NA	H11	RAM_D13	I/O	SDRAM Data 13
NA	J11	RAM_D14	I/O	SDRAM Data 14
NA	J12	RAM_D15	I/O	SDRAM Data 15
NA	B9	RAM_DQM0	O	SDRAM DQM0
NA	F2	RAM_DQM1	O	SDRAM DQM1
NA	G1	RAM_RASn	O	SDRAM RASn
NA	H1	RAM_WEn	O	SDRAM WEn

Table 498. SDRAM Pins

31.1.10. SPI Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
2	L2	SPI_MISO	I/O	SPI Master Input/Slave Output
1	M2	SPI_MOSI	I/O	SPI Master Output/Slave Input
3	K4	SPI_SCK	I/O	SPI Serial Clock
100	L1	SPI_SSn	I	SPI Slave Select

Table 499. SPI Pins

31.1.11. System Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
79	D3	FRESET	A	Fast Reset, Fast Falling Edge (<15ns) Powers Down Device
		PSWITCH	A	Power Switch
58	A8	LRADC1	A	Low Resolution ADC Input
		MIC_BIAS	A	microphone bias
60	C7	LRADC2	A	Low Resolution ADC Input & Temp sensor bias
		MIC_BIAS	A	alternate mic bias
53	C9	ONCE_DRn	I	Debug Reset
46	B11	ONCE_DSI	I	Debug Data In
51	C10	ONCE_DSK	O	Debug Clock
52	D9	ONCE_DSO	O	Debug Data Out
6	H6	TESTMODE	I	Test Mode Pin
77	E4	VddXTAL	A	Power for XTAL oscillator (generated on-chip)
75	A1	XTALI	A	Crystal In
74	A2	XTALO	A	Crystal Out

Table 500. System Pins

31.1.12. Timer Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
99	K2	TIO0	I/O	Timer 0 Pin
98	J4	TIO1	I/O	Timer 1 Pin

Table 501. Timer Pins

31.1.13. Pulse Width Modulator Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
99	K2	PWM0	O	Pulse Width Modulator #0
98	J4	PWM1	O	Pulse Width Modulator #1
83	D2	PWM2	O	Pulse Width Modulator #2
88	F4	PWM3	O	Pulse Width Modulator #3

Table 502. PWM Pins

31.1.14. USB Pins

100TQFP	144fpBGA	PIN	TYPE	DESCRIPTION
81	C1	USB_DM	A	USB Negative Data Line
80	C2	USB_DP	A	USB Positive Data Line

Table 503. USB Pins

TYPE	DESCRIPTION
A	Analog pin
I	Input pin
I/O	Input/output pin
O	Output pin
P	Power pin

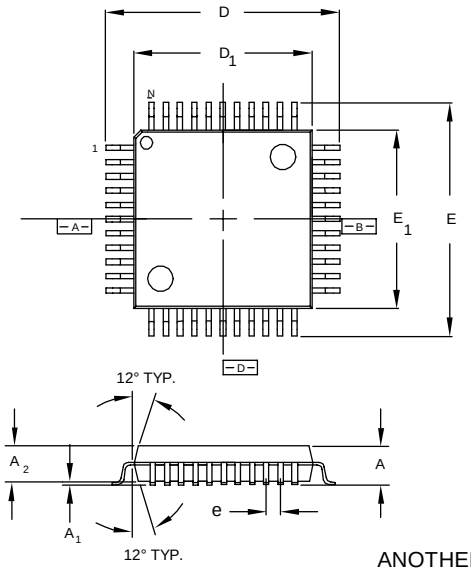
MODULE	DESCRIPTION
CODEC	Analog pins
DCDC	DCDC Converter pins
EMC-CF	External Memory Interface pins (CompactFlash)
EMC-SM	External Memory Interface pins (SmartMedia/NAND)
GPIO	General Purpose Input/Output pins
I^2C	I^2C pins
I^2S	I^2S pins
POWER	Power pins
SDRAM	SDRAM pins
SPI	SPI pins
SYSTEM	System pins
TIMER	Timer pins
USB	USB pins
PWM	Pulse Width Modulator

Almost all digital pins are powered down (i.e. high impedance) at reset, until reprogrammed by the DSP. The only exceptions are: TESTMODE, ONCE_DSI, ONCE_DSK, ONCE_DSO, ONCE_DRN; these pins are always active.

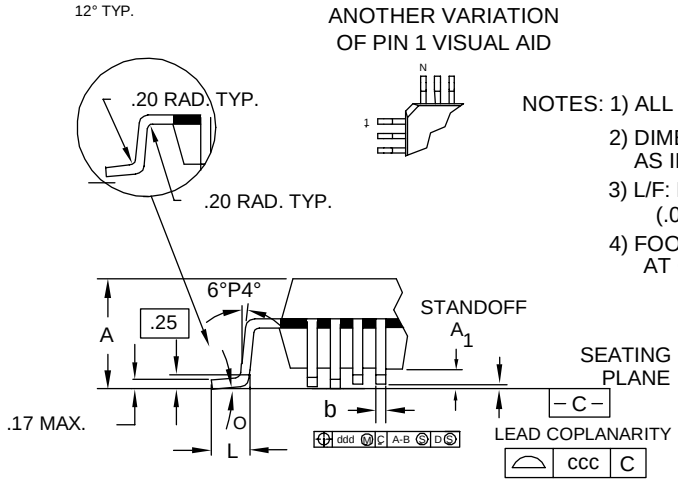
Table 504. Notes on Pin Placement and Definitions

32. PACKAGE DRAWINGS

32.1. 100-Pin TQFP



BODY + 2.00 mm FOOTPRINT		
	LEADS	100L-1.4 THK.
DIMS.	TOL. \ L/F	.127 .152
A	MAX.	1.60
A_1		.05 MIN./.15 MAX.
A_2	±.05	1.40
D	±.20	16.00
D_1	±.05	14.00
E	±.20	16.00
E_1	±.05	14.00
L	+.15/-.10	.60
e	BASIC	.50
b	±.05	.22
Θ		0°-7D
ddd	MAX.	.08
ccc	MAX.	.08



NOTES: 1) ALL DIMENSIONS IN MM.

2) DIMENSIONS SHOWN ARE NOMINAL WITH TOL. AS INDICATED

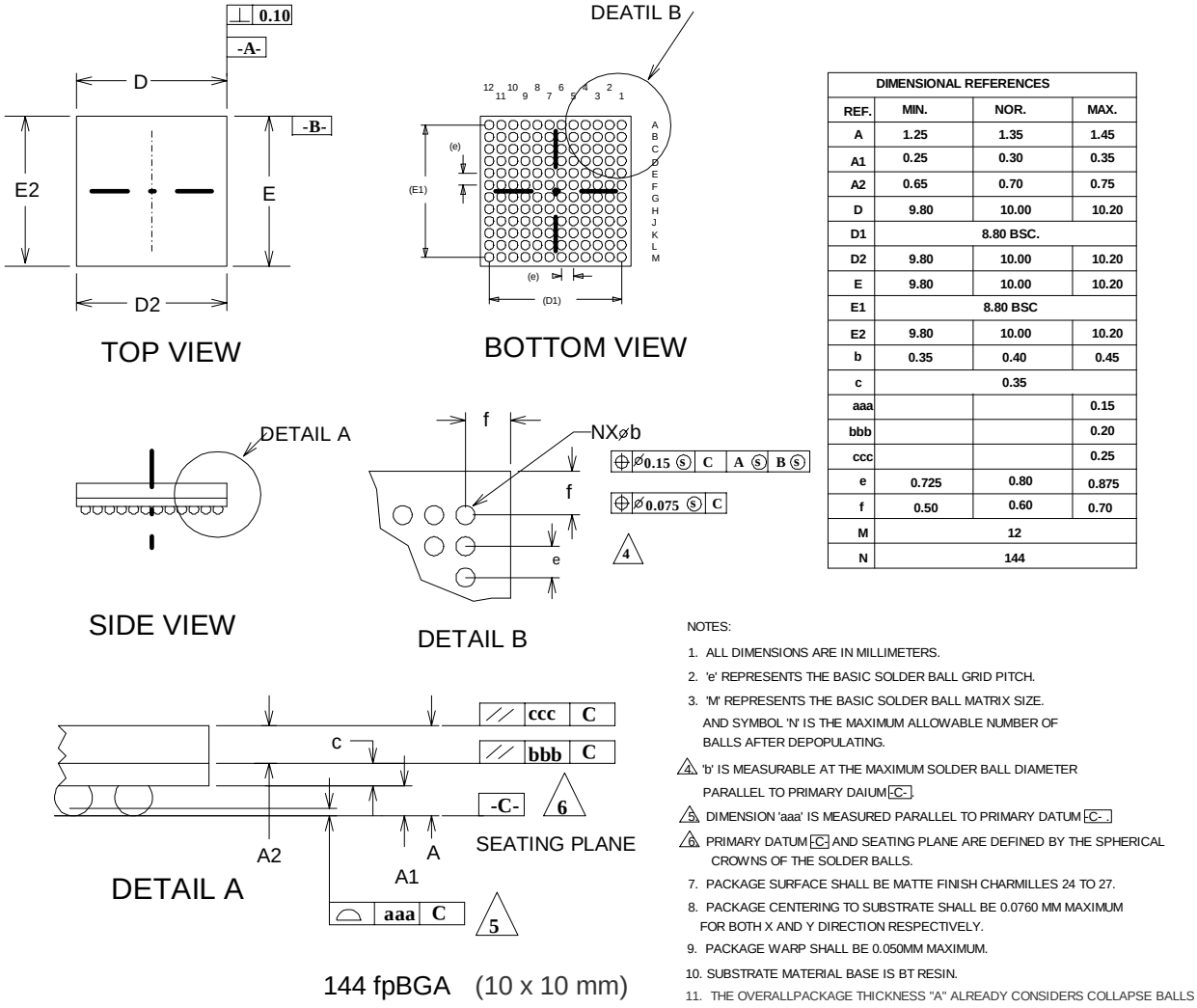
3) L/F: EFTEC 64T COPPER OR EQUIVALENT, 0.127 MM (.005") OR 0.15 MM (.006") THICK.

4) FOOT LENGTH "L" IS MEASURED AT GAGE PLANE, AT 0.25 ABOVE THE SEATING PLANE.

100 Pin 14x14 TQFP

Figure 134. 100-Pin TQFP Package Drawing

32.2. 144-Pin fpBGA



DIMENSIONAL REFERENCES			
REF.	MIN.	NOR.	MAX.
A	1.25	1.35	1.45
A1	0.25	0.30	0.35
A2	0.65	0.70	0.75
D	9.80	10.00	10.20
D1		8.80 BSC.	
D2	9.80	10.00	10.20
E	9.80	10.00	10.20
E1		8.80 BSC	
E2	9.80	10.00	10.20
b	0.35	0.40	0.45
c		0.35	
aaa			0.15
bbb			0.20
ccc			0.25
e	0.725	0.80	0.875
f	0.50	0.60	0.70
M		12	
N		144	

NOTES:

1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. 'e' REPRESENTS THE BASIC SOLDER BALL GRID PITCH.
3. 'M' REPRESENTS THE BASIC SOLDER BALL MATRIX SIZE. AND SYMBOL 'N' IS THE MAXIMUM ALLOWABLE NUMBER OF BALLS AFTER DEPOPULATING.
4. 'b' IS MEASURABLE AT THE MAXIMUM SOLDER BALL DIAMETER PARALLEL TO PRIMARY DAIUM -C-.
5. DIMENSION 'aaa' IS MEASURED PARALLEL TO PRIMARY DATUM -C-.
6. PRIMARY DATUM -C- AND SEATING PLANE ARE DEFINED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
7. PACKAGE SURFACE SHALL BE MATTE FINISH CHARMILLES 24 TO 27.
8. PACKAGE CENTERING TO SUBSTRATE SHALL BE 0.0760 MM MAXIMUM FOR BOTH X AND Y DIRECTION RESPECTIVELY.
9. PACKAGE WARP SHALL BE 0.050MM MAXIMUM.
10. SUBSTRATE MATERIAL BASE IS BT RESIN.
11. THE OVERALLPACKAGE THICKNESS "A" ALREADY CONSIDERS COLLAPSE BALLS

Figure 135. 144-Pin fpBGA Package Drawing

33. INDEX OF REGISTERS