personal imaging products

MFC2000

Multifunctional Peripheral Controller 2000
Hardware Description

CONEXANT

Ordering Information

Marketing Name	Device Set Order No.				
		Part No.	Package	Part No.	Package
MFC2000	xxx-xxx-xxx	xxxxx			

Revision History

Revision	Date	Comments
A	04/07/00	Initial, internal, preliminary release of document.
A	06/21/00	Second internal, preliminary release with revisions tracked.

Contents

Figures

Tables

This page is intentionally blank

1. Introduction

1.1   Scope

This document defines and describes all hardware functions of the MFC2000 chip. The MFC2000 design is based on the MFC1000 design with many minor modifications/enhancements. It has several new key functions to accomplish the following:

- Support a full color MFP
- Enhance connectivity to the PC
- Provide an internal Fax Modem
- Connect to external Conexant video chips

1.2   System Overview

The Conexant Multi-Functional Peripheral Controller 2000 (MFC2000) device set hardware, core code, application code, and evaluation system comprise a full color Multi-Functional Peripheral (MFP) system–needing only a power supply, scanner, printer mechanism, and paper path components to complete the machine. The standard device set hardware includes Conexant's MFC2000 chip, Conexant's SmartDAA or IA chip, and a Printer Interface chip. Optionally, a Conexant video chip with VIP interface can be used to support the video capture function. If V.17 or V.34 faxing without voice is required, the internal V.17/V.34 Fax Modem in the MFC2000 chip is used and the MFC2000 is connected to the external Conexant SmartDAA or IA chip. If the voice/speech function is required, the external Voice Fax Modem device from Conexant will be needed. Any other external interface device can be supported by using the external ARM for CPU and DMA accesses or by using the internal serial interface. An MFP system-level block diagram using the MFC 2000 is illustrated in Figure 1-1.
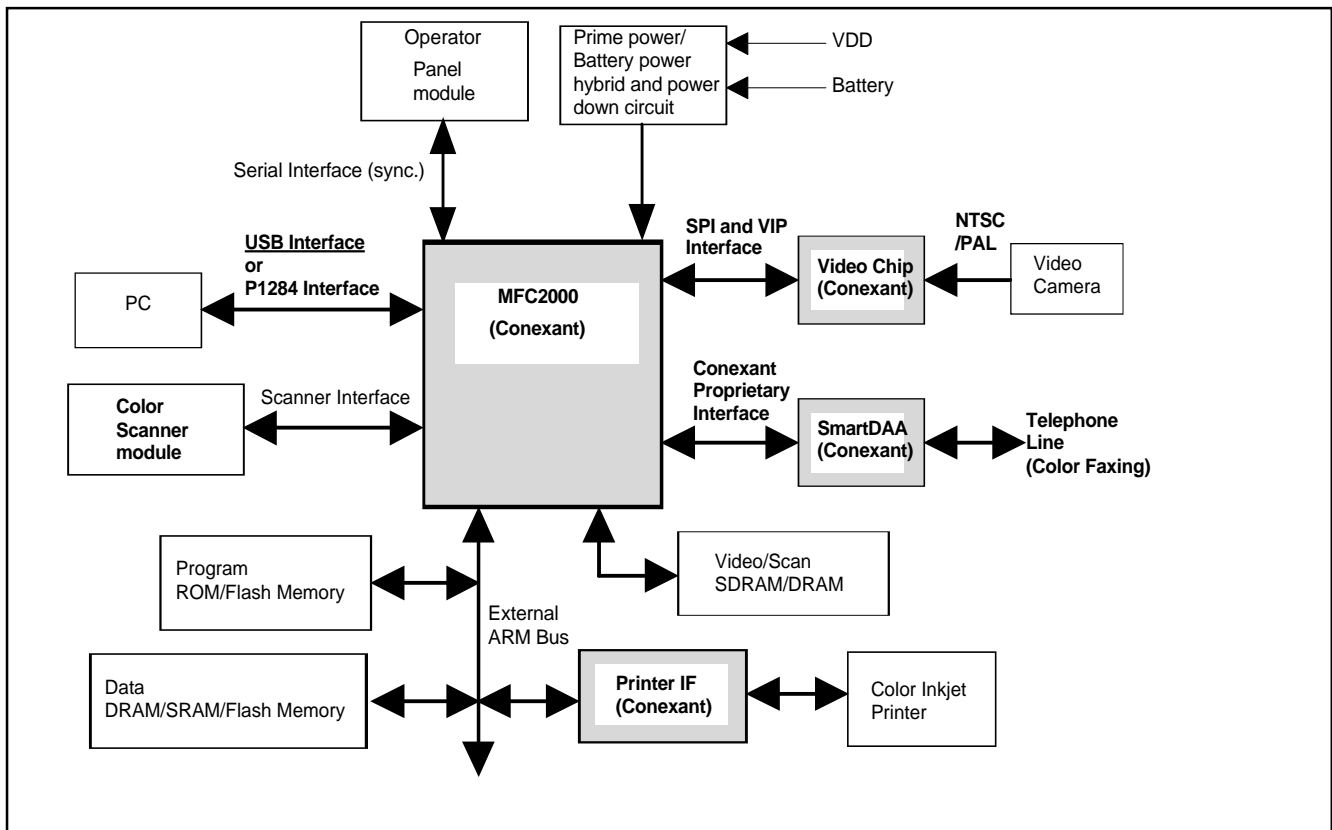
Figure 1-1. MFP System Diagram Using MFC2000

1.2.1   Integrated Full Color MFP Controller (MFC2000)

The MFC2000 provides the majority of the electronics necessary to build a color scan and color inkjet printer based MFP whose electronics are integrated into a one-chip solution including one CPU (ARM7TDMI) and two DSPs (Countach Imaging DSP subsystem and P80 core).

Full printer and copier functionality is provided by the following:

- 1284 parallel port interface
- USB serial port interface
- Color scanner interface/controller
- Countach Imaging DSP subsystem for video/scan/compression process
- Flash memory controller
- SDRAM/DRAM controllers
- Resolution conversion logic
- Inkjet data formatter
- External inkjet printing

In addition, the MFC2000 performs facsimile control/monitoring, compression/decompression, and 33.6 Kbps Fax Modem functions (P80 core). The MFC2000 interfaces with major MFP machine components like external modems, SmartDAA, external Fax IA, motors, sensors, external video chip, and operator control panel. The ARM7TDMI-embedded processor provides an external 48-MB direct memory access capability. An integrated 12-bit Pipeline ADC (PADC) and countach subsystem (DSP subsystem, combined with an advanced Conexant proprietary color image processing algorithm, provides state of the art image processing performance on any type of images, including text/half-tone and color images.

The full color MFP Engine provides the hardware and software necessary to develop a full-color Multifunctional Peripheral including an architecture for color printing, color faxing, color scanning, video capturing, and color copying. It also supports many of these operations concurrently.

1.2.1.1   Printing

The MFC2000 Controller supports color inkjet printing. Print speed throughput capabilities are inversely proportional to resolution and also depend on the external printer interface. For host printing, the host sends the image data with the print resolution; the MFC2000 performs no resolution conversion. If host printing and faxing need to be performed for the same image, the printing image data must be sent to the MFP. The MFC2000 converts the printing image data to the faxing image data locally and then faxes it out. An external printer interface chip is designed to support inkjet print mechanism/head subsystems. Different external printer interface chips can be designed and used to support other inkjet mechanisms and heads according to customer requirements.

1.2.1.2   Faxing

Both host-based color faxing and standalone color faxing are supported in addition to monochrome faxing. Host-based faxing can take place by using a Class One connection via the USB serial port or the P1284 parallel port. For host faxing, the host sends the image data with the fax resolution; the MFC2000 performs no resolution conversion. For standalone faxing, the resolution conversion is supported by the MFC2000. The standalone color scan-to-fax function is supported using the advanced Conexant proprietary color image processing technology:

- Shading correction
- Gamma correction
- Pixel-based dark-level correction
- Color/monochrome image processing
- Color conversion
- JPEG
- Multi-level resolution conversion.

1.2.1.3   Scanning

For the color scan-to-PC function, up to 8 bits of scan data per pixel can be sent to the host. JPEG compression can be used to reduce the PC upload speed.

1.2.1.4   Copying

The MFC2000 and associated firmware supports several modes of copying including standard, fine, superfine, and photo. Multiple copies of a single image can be made with a single pass. The standalone color/monochrome copy function is supported by using MFC2000's Inkjet print formatter, the external printer interface, and the advanced Conexant proprietary color image processing system.

1.2.2   MFC2000 Evaluation System

The Conexant MFC2000 Evaluation System provides demonstration, prototype development, and evaluation capabilities to full color MFP developers using the MFC2000 Engine device set. The MFC2000 Evaluation system provides flexibility for visibility and access, i.e., plug-on board for the modem, sockets for programmable parts, and connectors for an emulator (refer to Figure 1-2). Jumper options and test points are provided throughout the MFC2000 evaluation Main Board. The MFC2000 Evaluation System is the most convenient environment for the developer needing to experiment with the several interfaces encountered in the full color MFP, for example, color printer functions.

1.2.3   New Function Highlights

- PLL Clock Generation for several different clocks needed for ARM CPU, Countach Imaging DSP, Fax Modem core, and USB Interface
- 4 KB 2-way Set Associative Instruction Cache
- USB Interface (including USB Transceiver) to PC
- Video/ Color Scan Controller (up to 600 dpi) (including programmable ADC sampling rate)
- Countach Imaging DSP Subsystem for pixel-based dark level correction, shading correction, gamma correction, video/color scan image processing, color science and JPEG
- Countach Bus System which includes Countach Subsystem Interface, ARM Bus Interface, Video/Scan Interface, Countach Bus Unit, Countach DMA Controller, and SDRAM Controller.
- SmartDAA/IA Interface
- P80 Core + ARM IPB interface logic (V.34 Fax Modem core)
- Two Sync. Serial Interfaces
- Color Scan IA which includes Color Scan analog Front End, 12-bit PADC, Power-down Voltage Detection Circuit, and TADC reference voltage.
- SPI and VIP interface to the external Conexant video chip
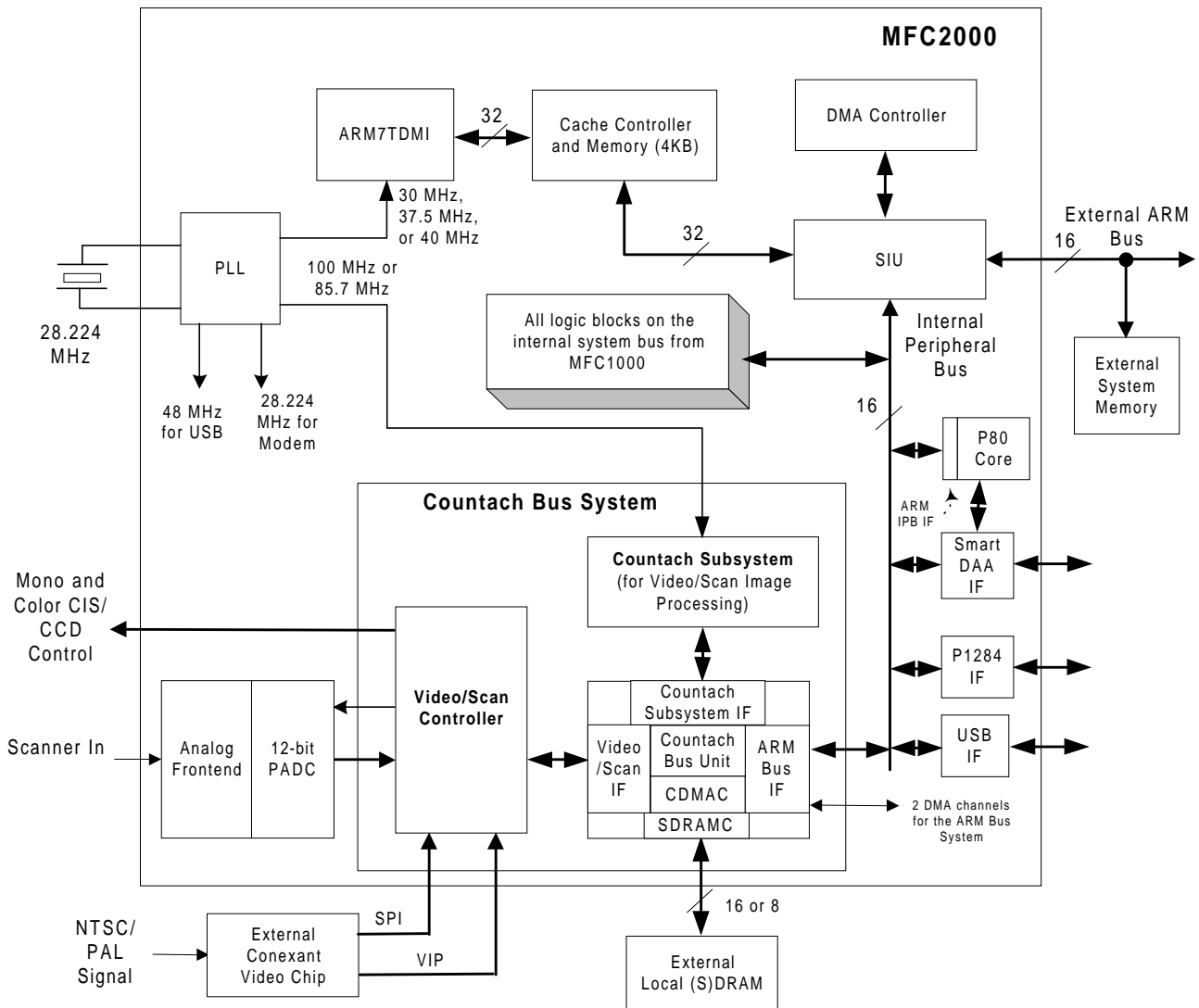
Figure 1-2: MFC2000 Function Diagram

1.3 Reference Documentation

Table 1-1. Reference Documentation

Document	Number
MFC2000 Data Sheet	100505
MFC2000 Firmware Architecture	100972
MFC2000 Hardware Description (this document)	100723
MFC2000 Programmer's Reference Manual	100971

This page is intentionally blank

2. MFC2000 Summary

2.1   MFC2000 Device Family

The MFC2000 contains an internal 32-bit RISC Processor with 64-MB address space, the Countach Imaging DSP (Conexant's DSP) subsystem including embedded data and program memory, and dedicated circuitry optimized for color scanning, color faxing, color copying, color printing, and multifunctional control and monitoring. The device family with relevant features is described in Table 2-1.

Table 2-1. MFC2000 Device Family

Device No.	Product Code	Data Modem Function	Voice Codec/Speaker Phone Functions	Smart DAA Support
CX0720X-11	BFH	Yes	Yes	Yes
CX0720X-12	BDH	Yes	No	Yes
CX0720X -13	BBH	No	Yes	Yes
CX0720X -14	B9H	No	No	Yes
CX0720X-15	B8H	No	No	No

2.2   MFC2000 System Block Diagram

The MFC2000 contains the ARM7TDMI RISC Processor (described separately in ARM7TDMI Manuals), Countach Imaging DSP, Modem DSP, and specialized hardware needed for the Multifunctional machine control and scanner and fax signal processing. The Countach Imaging DSP subsystem is on a separate data bus. Therefore, the ARM system data bus can operate in parallel with the Countach Imaging DSP subsystem data bus for most operations except the interaction time between two buses. The two-bus architecture is very important to provide enough bandwidth for full color MFP products. Figure 2-1 shows the MFP2000's two-bus architecture. The ARM Bus System (ABS) has two masters—ARM CPU and DMA Controller. They provide accesses to all specialized hardware functions including the Countach Imaging DSP subsystem as a peripheral on the ARM Bus System. ABS has several sections. The System Interface Unit (SIU) is the control center. The ARM CPU and Cache Controller are on the Internal System Bus (ISB). The Cache Memory is on the Internal Cache Bus (ICB). The DMA Controller is on the DMA Bus (DAB). All internal peripherals are on the Internal Peripheral Bus (IPB). The SmartDAA/IA Interface and P80 core are on the IPB of the ARM Bus System. The ARM7TDMI runs at a clock rate 40 MHz, 37.5 MHz, or 30 MHz. All external peripherals are on the ARM External Bus (AEB). There is a separate bus system for the Countach Imaging DSP subsystem called Countach Bus System (CBS). There are three sections, the Video/Scan Interface, the ARM Bus Interface, and the countach subsystem interface. The external SDRAM/DRAM is on the Countach External Bus (CEB).
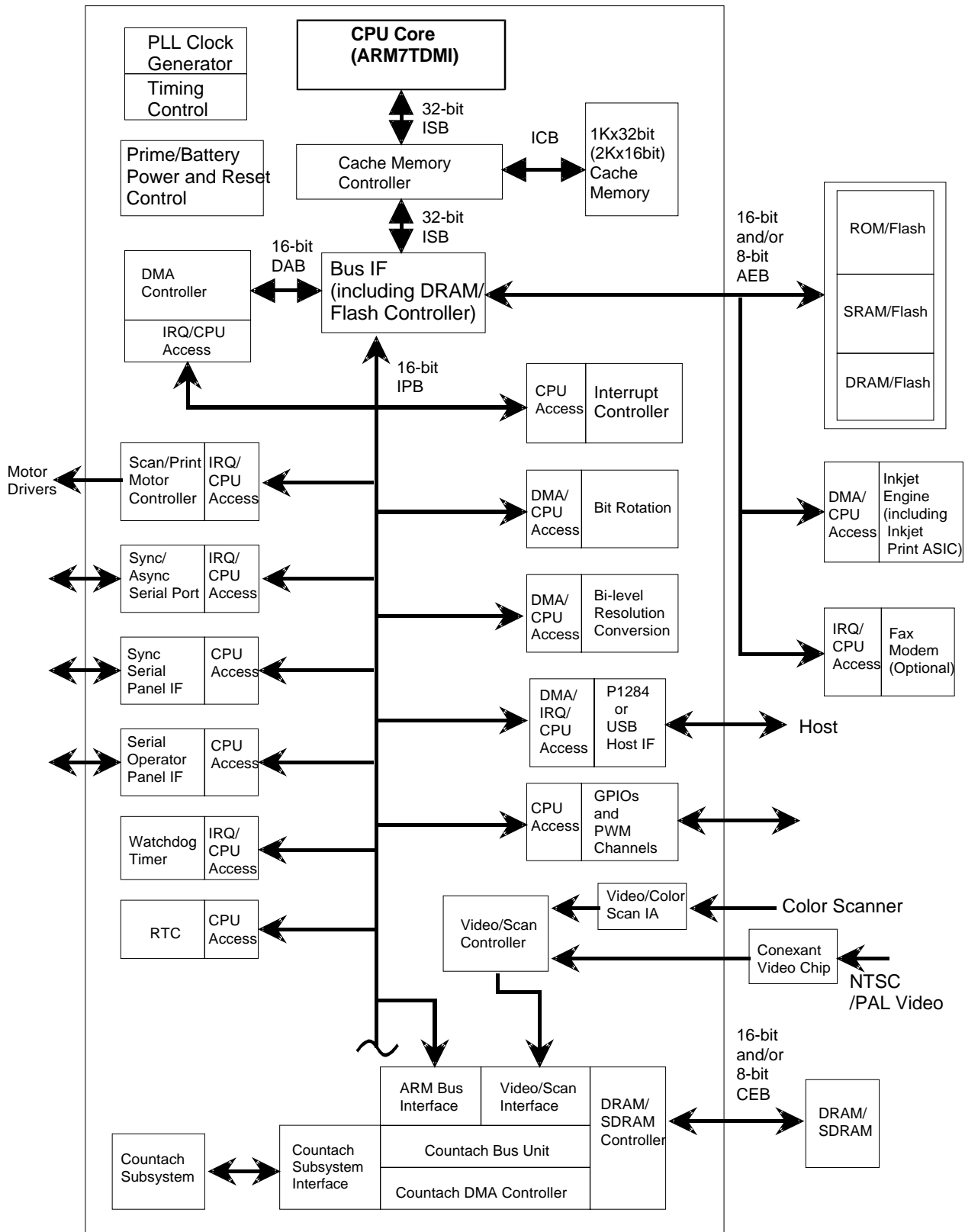
Figure 2-1. MFC2000 Organization

3.   Hardware Interface

3.1   Pin Description

Table 3-1. Pin Description (1 of 6)

Pin Name	Pin No.	I/O	Input Type	Output Type	Pin Description
PRTIRQn	U14	I	HU5VT	-	(Hysteresis, Pull up) Interrupt from the external printing ASIC (active low)
AUXCLK	K20	O	-	2XT3V	Auxiliary clock output for using as the master clock for external devices
A[11:0]/A[23:12]	A20,B20,B19,B18,B17,C20,C19,C18,C17,D20,D19,D18	I/O	5VT	3XT5VT	Address bus (12-bit), A[23:12] and A[11:0] are muxed out through same pins.
ALE	C16	O	-	2XT5VT	Address Latch output signal for latching A[23:12] externally
AE[2]/GPO[14]/SSTXD2/ ROM_CONFIG[0]	A19	I/O	D5VT	2XT5VT	(Pull down) Address bit for external ROM mux in the ROM interleave access mode or GPO[14] or TX data output for SSIF2 (ROM_CONFIG[0] input during the reset period)
AO[2]/GPO[15]/SSSTAT2/ROM_CONFIG[1]	A18	I/O	D5VT	2XT5VT	(Pull down) Address bit for external ROM mux in the ROM interleave access mode or GPO[15] or Status input for SSIF2 (ROM_CONFIG[1] input during the reset period)
AE[3]/GPO[16]/ CLK_CONFIG[0]	A17	I/O	D5VT	2XT5VT	(Pull down) Address bit for external ROM mux in the ROM interleave access mode or GPO[16] (CLK_CONFIG[0] input during the reset period)
AO[3]/GPO[17]/ CLK_CONFIG[1]	D16	I/O	D5VT	2XT5VT	(Pull down) Address bit for external ROM mux in the ROM interleave access mode or GPO[17] (CLK_CONFIG[1] input during the reset period)
D[15:0]	A12,B12,C12,A13,B13,C13,D13,A14,B14,C14,A15,B15,C15,D15,A16,B16	I/O	5VT	2XT5VT	Data bus (16-bit)
RDn	D12	O	-	3XT5VT	Read strobe (active low)
WREn/DOEEn	B9	O	-	4XT5VT	Write strobe for the lower byte (active low) or DRAM output enables selects used for non-interleave mode and interleave modes. DOEEn is used for reading the even address bank (active low).
WROn/DOEOn	C9	O	-	4XT5VT	Write strobe for the higher byte (active low) or DRAM output enables selects used for non-interleave mode and interleave modes. DOEOn is used for reading the odd address bank (active low).
ROMCSn	D10	O	-	2XT5VT	ROM chip select (active low)
CS[1]n	A9	O	-	2XT5VT	I/O chip select (active low).
CS[0]n	G18	O	3V	2XT3V	SRAM chip select (active low) (VRTC powered)
RASn[1:0]	F19,F18	O	-	2XT3V	DRAM row Address select for bank 0 and 1(active low) (VDRAM powered)
CASOn[1:0]	E17,F20	O	-	2XT3V	DRAM column odd address selects used for non-interleave mode and interleave mode. (VDRAM powered)

Table 3-1. Pin Description (2 of 6)

Pin Name	Pin No.	I/O	Input Type	Output Type	Pin Description
CASEn[1:0]	E20,E19	O	-	2XT3V	DRAM column even address selects used for non-interleave mode and interleave mode. (VDRAM powered)
DWRn	D17	O	-	2XT3V	DRAM write. (VDRAM powered)
DMAACK2	K19	O	-	1XT5VT	External DMA acknowledge output (channel 2).
DMAREQ2	F4	I	H5VT	-	(Hysteresis) External DMA request input (channel 2).
FCS0n/PWM[1]	D9	O	-	2XT5VT	Flash memory chip select 0 or PWM channel 1 output
FCS1n/PWM[2]	A8	O	-	2XT5VT	Flash memory chip select 1 or PWM channel 2 output signal.
RESETn	K18	I/O	HU5VT	2XT5VT	(Hysteresis, Pull up) MFC2000 Reset input/output
XIN	G20	I	OSC	-	Crystal oscillator input pin for RTC. (VRTC powered)
XOUT	H20	O	-	OSC	Crystal oscillator output pin for RTC. (VRTC powered)
PWRDWNn	H18	I	H3V	-	(Hysteresis) Indicate the loss of prime power (result in SYSIRQ). (VRTC powered)
WPROTn	H19	O	-	1XT3V	Write Protect during loss of VDD power. **Note**: The functional logic is powered by RTC battery power, but the output drive is powered by DRAM battery power. (VRTC powered)
BATRSTn	G17	I	H3V	-	(Hysteresis) Battery power reset input. (VRTC powered)
EXT_PWRDWN_SELn	G19	I	H3V	-	(Hysteresis) External power-down detector select input (active low)(VRTC powered)
SC_START[0]	V8	O	-	1XT3V	Scanner shift gate control 0
SC_CLK1/SC_CLK2A	U9	O	-	1XT3V	Scanner clock.
SC_LEDCTRL[0]	U7	O	-	1XT3V	Scanner LED control 0
SC_LEDCTRL[1]/ SC_START[1]	Y8	O	-	1XT3V	Scanner LED control 1 or Scanner shift gate control 1
SC_LEDCTRL[2]/ SC_START[2]	W8	O	-	1XT3V	Scanner LED control 2 or Scanner shift gate control 2
SSTXD1	J19	O	-	2XT3V	TX data for SSIF1
SSRXD1	H17	I	HU5VT	-	(Hysteresis, Pull up) RX data for SSIF1
SSCLK1	J20	I/O	H5VT	2XT5VT	(Hysteresis) Clock input or output for SSIF1
GPIO[0]/FWRn/CLAMP	J4	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[0] or flash write enable signal for NAND-type flash memory or scanner clamp control output
GPIO[1]/FRDn	M3	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[1] or flash read enable signal for NAND-type flash memory.
GPIO[2]/DMAREQ1/ SSCLK2	V1	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[2] or DMA channel 1 request input or clock input/output for SSIF2.
GPIO[3]/DMAACK1/ SSRXD2	U4	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[3] or DMA channel 1 acknowledge or RX data for SSIF2
GPIO[4]/CS[2]n	U3	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[4] or I/O chip select [2] (active low)
GPIO[5]/CS[3]n/PWM[3]	U2	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[5]or I/O chip select [3] (active low) or PWM channel 3 output

Table 3-1. Pin Description (3 of 6)

Pin Name	Pin No.	I/O	Input Type	Output Type	Pin Description
GPIO[6]/CS[4]n/ EADC_D[3]	U1	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[6] or I/O chip select [4] (active low) or external ADC data [3] input
GPIO[7]/CS[5]n/	T4	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[7] or I/O chip select [5] (active low).
GPIO[8]/IRQ[11]/ SSSTAT1/SC_CLK1/2B	T3	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[8] or external interrupt [11] or status input for SSIF1 or scan clock output
GPIO[9]/IRQ[13]/ EADC_D[2]	T2	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[9] or external interrupt [13] or external ADC data [2] input
GPIO[10]/RING_DETECT/PWM[4]	T1	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[10] or ring detection input or PWM channel 4 output
GPIO[11]/CPCIN/PWM[0]/ALT TONE	R4	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[11] or calling party control input or ALTTONE output
GPIO[12]/SASCLK/ SMPWRCTRL	R3	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[12] or clock input/output for SASIF or Scan Motor Power Control output
GPIO[13]/SASTXD/ PMPWRCTRL	R2	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[13] or TX data output for SASIF or Print Motor Power Control output
GPIO[14]/SASRXD/ RINGER	R1	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[14] or RX data input for SASIF or ringer output
GPIO[15]/IRQ[16]/ SC_CLK1/2C	P4	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[15] or external interrupt [16] or scan clock output
GPIO[16]/M_TXSIN	P3	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[16] or internal modem
GPIO[17]/M_CLKIN	P2	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[17] or internal modem
GPIO[18]/M_RXOUT	P1	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[18] or internal modem
GPIO[19]/M_SCK/MIRQn	N4	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[19] or internal modem or external modem interrupt input
GPIO[20]/M_STROBE/ MCSn	N3	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[20] or internal modem or external modem chip select
GPIO[21]/M_CNTRL_SIN	N2	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[21] or internal modem
GPIO[22]/EADC_Sample	N1	I/O	H5VT	2XT5VT	(Hysteresis) GPIO[22] or external ADC sample signal output
SM[3:0]/ GPO[7:4]	V7,W7,Y7,U6	O	-	1XT3V	Scan motor control [3:0] pins or GPO[7:4] pins.
PM[0]/SPI_SID/ EADC_D[0]/GPO[0]	U8	I/O	5VT	1XT5VT	Print motor control [0] output or GPO[0] output or data output for SPI or external ADC data [0] input
PM[1]/SPI_SIC/ EADC_D[1]/GPO[1]	W9	I/O	5VT	1XT5VT	Print motor control [1] output or GPO[1] output or clock output for SPI or external ADC data [1] input
PM[2]/SMI0/GPO[2]	Y9	O	-	1XT3V	Print motor control [2] output or GPO[2] output or scan motor current control 0.
PM[3]/SMI1/GPO[3]	Y12	O	-	2XT3V	Print motor control [3] output or GPO[3] output or scan motor current control 1.
TONE	V9	I/O	H5VT	1XT5VT	(Hysteresis) Tone output signal.
PIODIR	C1	O	-	2XT3V	PIOD[7:0] is output when PIODIR is high and PIOD[7:0] is input when PIODIR is low.
STROBEn	A2	I	H5VT	-	(Hysteresis) Input from PC (active low)
AUTOFDn	G3	I	H5VT	-	(Hysteresis) Input from PC (active low)
SLCTINn	G2	I	H5VT	-	(Hysteresis) Input from PC (active low)
INITn	G1	I	H5VT	-	(Hysteresis) Input from PC (active low)
BUSY	A1	O	-	2XT3V	PIO Returned status to PC
ACKn	D3	O	-	2XT3V	PIO Returned status to PC (active low)

Table 3-1. Pin Description (4 of 6)

Pin Name	Pin No.	I/O	Input Type	Output Type	Pin Description
SLCTOUT	C3	O	-	2XT3V	PIO Returned status to PC
PE	B2	O	-	2XT3V	PIO Returned status to PC
FAULTn	B1	O	-	2XT3V	PIO Returned status to PC (active low)
PIOD[7:0]	D2,D1,C2,H4,H3,H2,H1,G4	I/O	H5VT	2XT5VT	(Hysteresis) Driven by PC or MFC2000 and used to send data or address depending on which mode is used
TESTER_MODE	J2	I	HD5VT	-	(Hysteresis) For test only, It must be 'low' for the normal operation
ADCREFp	Y3	I			Positive reference voltage for the scan PADC
ADCREFn	Y2	I			Negative reference voltage for the scan PADC
POWER1	Y1	I			Voltage input for power-down detection circuit 1
POWER2	W4	I			Voltage input for power-down detection circuit 2
ADGA	Y5	-			Scan PADC analog ground
ADVA	V5	-			Scan PADC analog Power
ADGD	U5	-			Scan PADC digital ground
SDAA_SPKR	V12	O			Analog telephone line monitoring output from SSD
ADCV	Y4	-			Scan PADC internal ground
SCIN	W5	I			Analog scan input signal
SENIN[2:0]	V6,W6,Y6	I			Analog sensor inputs for TADC
TCK	W3	I	HD5VT	-	(Hysteresis, Pull down) Test clock input for JTAG. It is positive edge-triggered.
TMS	W2	I	HU5VT	-	(Hysteresis, Pull up) Test mode select input for JTAG. Selects the next state in the TAP state machine.
TRSTn	W1	I	HD5VT	-	(Hysteresis, Pull down) Suggestion by Lauterbach for JTAG: connect this signal to RESETn in normal mode and disconnect in debug mode.
TDI	V4	I	HU5VT	-	(Hysteresis, Pull up) Test data input for JTAG. Serial data input to the JTAG shift register.
TDO	V3	O	-	1XT5VT	Test data output for JTAG. Serial data output from the JTAG shift register.
TEST	V2	I	HD5VT	-	(Hysteresis, Pull down) For test only, It must be 'low' for the normal operation.
SCANMOD	J1	I	HD5VT	-	(Hysteresis, Pull down) For the scan test only, It must be 'low' for normal operations.
P80_SEL	J3	I	H3V	-	P80 DSP test and DFT scan mode select, This pin is only used for the test mode.
PLLREF_XIN	Y15	I	OSC	-	Crystal input pin for PLL
PLLREF_XOUT	W15	O	-	OSC	Crystal output pin for PLL
PLLVDD	U15	-			+3.3V digital power for PLL
PLLVSS	U16	-			+3.3V digital ground for PLL
SDDATA[15:0]	N20,P17,P18,P19,P20,R17,R18,R19,R20,T17,T18,T19,T20,U17,U18,U19	I/O	5VT	2XT5VT	Countach (S)DRAM data bus (16 bits)

Table 3-1. Pin Description (5 of 6)

Pin Name	Pin No.	I/O	Input Type	Output Type	Pin Description
SDADDR[12:0]	V20,W20,Y20, W19,Y19,W18, Y18,W17,Y17,V 16,W16,Y16,V1 5	O	-	2XT3V	Countach (S)DRAM address bus (13 pins)
SDCASn	U20	O	-	2XT3V	Countach (S)DRAM column address strobe (active low)
SDRASn	V19	O	-	2XT3V	Countach (S)DRAM row address strobe (active low)
SDWRn	V18	O	-	2XT3V	Countach (S)DRAM write strobe (active low)
SDCSn	V17	O	-	2XT3V	Countach (S)DRAM chip select
SDCLK100MHz	M19	O	5VT	2XT5VT	Countach (S)DRAM clock
USB_Dp	B8	I/O			Positive data input/output pin for USB
USB_Dn	C8	I/O			Negative data input/output pin for USB
SDAA_PWRCLK	E1	I/O			Positive power/clock output from SSD
SDAA_PWRCLKn	E2	I/O			Negative power/clock output from SSD
SDAA_DIBp	E4	I/O			Positive data input/output pin for SDAA
SDAA_DIBn	E3	I/O			Negative data input/output pin for SDAA
EV_VD[0]/EADC_D[4]/ MREQn	W12	I/O	3V	2XT3V	External video data [0] input for VIP or external ADC data [4] input or Memory Request (active low)-indicates that the following cycle is a memory access.
EV_VD[1]/EADC_D[5]	U12	I/O	3V	2XT3V	External video data [1] input for VIP or external ADC data [5] input
EV_VD[2]/EADC_D[6]/ OPCn	Y13	I/O	3V	2XT3V	External video data [2] input for VIP or external ADC data [6] input or Op Code fetch (active low)-LOW indicates that the processor is fetching an instruction from memory.
EV_VD[3]/EADC_D[7]	W13	I/O	3V	2XT3V	External video data [3] input for VIP or external ADC data [7] input
EV_VD[4]/EADC_D[8]/ MAS[0]	U13	I/O	3V	2XT3V	External video data [4] input for VIP or external ADC data [8] input or Memory access size MAS[1:0]: 00 - byte, 01 - halfword, 10 - word, 11 - Reserved during the normal operation
EV_VD[5]/EADC_D[9]/ MAS[1]	Y14	I/O	3V	2XT3V	External video data [5] input for VIP or external ADC data [9] input or Memory access size MAS[1:0]: 00 - byte, 01 - halfword, 10 - word, 11 - Reserved during the normal operation
EV_VD[6]/EADC_D[10]	W14	I/O	3V	2XT3V	External video data [6] input for VIP or external ADC data [10] input
EV_VD[7]/EADC_D11]/ ABORT	V14	I/O	3V	2XT3V	External video data [7] input for VIP or external ADC data [11] input or aborted bus cycle-the address selected is outside of CS's address ranges.
EV_CLK	V13	I	H3V	-	(Hysteresis) External video clock input
W_Rn	N19	O	D5VT	2XT5VT	(Pull down) The bus access is a read operation when W_Rn is LOW and write when W_Rn is HIGH.
XAKn	N18	O	U5VT	2XT5VT	(Pull up) SIU Transaction Acknowledge. The D[15:0] data will be transferred during this MCLK cycle.

Table 3-1. Pin Description (6 of 6)

Pin Name	Pin No.	I/O	Input Type	Output Type	Pin Description
DMACYC/ CLK_CONFIG[2]	M18	I/O	U5VT	2XT5VT	(Pull up) DMA Cycle-the DMA logic has control of the external bus. (CLK_CONFIG[2] input during the reset period)
WAITn	J17	O	U5VT	2XT5VT	(Pull up) Wait (active low)-reflects the wait states being used by the ARM processor.
CACHEHIT/ JTAG_MODE_SEL	J18	I/O	U5VT	2XT5VT	(Pull up) Cache hit-the ARM is retrieving data from the cache memory (JTAG_MODE_SEL during the reset period, "1" – ARM JTAG selected
SEQ	N17	O	D5VT	2XT5VT	(Pull down) Sequential Address Access. (Used with nMREQ to indicate memory access type. Only required if using co-processor cycles)
SSD_DIBRX	F3	O			Internal test pin. Leave it open.
TX_DATA	F2	O			Internal test pin. Leave it open.
SDAA_GPIO_INT	F1	O			Internal test pin. Leave it open.
VSS	L1,L2,L3,L4,U10,V10,W10,Y10,L17,L18,L19,L20,A11,B11,C11,D11	-			Digital ground (16 pins)
VDD	A10,B10,C10,K1,K2,K3,K4,U11,V11,W11,Y11,K17,M20	-			+3.3V digital power (13 pins)
P80VSS	M1	-			Digital ground for P80 DSP
P80VDD	M2	-			+3.3V digital power for P80 DSP
VGG1	M17	-			+5V Power for the +5V tolerant pads
VGG2	D14	-			+5V Power for the +5V tolerant pads
VGG3	D5	-			+5V Power for the +5V tolerant pads
VGG4	M4	-			+5V Power for the +5V tolerant pads
VDRAM	E18	-			Battery Power for DRAM refresh.
VRTC	F17	-			Battery Power for RTC
(NC)	A3,B3,A4,B4,C4,D4,A5,B5,C5,A6,B6,C6,D6,A7,B7,C7,D7,D8	-			18 RESERVED pins

3.2   Maximum Ratings

Table 3-2. Maximum Ratings

Parameter	Symbol	Limits	Unit
VDD Digital Power	VDD	-0.5 to +4.6	V
Battery Power	VRTC	-0.5 to +4.6	V
	VDRAM	-0.5 to +4.6	V
VGG Digital Power	VGG	-0.5 to +6.0	V
Digital GND	GND	-0.5 to +0.5	V
Digital Input (3V)	VI	-0.5 to +4.6	V
Digital Input (5VT)	VI	-0.5 to +6.0	V
Operating Temperature Range (Commercial)	T	0 to 70	$^{\circ}$C
Storage Temperature Range	Tstg	-40 to 80	$^{\circ}$C
Voltage Applied to Outputs in High Z State (3V)	VHz	-0.5 to 4.6	V
Voltage Applied to Outputs in High Z State (5VT)	VHz	-0.5 to 6.0	V
Static Discharge Voltage (25°C)	ESD	$\pm$2500	V
Latch-up Current (25°C)	Itrig	$\pm$400	mA

3.3   Electrical Characteristics

Table 3-3. Digital Input Characteristics

Symbol	Description	VIL		VIH		Hysteresis	Pullup/Pulldown Resistance
		(V min.)	(V max.)	(V min.)	(V max.)	(V min.)	(K ohms)
3V	3V CMOS input	0	0.8	2.0	VDD	--	--
U3V	3V CMOS input w/pullup	0	0.8	2.0	VDD	--	50-200
H3V	3V CMOS input w/hysteresis	0	0.3*VDD	0.7*VDD	VDD	.5	--
HD3V	3V CMOS input w/hysteresis and pull down	0	0.3*VDD	0.7*VDD	VDD	.5	--
HU3V	3V CMOS input w/hysteresis and pullup	0	0.3*VDD	0.7*VDD	VDD	.5	50-200
H5VT	5V tolerant CMOS input w/hysteresis	0	0.3*VDD	0.7*VDD	5.25	.3	--
U5VT	5V tolerant CMOS input w/pullup	0	0.3*VDD	0.7*VDD	5.25	--	50-200
D5VT	5V tolerant CMOS input w/pulldown	0	0.3*VDD	0.7*VDD	5.25	--	50-200
HU5VT	5V tolerant CMOS input w/hysteresis and pullup	0	0.3*VDD	0.7*VDD	5.25	.3	50-200
HD5VT	5V tolerant CMOS input w/hysteresis and pulldown	0	0.3*VDD	0.7*VDD	5.25	.3	50-200
5VT	5V tolerant CMOS input	0	0.8	2.0	5.25	--	--
OSC**	3V CMOS input	0	0.3*VDD	0.7*VDD	VDD	--	--
** These parameters can only be tested under low speed XIN clock.							

Table 3-4. Output Characteristics

Output Type	Description	VOL (V max)	IOL (mA)	VOH (V min)	IOH (mA)	CL (pF)
1X3V	1X CMOS Output, 3V	0.4	-2.0	2.4	2.0	
1X5VT	1X CMOS Output, 5V tolerant	0.4	-2.0	2.4	2.0	
1XT5VT	1X CMOS Output, tristatable, 5V tolerant	0.4	-2.0	2.4	2.0	
2X3V	2X CMOS Output, 3V	0.4	-4.0	2.4	4.0	
2XT3V	2X CMOS output, tristatable, 3V	0.4	-4.0	2.4	4.0	
2X5VT	2X CMOS output, 5V tolerant	0.4	-4.0	2.4	4.0	
2XT5VT	2X CMOS output, tristatable, 5V tolerant	0.4	-4.0	2.4	4.0	
3XT3V	3X CMOS output, 3V	0.4	-8.0	2.4	8.0	
3XT5VT	3X CMOS output, tristatable, 5V tolerant	0.4	-8.0	2.4	8.0	
4XT5VT	4X CMOS output, tristatable, 5V tolerant	0.4	-12.0	2.4	12.0	

Table 3-5. Power Supply Requirements

Symbol	Description	Operating Voltage		Current*	
		Min. (V)	Max. (V)	Typ. @ 25 ºC(mA)	Max.@ 0ºC (mA)
VGG	Digital Power for 5VT	4.75	5.25		
VDD	Digital Power	3.0	3.6		
GND	Digital GND	0	0		
IDD	Total Digital Current			(TBD)	(TBD)
VBAT	Battery Power	2.7	3.6		
VDRAM	Battery Power	2.7	3.6		
Note: * Maximum power supply current is measured at 3.6V.					

Table 3-6. Battery Power Supply Current Requirements

Operating Voltage (V)	VBAT		VDRAM	
	Typ.@25ºC (μa)	Max.@70ºC (μa)	Typ.@25ºC (μa)	Max.@70ºC (μa)
2.7	4	tbd	100	tbd
3.0	tbd	tbd	tbd	tbd
3.3	6	tbd	tbd	tbd
3.6	tbd	tbd	tbd	tbd
Note: Battery power supply current is measured when a 32KHz crystal is used. The DRAM battery currents that are listed are somewhat dependent on the type of DRAM used. This particular configuration had 1 interleaved DRAM bank in backup mode.				

3.4　Pin Layout
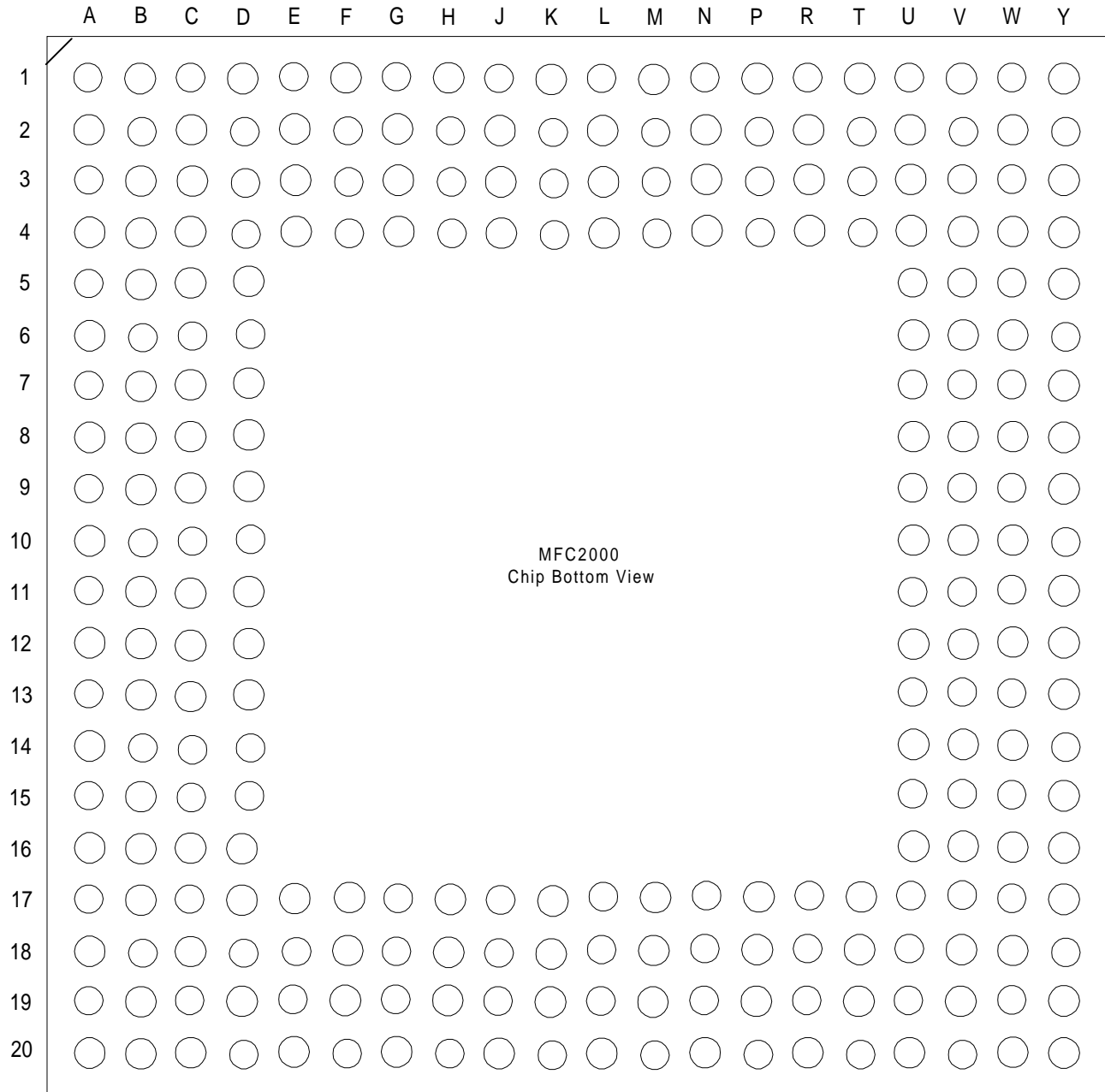
Figure 3-1. MFC2000 BGA Bottom View

4. CPU and Bus Interface

4.1 Memory Map and Chip Select Description

4.1.1 Memory Map

The ARM7TDMI Core is capable of directly accessing 4 GB of memory (A31-A0). The MFC2000 is designed to directly access 64-MB of memory composed of internal and external memory spaces by means of the 26-bit system address bus (A25-A0). The MFC2000 internally decodes address range 00000000H-03FFFFFFH (64 M). Address range 01000000H-01FFFFFFH (16 M) is arranged for the internal registers/memory and external Countach Imaging DSP Subsystem memory. Address range 00000000H-00FFFFFFH (16 M) and Address range 02000000H-03FFFFFFH (32 M) are arranged for the external device/memory use on the ARM Bus. Only 24 address lines (A23-A0) are brought out of the MFC2000 chip, and the lower half and the higher half are multiplexed through the same 12 pins. The 16 MB address range (maximum) can be decoded externally, if necessary. Figure 4-1 and Figure 4-2 show the MFC2000 memory map with memory type designations and locations and provides memory segmentation into select signal ranges.

4.1.1.1   Internal Memory Space

The MFC2000 internal memory occupies 128 kB of the address range from 01FE0000h through 01FFFFFFh).

Internal memory space includes the following:

Cache memory space (64 kB)	(01FE0000h-01FEFFFFh)
(Reserved space) (32 kB)	(01FF0000h-01FF7FFFh)
Internal register space (4 kB)	(01FF8000h-01FF8FFFh)
Internal RAM space (28 kB)	(01FF9000h-01FFFFFFh)

The cache memory space includes the following:

1. The cache memory (4096 bytes)   (01FE0000h-01FE0FFFh)
2. The Tag memory (4096 bytes)   (01FE1000h-01FE1FFFh)
3. (Reserved) (56 kB)   (01FE2000h-01FEFFFFh)

The internal register address range consists of 3 sections:

1. The first (lowest) section (01FF8000h to 01FF87FFh, 2 kB), is reserved for operational registers, i.e., those that are modified during normal operation, but which are not intended to require firmware initialization after reset.
2. The second section (01FF8800h to 01FF8DFFh, 1.5 kB) contains the setup registers, i.e., those that are generally written only once for system initialization after reset.
3. The third section (01FF8E00h to 01FF8FFFh, 512 bytes) is reserved for testing purposes.

> *Note*:   *All internal register accesses are two CPUCLK-cycle operations.*

The internal RAM space includes the following:

1. Bit rotation RAM area
    Bit rotation RAM: 512 halfwords (1 kB)   (01FF9000h-01FF93FFh)
    (Reserved) (3072 bytes)   (01FF9400h-01FF9FFFh)

2. Countach Subsystem memory area
    Countach Scratch Pad (512 bytes) (256 halfwords)(01FFA000h-01FFA1FFh)
    Countach Data DMA Channel 0 (1 halfword)   (01FFA200h-01FFA201h)
    Countach Data DMA Channel 1 (1 halfword)   (01FFA202h-01FFA203h)
    Countach Data DMA Channel 2 (1 halfword)   (01FFA204h-01FFA205h)
    Countach Data DMA Channel 3 (1 halfword)   (01FFA206h-01FFA207h)
    Reserved (344 bytes)   (01FFA208h-01FFA35Fh)
    Countach Program DMA Address (5 bytes)   (01FFA360h-01FFA364h)
    Reserved (155 bytes)   (01FFA365h-01FFA3FFh)
    VSI Buffer (256 bytes) (2x64 halfwords)   (01FFA400h-01FFA4FFh)
    (Reserved) (23296 bytes)   (01FFA500h-01FFFFFFFh)

4.1.1.2   External Countach Imaging DSP Subsystem Memory Space

The external memory space (8 MB) is allocated to the external DRAM/SDRAM on the Countach Imaging DSP Bus Subsystem.

    Countach Imaging DSP Subsystem SDRAM/DRAM (8 M)   (01000000h-017FFFFFh)

4.1.1.3   External Memory Space

The external memory space (up to 48 M) consists of ROM, DRAM/ARAM, Flash memory, SRAM, modem, and variable-use spaces with assigned chip selects. Most external chip selects have programmable address ranges, start locations, wait states, and read and write strobe timing. SRAM and DRAM/ARAM chip select controls are battery-backed up. Refer to Figure 4-1 for the MFC2000 memory map and to Figure 4-2 for the internal memory map.

External memory spaces include the following:

ROMCSn, ROM (4 M)	(00000000h-003FFFFFh)
CS5n, general (4 M)	(00400000h-007FFFFFh)
FCS0n, NOR type Flash memory (2 M)	(00800000h-009FFFFFh)
FCS1n, NOR type Flash memory (2 M)	(00A00000h-00BFFFFFh)
Address location for generating FWRn and FRDn for the NAND type Flash memory	(00C00000h-00C0003Fh)
(Reserved)	(00C00002h-00C1FFFFh)
MCSn, modem (128 K)	(00C20000h-00C2FFFFh)
P80_CSn	(00C30000h - 00C37FFFh)
SDAA_CSn	(00C38000h - 00C3FFFFh)
CS4n, optional general (128 K)	(00C40000h-00C5FFFFh)
CS3n, optional general (128 K)	(00C60000h-00C7FFFFh)
CS2n, optional general (512 K)	(00C80000h-00CFFFFFh)
CS1n, general (1 M)	(00D00000h-00DFFFFFh)
CS0n, SRAM or general (2 M)	(00E00000h-00FFFFFFh)
SDRAM (For internal and Countach Imaging DSP Subsystem) (16 M)	(01000000h-01FFFFFFh)
RAS0n, DRAM or ARAM (16 M)	(02000000h-02FFFFFFh)
RAS1n, DRAM or ARAM (16 M)	(03000000h-03FFFFFFh)

Table 4-1. Fixed-Location and Size Chip Selects

Chip Select	Device
ROMCSn	ROM
FCS1n/FCS0n	NAND- or NOR-type Flash Memory
CS0n	SRAM, or other
CS1n (Optional)	General Use
Optional MCSn (Optional)	External Fax Modem (optional)
CS2n	I/O Devices, or other
CS3n (Optional)	General Use
CS4n (OptionaL0	General Use
CS5n	4 MB ROM, SRAM, or other

DRAM/ARAM chip selects can also be programmed to 1 of 4 sizes (from 512 k to 16 M).

ROM Chip Select (ROMCSn)

ROMCSn selects external ROM located in 4-MB address space 00000000h-003FFFFFh, and is active for read and write accesses. The ROMCtrl register can be used to select 0 to 7 (default) wait states, and 0 or 1 (default) read and write strobe on delays. For customers that choose to use NOR-type flash memory in the ROM address area, the write operation is also allowed in the ROM address area.

Chip Select 5 (CS5n)

CS5n is an active Read/Write select signal for the 4 MB address range (00400000h to 007FFFFFh) directly below the ROMCSn address range. The CS5Ctrl register can be used to select 0 to 7 (default) wait states, and 0 or 1 (default) read and write strobe on delays. GPIO[7] (default) can be configured as CS5n using the GPIO[7]/CS5n bit of the GPIOConfig register.

SRAM Chip Select 0 (CS0n)

CS0n is designed for use in selecting external SRAM, but can also be used for other purposes. It has 2 MB address range (00E00000h to 00FFFFFFh). The CS0n can also be programmed for 0 (default) to 7 wait states, 0 (default) or 1 read and write strobe on delays, and normal (default) or early write strobe off times using the CS0Ctrl register.

DRAM Chip Select (RASn[1:0], CASOn[1:0] and CASEn[1:0])

DRAM address space can be selected in 2 separate memory blocks (Bank 0: RASn[0] and Bank 1: RASn[1]). Separate control bits are provided in the Backup Configuration register to enable and disable each of the memory banks (Default: Bank0 is enabled and 8-bit DRAM is selected). Non-interleaved DRAM accesses and 2-way interleaved DRAM accesses are supported. CASOn[1:0] and CASEn[1:0] are used differently for different access modes. RASn is asserted before CASn for normal read and write operations. Also, RAS can be kept active and CASn is toggled to do burst mode operations. CASn-Before-RASn refresh mode is the only refresh mode for MFC2K (For more DRAM information, see the DRAM Controller section.)

The address ranges of the two memory banks (RAS0n and RAS1n) are continuous around the midpoint of the DRAM memory bank. The RASn[1] starting address is 03000000h and grows larger based on the size of the memory. The end of the RASn[0] bank ends at 03000000h and grows smaller from that point. Each bank has separate configuration controls. The memory range is programmed through the address multiplexer selections for bank 0 and bank 1 in the DRAMCtrl register.

Flash Memory Chip Selects (FCS1n and FCS2n)

FCS0n and FCS1n are multiplexed with PWM[1] and PWM[2] and output through FCS0n/PWM[1] and FCS1n/PWM[2] pins. After reset, the Flash disable bit (bit 0) of the FlashCtrl register is 0 and the FCS0n/PWM[1] and FCS1n/PWM[2] pins are used as FCS0n and FCS1n. FCS0n and FCS1n can access either NOR-type (default) or NAND-type flash memory, selectable with the NANDFlashEnb bit (bit 6) of the FlashCtrl register. When enabled for NOR-type flash memory (default), FCS1n can be activated by accessing the 2-MB (00A00000h-00BFFFFFh) flash memory address area. FCS0n can be activated by accessing the 2-MB (00800000h - 009FFFFFh) flash memory address area. Firmware controls the flash memory access block size. If enabled for NAND-type flash memory, FCS0n and FCS1n revert to the GPO function and output bit 9 and bit 8 values of the FlashCtrl register . 0 to 7 (default) wait states and normal (default) or early off of the write strobe can be chosen using the FlashCtrl register described in the SIU section.

Modem Chip Select (MCSn)

The 128 kB address space from 00C20000h to 00C2FFFFh is reserved for the external modem and selected with MCSn. It is muxed with the M_STROBE signal of the modem IA on the pin. M_STROBE is usually used to interface the embedded DSP to the external modem IA if the embedded V.34 modem DSP is used. MCSn can be selected and muxed out for the external modem if the embedded modem DSP is not used. MCSn can be programmed for 0 to 7 (default) wait states, 0 (default) or 1 read and write strobe on delays, and normal (default) or early write strobe off times using the MCSCtrl register.

P80 Chip Select (P80_CSn)

Address space form 00C3000 to 00C7FFF has been reserved for the P80 functions.

Smart Data Access Arraignment (SDAA_CSn)

Address space form 00C3800 to 00CFFFF has been reserved for the SDAA functions.

4.1.1.4   External I/O Chip Selects

Chip Select [2] (CS2n)

The 512 kB address space from 00C80000h to 00CFFFFFh is selected using the external I/O chip selects CS2n. GPIO[4] (default) can be configured as CS2n using the GPIO[4]/CS2n bit of the GPIOConfig register. CS2n can be programmed for 0 (default) to 7 wait states, 0 (default) to 3 read and write strobe delays, and normal (default) or early write strobe off times using the CS2Ctrl register.

Chip Select [4:3] (CS4n-CS3n)(optional)

The 256 kB address space from 00C40000h to 00C7FFFFh can optionally be selected using the two external I/O chip selects CS4n and CS3n. These chip selects are configured identically to CS2n.

GPIO[6] (default) can be configured as CS4n using the GPIO[6]/CS4n bit of the GPIOConfig register. Likewise, GPIO[5] (default) can be configured as CS3n by using the GPIO[5]/CS3n bit in the GPIOconfig1 register.

The top 128 kB (00C40000h to 00C5FFFFh) are addressed by CS4n. CS4n is active for read-access only (internally gated with the read strobe) when the CS4nReadOnly bit (bit 8) of the SIUConfig register is 1. CS4n is active for both read and write access when the CS4nReadOnly bit (bit 8) of the SIUConfig register is 0. The next 128 kB (00C60000h to 00C7FFFFh) is addressed by CS3n. CS3n is active for write-access only (internally gated with the write even strobe) when the CS3nWriteOnly bit (bit 7) of the SIUConfig register is 1. If the external I/O device using CS3n is a 16-bit device, 16-bit access must be done. No high-byte or low-byte access can be done. CS3n is active for both read and write access when the CS3nWriteOnly bit (bit 7) of the SIUConfig register is 0.

Chip Select 1 (CS1n)

The next address range below those of CS4n-CS2n is the 1-MB range (00D00000h to 00DFFFFFh) selected by CS1n. CS1n can be programmed for 0 (default) to 7 wait states, 0 (default) to 3 read and write strobe delays, and normal (default) or early write strobe off times using the CS1Ctrl register.
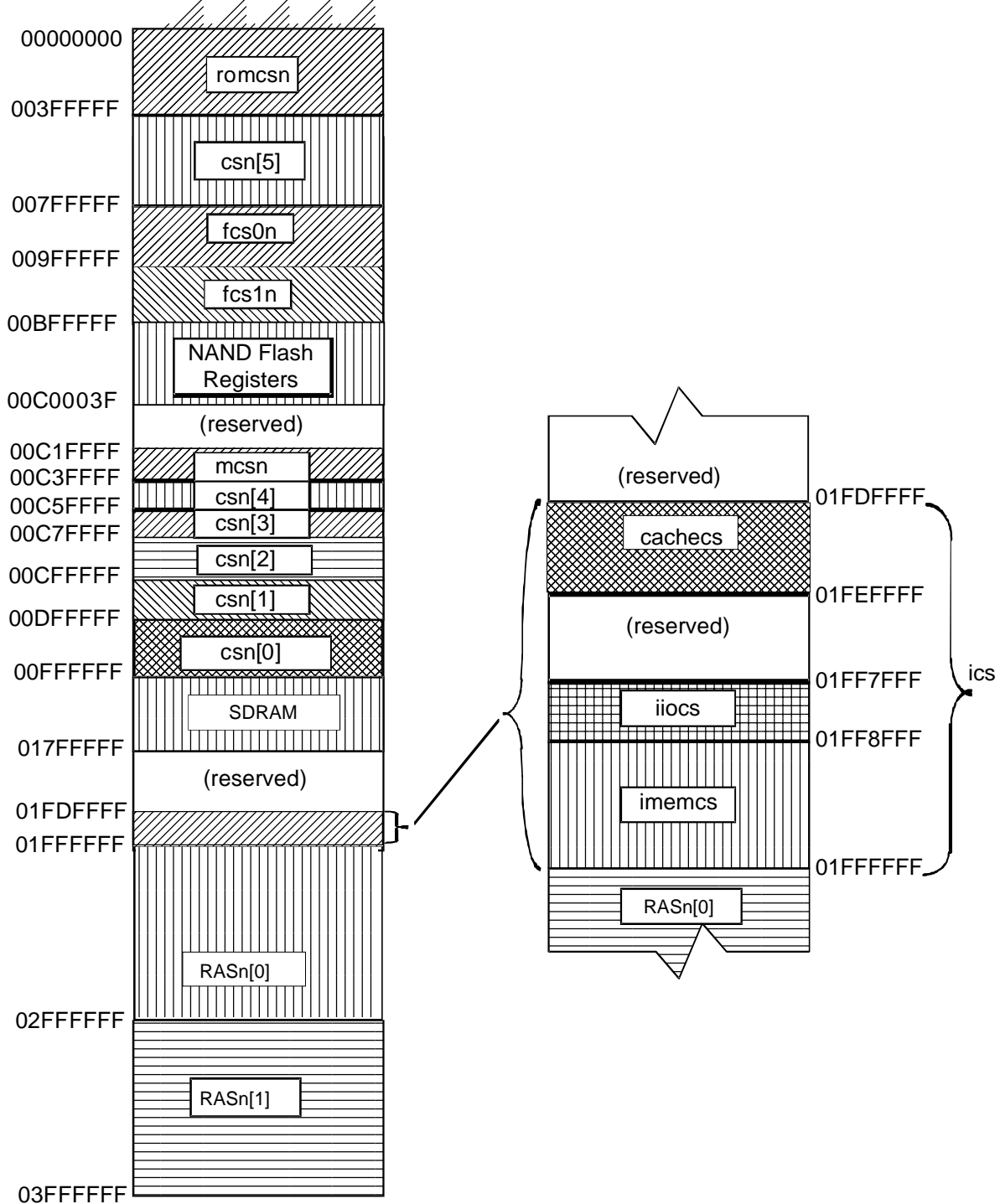
Figure 4-1. MFC2000 Memory Map
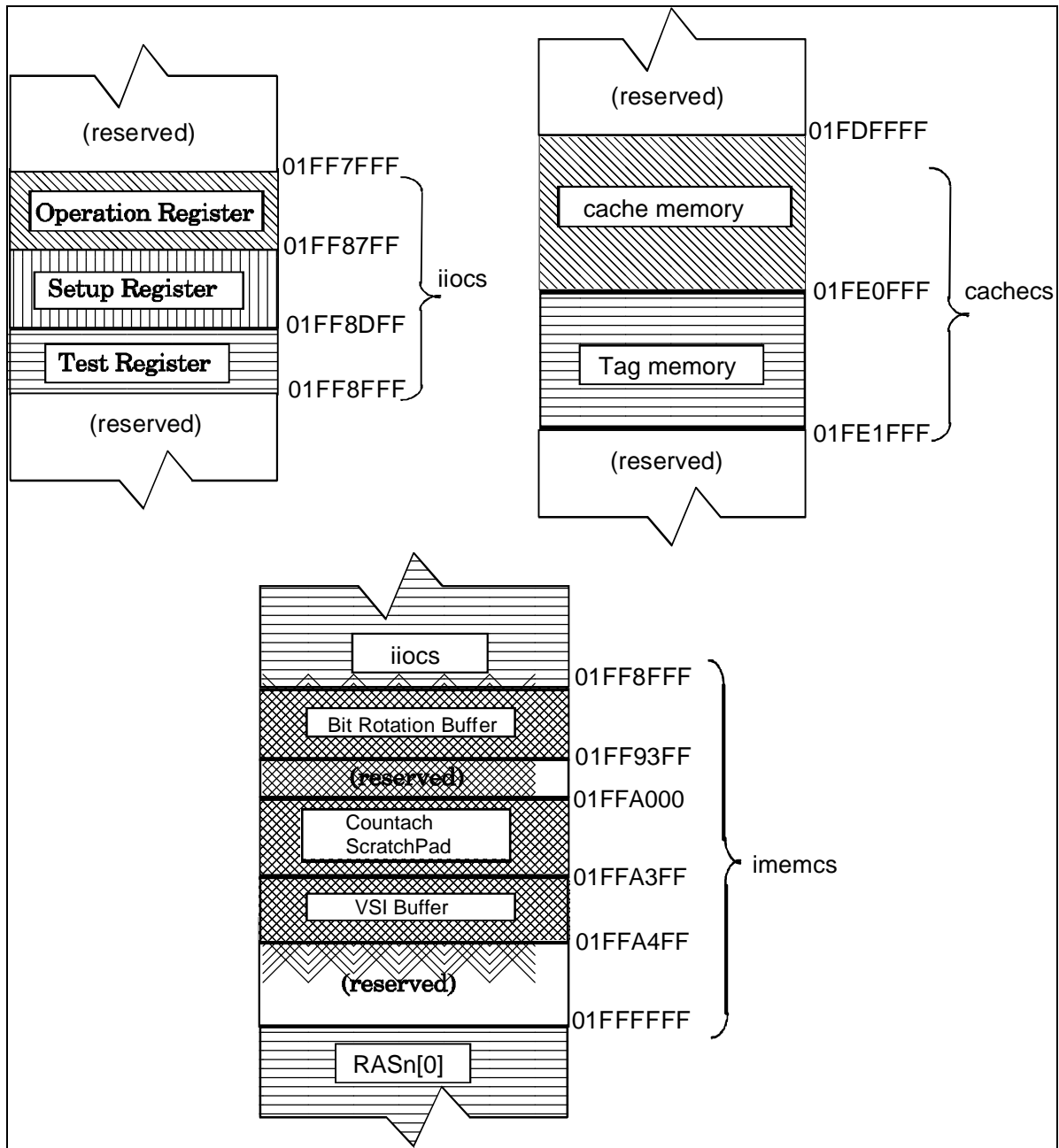
Figure 4-2. MFC2000 Internal Memory Map

4.1.2   Register Map

Table 4-2. Operation Register Map (1 of 9)

Operation registers are located from 01FF8000H to 01FF87FFH.

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF8000-01	TADCCtrl	TADC	R/W
01FF8002-03	TADCInsData	TADC	R
01FF8004-05	TADCCh0Data	TADC	R
01FF8006-07	TADCCh1Data	TADC	R
01FF8008-09	TADCCh2Data	TADC	R
01FF800A-1F	(Not Used)	—	—
01FF8020-21	IRQFIQEvent1	Interrupt Controller	R
01FF8022-23	IRQFIQEvent2	Interrupt Controller	R(Bit[6:2], Bit[0]) R/W(Bit[1])
01FF8024-25	IRQEnable1	Interrupt Controller	R/W
01FF8026-27	IRQEnable2	Interrupt Controller	R/W
01FF8028-29	FIQEnable1	Interrupt Controller	R/W
01FF802A-2B	FIQEnable2	Interrupt Controller	R/W
01FF802C-2D	EIRQConfigClr	Interrupt Controller	R/W
01FF802E-2F	IRQTimer1	Interrupt Controller	R/W
01FF8030-31	IRQTimer2	Interrupt Controller	R/W
01FF8032-3F	(Not Used)	—	—
01FF8040	WatchdogEnRetrigger	Watchdog Timer	W
01FF8042	WatchdogInterval	Watchdog Timer	R/W
01FF8044	HWVersion	Watchdog Timer	R
01FF8046	ProductCode	Watchdog Timer	R
01FF8048-4B	(Not Used)	—	—
01FF804C-4D	SSCurTimer1	Scan/Print Motor Controller	R/W
01FF804E-4F	SSCurTimer2	Scan/Print Motor Controller	R/W
01FF8050-51	SStepCtrl	Scan/Print Motor Controller	R/W
01FF8052-53	SStepTimer	Scan/Print Motor Controller	R/W
01FF8054-55	SSDelayTimer	Scan/Print Motor Controller	R/W
01FF8056-57	SMPattern/GPO	Scan/Print Motor Controller	R/W
01FF8058-59	VPStepCtrl	Scan/Print Motor Controller	R/W
01FF805A-5B	VPStepTimer	Scan/Print Motor Controller	R/W
01FF805C-5D	VPMPattern/GPO	Scan/Print Motor Controller	R/W
01FF805E-5F	(Not Used)	—	—
01FF8060-61	RotCtrl	Bit Rotation Block	R/W
01FF8062-63	RotPackeddata	Bit Rotation Block	R/W
01FF8064-67	(Not Used)	—	—
01FF8069-6B	TotalBinDatCntr	Bi-level Resolution Conversion	R
01FF806C-6D	FirstBlkDatCnt	Bi-level Resolution Conversion	R
01FF806E-6F	LastBlkDatCnt	Bi-level Resolution Conversion	R
01FF8070-71	BiRCInFIFO0	Bi-level Resolution Conversion	R/W

Table 4-2. Operation Register Map (2 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF8072-73	BiRCInFIFO1	Bi-level Resolution Conversion	R/W
01FF8074-75	BiRCInFIFO2	Bi-level Resolution Conversion	R/W
01FF8076-77	BiRCInFIFO3	Bi-level Resolution Conversion	R/W
01FF8078-79	BiRCInHold	Bi-level Resolution Conversion	R/W
01FF807A-7B	BiRCInFIFOCtrl	Bi-level Resolution Conversion	R/W
01FF807C-7D	BiResConRatio	Bi-level Resolution Conversion	R/W
01FF807E-7F	BiResConCtrl	Bi-level Resolution Conversion	R/W
01FF8080-81	BiRCOutFIFO0	Bi-level Resolution Conversion	R/W
01FF8082-83	BiRCOutFIFO1	Bi-level Resolution Conversion	R/W
01FF8084-85	BiRCOutFIFO2	Bi-level Resolution Conversion	R/W
01FF8086-87	BiRCOutFIFO3	Bi-level Resolution Conversion	R/W
01FF8088-89	BiRCOutHold	Bi-level Resolution Conversion	R/W
01FF808A-8B	BiRCOutFIFOCtrl	Bi-level Resolution Conversion	R/W
01FF808C-8D	SinglingMask	Bi-level Resolution Conversion	R/W
01FF808E-8F	HSZeroNo	Bi-level Resolution Conversion	R/W
01FF8090-1	Sec_Min	Battery RTC	R(bit[7]), R/DW(bit[5:0]) R(bit[15]), R/DW(bit[13:8])
01FF8092-3	Hour_Day	Battery RTC	R(bit[7]), R/DW(bit[4:0]) R(bit[15]), R/DW(bit[12:8])
01FF8094-5	Month_Year	Battery RTC	R(bit[7]), R/DW(bit[3:0]) R(bit[15]), R/DW(bit[12:8])
01FF8096-7	RTCCtrl	Battery RTC	DR/DW
01FF8098-9	BackupConfig	Battery RTC	R/W
01FF809A-F	(Not Used)	—	—
01FF80A0-1	LockEnb	Prime Power Reset Logic	DW
01FF80A2-7	(Not Used)	—	—
01FF80A8-9	CPCThreshold	CPC Logic	R/W
01FF80AA-B	CPCStatCtrl	CPC Logic	R/W
01FF80AC-F	(Not Used)	—	—
01FF80B0-1	ToneGenF1	ToneGen Block	R/W
01FF80B2-3	ALTToneGen	ToneGen Block	R/W
01FF80B4-5	BellCtrl	Bell Ringer	R/W
01FF80B6-7	BellPeriod	Bell Ringer	R/W
01FF80B8-9	BellPhase	Bell Ringer	R/W
01FF80BA-B	ToneGenF2	ToneGen Block	R/W
01FF80BC-D	ToneGenSwitch	ToneGen Block	R/W
01FF80BE-F	ToneGenTotal	ToneGen Block	R/W
01FF80C0-C1	PWMCh0Ctrl	PWM Logic	R/W
01FF80C2-C3	PWMCh1Ctrl	PWM Logic	R/W
01FF80C4-C5	PWMCh2Ctrl	PWM Logic	R/W
01FF80C6-C7	PWMCh3Ctrl	PWM Logic	R/W

Table 4-2. Operation Register Map (3 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF80C8-C9	PWMCh4Ctrl	PWM Logic	R/W
01FF80CA-CF	(Not Used)	—	—
01FF80D0-DF	(Not Used)	—	—
01FF80E0-EF	(Reserved)	—	—
01FF80F0-1	SASCmd	SASIF	R/W
01FF80F2-3	SASData	SASIF	R/W
01FF80F4-5	SASDiv	SASIF	R/W
01FF80F6-7	(Not Used)	—	—
01FF80F8-9	SASIRQSTS	SASIF	R/W
01FF80FA-FF	(Not Used)	—	—
01FF8100	SSCmd	SSIF	R/W
01FF8102	SSData	SSIF	R/W
01FF8104	SSDiv	SSIF	R/W
01FF8106-07	(Not Used)	—	—
01FF8108-09	SSCmd2	SSIF2	R/W
01FF810A-0B	SSData2	SSIF2	R/W
01FF810C-0D	SSDiv2	SSIF2	R/W
01FF810E-0F	(Not Used)	—	—
01FF8110-11	T4DataFIFO0	T4/T6 Block	R/W
01FF8112-13	T4DataFIFO1	T4/T6 Block	R/W
01FF8114-15	T4DataFIFO2	T4/T6 Block	R/W
01FF8116-17	T4DataFIFO3	T4/T6 Block	R/W
01FF8118-19	T4DataHold	T4/T6 Block	R/W
01FF811A-1B	T4DataFIFOCtrl	T4/T6 Block	R/W
01FF811C-1D	T4DataPort	T4/T6 Block	R/W
01FF811E-1F	T4DataPortTfr	T4/T6 Block	R/W
01FF8120-4F	(Not Used)	—	—
01FF8150-51	T4RefDataFIFO0	T4/T6 Block	R/W
01FF8152-53	T4RefDataFIFO1	T4/T6 Block	R/W
01FF8154-55	T4RefDataFIFO2	T4/T6 Block	R/W
01FF8156-57	T4RefDataFIFO3	T4/T6 Block	R/W
01FF8158-59	T4RefDataHold	T4/T6 Block	R/W
01FF815A-5B	T4RefDataFIFOCtrl	T4/T6 Block	R/W
01FF815C-5D	T4RefDataPort	T4/T6 Block	R/W

Table 4-2. Operation Register Map (4 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF815E-5F	T4RefDataPortTfr	T4/T6 Block	R/W
01FF8160-61	T4CurDataFIFO0	T4/T6 Block	R/W
01FF8162-63	T4CurDataFIFO1	T4/T6 Block	R/W
01FF8164-65	T4CurDataFIFO2	T4/T6 Block	R/W
01FF8166-67	T4CurDataFIFO3	T4/T6 Block	R/W
01FF8168-69	T4CurDataHold	T4/T6 Block	R/W
01FF816A-6B	T4CurDataFIFOCtrl	T4/T6 Block	R/W
01FF816C-6D	T4CurDataPort	T4/T6 Block	R/W
01FF816E-6F	T4CurDataPortTfr	T4/T6 Block	R/W
01FF8170-71	T4Config	T4/T6 Block	R/W
01FF8172-73	T4Control	T4/T6 Block	R/W
01FF8174-75	T4 Status	T4/T6 Block	R
01FF8176-77	T4IntMask	T4/T6 Block	R/W
01FF8178-79	T4Bytes	T4/T6 Block	R/W
01FF817A-7B	T4FIFOBitRem	T4/T6 Block	R/W
01FF817C-7F	(Not Used)	—	—
01FF8180-81	(Not Used)	—	—
01FF8182-83	DMA1Config	DMA Controller	R/W
01FF8184-85	DMA1CntLo	DMA Controller	R/W
01FF8186-87	DMA1CntHi	DMA Controller	R/W
01FF8188-89	DMA2CntLo	DMA Controller	R/W
01FF818A-8B	DMA2CntHi	DMA Controller	R/W
01FF818C-8D	DMA2BufCntLo	DMA Controller	R/W
01FF818E-8F	DMA2BufCntHi	DMA Controller	R/W
01FF8190-91	DMA3CntLo	DMA Controller	R/W
01FF8192-93	DMA3CntHi	DMA Controller	R/W
01FF8194-95	DMA4CntLo	DMA Controller	R/W
01FF8196-97	DMA4CntHi	DMA Controller	R/W
01FF8198-99	DMA5CntLo	DMA Controller	R/W
01FF819A-9B	DMA5CntHi	DMA Controller	R/W
01FF819C-9D	DMA6CntLo	DMA Controller	R/W
01FF819E-9F	DMA6CntHi	DMA Controller	R/W
01FF81A0-A1	DMA7CntLo	DMA Controller	R/W
01FF81A2-A3	DMA7CntHi	DMA Controller	R/W
01FF81A4-A5	DMA8CntLo	DMA Controller	R/W
01FF81A6-A7	DMA8CntHi	DMA Controller	R/W
01FF81A8-A9	DMA9CntLo	DMA Controller	R/W
01FF81AA-AB	DMA9CntHi	DMA Controller	R/W
01FF81AC-AD	DMA10CntLo	DMA Controller	R/W
01FF81AE-AF	DMA10CntHi	DMA Controller	R/W

Table 4-2. Operation Register Map (5 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF81B0-B1	DMA0Config	DMA Controller	R/W
01FF81B2-B3	DMA2Config	DMA Controller	R/W
01FF81B4-B5	DMA2BlkSize	DMA Controller	R/W
01FF81B6-B7	DMA2BufBlkSize	DMA Controller	R/W
01FF81B8-B9	(Not Used)	—	—
01FF81BA-BB	DMA5BlkSize	DMA Controller	R/W
01FF81BC-BD	DMA6/10Throttle	DMA Controller	R/W
01FF81BE-BF	DMA9BlkSize	DMA Controller	R/W
01FF81C0-C1	DMA10BlkSize	DMA Controller	R/W
01FF81C2-C3	DMAIncConfig	DMA Controller	R/W
01FF81C4-C5	DMACntEnbConfig	DMA Controller	R/W
01FF81C6-C7	DMAEndian	DMA Controller	R/W
01FF81C8-C9	DMAUSB0CntLo	DMA Controller	R/W
01FF81CA-CB	DMAUSB0CntHi	DMA Controller	R/W
01FF81CC-CD	DMAUSB0BlkSiz	DMA Controller	R/W
01FF81CE-CF	DMAUSB1CntLo	DMA Controller	R/W
01FF81D0-D1	DMAUSB1CntHi	DMA Controller	R/W
01FF81D2-D3	DMAUSB1BlkSiz	DMA Controller	R/W
01FF81D4-D5	DMAUSB2CntLo	DMA Controller	R/W
01FF81D6-D7	DMAUSB2CntHi	DMA Controller	R/W
01FF81D8-D9	DMAUSB2BlkSiz	DMA Controller	R/W
01FF81DA-DB	DMAUSB3CntLo	DMA Controller	R/W
01FF81DC-DD	DMAUSB3CntHi	DMA Controller	R/W
01FF81DE-DF	DMAUSB3BlkSiz	DMA Controller	R/W
01FF81E0-E1	DMA11CntLo	DMA Controller	R/W
01FF81E2-E3	DMA11CntHi	DMA Controller	R/W
01FF81E4-E5	DMA11BlkSiz	DMA Controller	R/W
01FF81E6-E7	DMA12CntLo	DMA Controller	R/W
01FF81E8-E9	DMA12CntHi	DMA Controller	R/W
01FF81EA-EB	DMA12BlkSiz	DMA Controller	R/W
01FF81EC-FF	(Not Used)	—	—
01FF8200-01	PIOCtrl	PIO	R/W
01FF8202-03	PIOIF	PIO	R/W
01FF8204-05	PIOData	PIO	R/W
01FF8206-07	PIOAckPW	PIO	R/W
01FF8208-09	PIORevDataSTS	PIO	R/W
01FF820A-0B	PIODataBusSTS	PIO	R/W
01FF820C-0D	PIOHostTimeOut	PIO	R/W
01FF820E-0F	PIOIRQSTS	PIO	R/W
01FF8210-11	PIOIRQMask	PIO	R/W

Table 4-2. Operation Register Map (6 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF8212-13	PIOFIFOIF	PIO	R/W
01FF8214-1F	(Not Used)	—	—
01FF8220-21	PIOOutFIFO0	PIO	R/W
01FF8222-23	PIOOutFIFO1	PIO	R/W
01FF8224-25	PIOOutFIFO2	PIO	R/W
01FF8226-27	PIOOutFIFO3	PIO	R/W
01FF8228-29	PIOOutHold	PIO	R/W
01FF822A-2B	PIOOutFIFOCtrl	PIO	R/W
01FF822C-2F	(Not Used)	—	—
01FF8230-31	PIOInFIFO0	PIO	R/W
01FF8232-33	PIOInFIFO1	PIO	R/W
01FF8234-35	PIOInFIFO2	PIO	R/W
01FF8236-37	PIOInFIFO3	PIO	R/W
01FF8238-39	PIOInHold	PIO	R/W
01FF823A-3B	PIOInFIFOCtrl	PIO	R/W
01FF823C-4F	(Not Used)	—	—
01FF8250-5B	(Not Used)	—	—
01FF825C-5D	VSHiAddr1	Countach Bus System - CDMAC	R/W
01FF825E-5F	VSLoAddr1	Countach Bus System - CDMAC	R/W
01FF8260-61	VSHiAddr2	Countach Bus System - CDMAC	R/W
01FF8262-63	VSLoAddr2	Countach Bus System - CDMAC	R/W
01FF8264-65	VSHiAddrStep	Countach Bus System - CDMAC	R/W
01FF8266-67	VSLoAddrStep	Countach Bus System - CDMAC	R/W
01FF8268-69	VSMode	Countach Bus System - CDMAC	R/W
01FF826A-6B	ABc2aBlkSiz	Countach Bus System - CDMAC	R/W
01FF826C-6D	ABa2cHiAddr	Countach Bus System - CDMAC	R/W
01FF826E-6F	ABa2cLoAddr	Countach Bus System - CDMAC	R/W
01FF8270-71	ABc2aHiAddr	Countach Bus System - CDMAC	R/W
01FF8272-73	ABc2aLoAddr	Countach Bus System - CDMAC	R/W
01FF8274-75	ABa2cThrottle	Countach Bus System - CDMAC	R/W
01FF8276-77	ABc2aThrottle	Countach Bus System - CDMAC	R/W
01FF8278-7F	(Not Used)	—	—
01FF8280-81	DRAMConfig	Countach Bus System - SDRAMC	R/W
01FF8282-83	ABIIrqStat	Countach Bus System - ABI	R/W
01FF8284-85	ABIIrqEnable	Countach Bus System - ABI	R/W
01FF8286-87	ABICountachCtrl	Countach Bus System - ABI	R/W
01FF8288-89	VSIMode	Countach Bus System - VSI	R/W
01FF828A-A7	(Not Used)	—	—
01FF82A8-A9	DefRdHiAddr	Countach Bus System - CBU	R/W
01FF82AA-AB	DefRdLoAddr	Countach Bus System - CBU	R/W

Table 4-2. Operation Register Map (7 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF82AC-AD	DefWrHiAddr	Countach Bus System – CBU	R/W
01FF82AE-AF	DefWrLoAddr	Countach Bus System – CBU	R/W
01FF82B0-B1	DefRdData	Countach Bus System – CBU	R/W
01FF82B2-B3	DefWrData	Countach Bus System – CBU	R/W
01FF82B4-FF	(Not Used)	—	—
01FF8300-01	ABIA2Cbuff1	Countach Bus System – ABI	R/W
01FF8302-03	ABIA2Cbuff2	Countach Bus System – ABI	R/W
01FF8304-05	ABIA2Cbuff3	Countach Bus System – ABI	R/W
01FF8306-07	ABIA2Cbuff4	Countach Bus System – ABI	R/W
01FF8308-09	ABIC2Abuff1	Countach Bus System – ABI	R/W
01FF830A-0B	ABIC2Abuff2	Countach Bus System – ABI	R/W
01FF830C-0D	ABIC2Abuff3	Countach Bus System – ABI	R/W
01FF830E-0F	ABIC2Abuff4	Countach Bus System – ABI	R/W
01FF8310-11	CSIDMABuff1	Countach Bus System – CSI	R/W
01FF8312-13	CSIDMABuff2	Countach Bus System – CSI	R/W
01FF8314-15	CSIDMABuff3	Countach Bus System – CSI	R/W
01FF8316-17	CSIDMABuff4	Countach Bus System – CSI	R/W
01FF8318-FF	(Not Used)	—	—
01FF8400-FF	(Not Used)	—	—
01FF8500-3F	(Not Used)	—	—
01FF8540-41	ScanCtrl	Video/Scan Controller	R/W
01FF8542-43	ScanCtrlStat	Video/Scan Controller	R only
01FF8544-45	VSCIRQStatus	Video/Scan Controller	R/W
01FF8546-47	VSCCtrl	Video/Scan Controller	R/W
01FF8548-49	VidCaptureCtrl	Video/Scan Controller	R/W (Bit[8] - R only)
01FF854A-4B	SPI_Ctrl	Video/Scan Controller	R/W (Bit[8] - R only)
01FF854C-4D	SPI_Stat	Video/Scan Controller	R only
01FF854E-7F	(Not Used)	—	—
01FF8580-81	USBEP1FIFO1	USB Interface	R/W
01FF8582-83	USBEP1FIFO2	USB Interface	R/W
01FF8584-85	USBEP1FIFO3	USB Interface	R/W
01FF8586-87	USBEP1FIFO4	USB Interface	R/W
01FF8588-89	USBEP1Hold	USB Interface	R/W
01FF858A-8B	USBEP1Ctrl	USB Interface	R/W
01FF858C-8D	USBEP1Data	USB Interface	R/W
01FF858E-8F	USBEP1Tran	USB Interface	R/W
01FF8590-9F	(Not Used)	—	—
01FF85A0-A1	USBEP2FIFO1	USB Interface	R/W
01FF85A2-A3	USBEP2FIFO2	USB Interface	R/W
01FF85A4-A5	USBEP2FIFO3	USB Interface	R/W
01FF85A6-A7	USBEP2FIFO4	USB Interface	R/W
01FF85A8-A9	USBEP2Hold	USB Interface	R/W

Table 4-2. Operation Register Map (8 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF85AA-AB	USBEP2Ctrl	USB Interface	R/W
01FF85AC-AD	USBEP2Data	USB Interface	R/W
01FF85AE-AF	USBEP2Tran	USB Interface	R/W
01FF85B0-B1	USBEP3FIFO1	USB Interface	R/W
01FF85B2-B3	USBEP3FIFO2	USB Interface	R/W
01FF85B4-B5	USBEP3FIFO3	USB Interface	R/W
01FF85B6-B7	USBEP3FIFO4	USB Interface	R/W
01FF85B8-B9	USBEP3Hold	USB Interface	R/W
01FF85BA-BB	USBEP3Ctrl	USB Interface	R/W
01FF85BC-BD	USBEP3Data	USB Interface	R/W
01FF85BE-BF	USBEP3Tran	USB Interface	R/W
01FF85C0-C1	USBEP4FIFO1	USB Interface	R/W
01FF85C2-C3	USBEP4FIFO2	USB Interface	R/W
01FF85C4-C5	USBEP4FIFO3	USB Interface	R/W
01FF85C6-C7	USBEP4FIFO4	USB Interface	R/W
01FF85C8-C9	USBEP4Hold	USB Interface	R/W
01FF85CA-CB	USBEP4Ctrl	USB Interface	R/W
01FF85CC-CD	USBEP4Data	USB Interface	R/W
01FF85CE-CF	USBEP4Tran	USB Interface	R/W
01FF85D0-D1	USBEP0Buf12	USB Interface	R/W
01FF85D2-D3	USBEP0Buf34	USB Interface	R/W
01FF85D4-D5	USBEP0Buf56	USB Interface	R/W
01FF85D6-D7	USBEP0Buf78	USB Interface	R/W
01FF85D8-D9	USBVenBuf12	USB Interface	R/W
01FF85DA-DB	USBVenBuf34	USB Interface	R/W
01FF85DC-DD	USBVenBuf56	USB Interface	R/W
01FF85DE-DF	USBVenBuf78	USB Interface	R/W
01FF85E0-E1	USBVenStDat12	USB Interface	R/W
01FF85E2-E3	USBVenStDat34	USB Interface	R/W
01FF85E4-E5	USBVenStDat56	USB Interface	R/W
01FF85E6-E7	USBVenStDat78	USB Interface	R/W
01FF85E8-E9	USBDesAdr	USB Interface	R/W
01FF85EA-EB	USBIRQ	USB Interface	R/W
01FF85EC-ED	USBSoftReset	USB Interface	W
01FF85EE-EF	USBStall	USB Interface	W
01FF85F0-F1	USBPOSTDat1/2	USB Interface	R
01FF85F2-F3	USBPOSTDat3/4	USB Interface	R
01FF85F4-F5	USBPOSTDat5/6	USB Interface	R
01FF85F6-F7	USBPOSTDat7/8	USB Interface	R
01FF85F8-FF	(Not Used)	USB Interface	—
01FF85EC-FF	(Not Used)	—	—

Table 4-2. Operation Register Map (9 of 9)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF8600-01	SASTxFIFOHW0	SASIF	R/W
01FF8602-03	SASTxFIFOHW1	SASIF	R/W
01FF8604-05	SASTxFIFOHW2	SASIF	R/W
01FF8606-07	SASTxFIFOHW3	SASIF	R/W
01FF8608-09	SASTxFIFOHW4	SASIF	R/W
01FF860A-0B	SASTxFIFOHW5	SASIF	R/W
01FF860C-0D	SASTxFIFOHW6	SASIF	R/W
01FF860E-0F	SASTxFIFOHW7	SASIF	R/W
01FF8610-11	SASRxFIFOHW0	SASIF	R/W
01FF8612-13	SASRxFIFOHW1	SASIF	R/W
01FF8614-15	SASRxFIFOHW2	SASIF	R/W
01FF8616-17	SASRxFIFOHW3	SASIF	R/W
01FF8618-19	SASRxFIFOHW4	SASIF	R/W
01FF861A-1B	SASRxFIFOHW5	SASIF	R/W
01FF861C-1D	SASRxFIFOHW6	SASIF	R/W
01FF861E-1F	SASRxFIFOHW7	SASIF	R/W
01FF8620-7FF	(Not Used)	—	—

Table 4-3. Setup Registers (1 of 2)

Setup Registers are located from 01FF8800H to 01FF8DFFH.

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF8800-01	SIUConfig	SIU	R/W
01FF8802-03	ROMCtrl	SIU	R/W
01FF8804-05	CS0/CS5Ctrl	SIU	R/W
01FF8806-07	CS1/2Ctrl	SIU	R/W
01FF8808-09	MCSCtrl	SIU	R/W
01FF880A-0B	FlashCtrl	SIU	R/W
01FF880C-0D	RotPackCtrl	SIU	R/W
01FF880E-0F	CS3/4Ctrl	SIU	R/W
01FF8820-21	DRAMCtrl	DRAM/Flash Memory Controller	R/W
01FF8822-2F	(Not Used)	—	—
01FF8830-31	GPIOConfig	GPIO Block	R/W
01FF8832-33	GPIOData	GPIO Block	R/W
01FF8834-35	GPIODir	GPIO Block	R/W
01FF8836-4F	(Not Used)	—	—
01FF8850-51	SstepClk	Scan/Print Motor Control	R/W
01FF8852-53	VPStepClk	Scan/Print Motor Control	R/W
01FF8854-5F	(Not Used)	—	—
01FF8860-6F	(Not Used)	—	—
01FF8870-71	RotNN	Bit Rotation Block	R/W
01FF8872-73	BRBWarp	Bit Rotation Block	—
01FF8874-75	RotLineLength	Bit Rotation Block	R/W
01FF8876-8F	(Not Used)	—	—
01FF8880-1	125µSprescaler	Fax Timing Block	R/W
01FF8882-3	ICLKPeriod	Fax Timing Block	R/W
01FF8884-5	MSINTPeriod	Fax Timing Block	R/W
01FF8886-7	INTClear	Fax Timing Block	W
01FF8888-8F	(Not Used)	—	—
01FF8890-91	ScanCycle	Video/Scan Controller	R/W
01FF8892-93	ScanConfig	Video/Scan Controller	R/W
01FF8894-95	ScanDotCtrl	Video/Scan Controller	R/W
01FF8896-97	ScanLength	Video/Scan Controller	R/W
01FF8898-99	ScanStartDelay	Video/Scan Controller	R/W
01FF889A-9B	StartEdges	Video/Scan Controller	R/W
01FF889C-9D	StartConfig	Video/Scan Controller	R/W
01FF889E-9F	Clk2aEdges	Video/Scan Controller	R/W
01FF88A0-A1	Clk2bEdges	Video/Scan Controller	R/W
01FF88A2-A3	Clk2cEdges	Video/Scan Controller	R/W
01FF88A4-A5	ADCSampleCfg	Video/Scan Controller	R/W

Table 4-3. Setup Registers (2 of 2)

Address	Register Name	Block Name	R, DR (Dummy Read), W, DW (Dummy Write)
01FF88A6-A7	ClampCtrl	Video/Scan Controller	R/W
01FF88A8-A9	ClampDelay	Video/Scan Controller	R/W
01FF88AA-AB	ClampEdges	Video/Scan Controller	R/W
01FF88AC-AD	LED0Edges	Video/Scan Controller	R/W
01FF88AE-AF	LED1Edges	Video/Scan Controller	R/W
01FF88B0-B1	LED2Edges	Video/Scan Controller	R/W
01FF88B2-B3	LED0PWM	Video/Scan Controller	R/W
01FF88B4-B5	LED1PWM	Video/Scan Controller	R/W
01FF88B6-B7	LED2PWM	Video/Scan Controller	R/W
01FF88B8-B9	LEDConfig	Video/Scan Controller	R/W
01FF88BA-BB	ScanIAConfig	Video/Scan Controller	R/W
01FF88BC-BD	ScanCtrlDelay	Video/Scan Controller	R/W
01FF88BE-BF	ADCConfig	Video/Scan Controller	R/W
01FF88C0-C1	SPIConfig	Video/Scan Controller	R/W
01FF88C2-C3	SPIData	Video/Scan Controller	R/W (Bit[15:8] – R only)
01FF88C4-C5	(Not Used)	—	—
01FF88C6-C7	VidLineCfg	Video/Scan Controller	R/W
01FF88C8-C9	VidLLStat	Video/Scan Controller	R only
01FF88CA-CB	VidOddFLStat	Video/Scan Controller	R only
01FF88CC-CD	VidEvenFLStat	Video/Scan Controller	R only
01FF88CE-CF	(Not Used)	—	—
01FF88D0-DF	(Not Used)	—	—
01FF88E0-FF	Reserved	—	—
01FF88900-FFF	(Not Used)	—	—

4.2   Cache Memory Controller

4.2.1   Functional Description

4.2.1.1   Cache Summary

- 4 kB instruction cache RAM with expansion capability
- Physical address cache access and cache tags
- Two Way Set Associative with LRU algorithm
- 16 bytes cache line size with 128 cache lines in each way
- Supports both ARM and thumb mode instructions
- Cache memory can be enabled or disabled
- Provides lock function and flush function
- Interfaces between ARM7TDMI and SIU (System Interface Unit)

4.2.1.2   Cache Overview

The Cache Controller is an instruction only cache; a level 1 cache for ARM7TDMI. The cache, when enabled, will support zero wait state sequential instruction access from ARM provided the instruction is in the cache and valid (Cache hit). If an instruction is not found in the cache memory (Cache miss), the Cache Controller will activate the LRU (Least Recently Used) replacement algorithm. In this case, the ARM will incur a number of wait states depending on the memory speed.

The 4 kB Cache Memory is divided into two Ways, which means 2 kB per Way. Each Way is further divided into 128 Cache Lines with 16 bytes of instructions in each Line. If a Cache miss is detected and Cache Line fill is required, the Cache Controller will replace the least recently used (LRU) Cache line, the Cache Line fill operation is done in burst (sequential), minimizing the overhead.

The ability for the software to lock the entire Cache or individual line and to flush the entire Cache Memory contents are provided. In addition, the Cache Memory and Cache Tags can be placed in Test Mode for power-up verification and system diagnoses. The entire Cache Memory and Cache Controller can also be disabled allowing the ARM to bypass the Cache Controller unit.
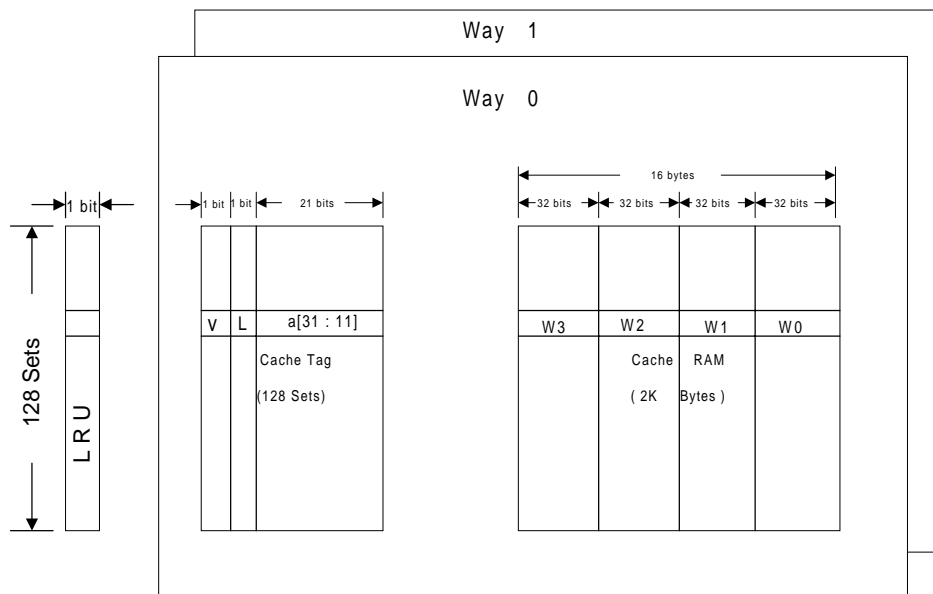
Figure 4-3. MFC2000 Cache Organization

Table 4-4. Cache Tag Data Format (for Test Mode Read/Write Operation)

Cache Tag Data Format	
Bits	**Description**
31	LRU bit, accessible only through Way 0 Tag Read
30:23	Unused Bits
22	Lock
21	Valid
20:0	A[31:11]

4.2.1.3   Cache RAM

The Cache RAM consists of four 512 X 16 Asynchronous Static RAM modules, and they are organized into two 512 words (32 bit/word) to support two way set associative. The memory map for direct accesses while in Test Mode or Lock Mode is as follows:

WAY 0: $01FE0000-$01FE07FF

WAY 1: $01FE0800-$01FE0FFF

4.2.1.4   Cache Tags

The Cache Tags are defined as follows:

Valid (1 bit):   A 1 in this bit indicates that the Tags and data at the addressed Cache Line are both valid. Neither Tags nor data have meaning if this bit is 0. Upon power up, the tag memories will undergo an automatic flush operation that requires 128 clocks. During the flush operation, the cache is disabled.

Lock (1 bit):   A 1 in this bit indicates that the Tags and data at the addressed Cache Line are both valid and locked and should not be replaced when a Cache miss is detected.

LRU (1 bit):   Indicates that the Tags and data at the addressed Cache line (if not locked) at Way 0 can be replaced if this bit is 0. If this bit is a 1, the Cache Line and Tags in Way 1 should be chosen for replacement.

Unused (8 bits): Unused bits are undefined.

A[31:11]:   Address Tag bits which together with Cache address (A[10:4]), uniquely identify a Cache Line in the entire 32-bit physical address space.

The Cache Tags are memory mapped to the following address space (not fully utilized) when in the Test Mode or Lock Mode :

WAY 0: $01FE1000-$01FE17FF

WAY 1: $01FE1800-$01FE1FFF

It should be noted that bits 2-3 of the addresses are not decoded during the Tag entry accesses, i.e., $sa+00, $sa+04, $sa+08, and $sa+0C all access the same Tag entry.

4.2.1.5   Accessing the Cache

To access the Cache during an instruction fetch, the Cache Controller performs the normal cache operation. If accesses are performed during Test mode or Lock mode, the Tag RAM and Cache RAM are treated as regular memories.

If the Cache is enabled, regardless of cache hit or miss, the Cache Controller asserts one wait state for every non-sequential cycle to start the instruction fetch cycle.

If the access results in a hit, the wait state is de-asserted and a 32-bit Cache data is output to the ARM. Subsequent Sequential (S-cycles) access(es) require zero wait state if they are found in the same Cache Line. If the access crosses Cache Line boundary, the Cache Controller will add one wait state for the first S cycle that crosses the boundary, and then add additional wait states if it results in a cache miss.

If the access misses the Cache, the Cache Controller extends the wait states until the corresponding cache data or cache line is received from external memory through SIU. The number of wait states inserted is affected by the status of the lock bit for the corresponding Cache Line. If the Line is not locked, then the Cache Line fill operation will be performed and the required wait state will depend on the speed and the data width of the external devices. If the Cache Line is locked, the Cache Controller will re-generate the ARM cycle and forward the cycle to SIU. The required wait states in this case will be much less than that of Cache Line fill, but still a few cycles more than a simple pass-through operation when the Cache is disabled. This is due to the time required for the tag comparison. It should be noted that, in the case of Cache Line fill, the requested data is not transferred to the ARM until the Cache Line fill operation is completed.

4.2.1.6   Definition of a Cache Hit

There are two requirements for a Cache hit. First the ARM A[31:11] must match the Cache Address Tags accessed by A[10 :4] in an instruction fetch cycle. Second, the addressed Cache Line must be Valid.

4.2.1.7   LRU Algorithm

The LRU (Least Recently Used) algorithm is applied when a Cache miss is detected. This algorithm first looks for a non-valid Line in the Set for a replacement. The order that is used for this checking is Way 0 first, then Way 1. If both lines associated with the Set are valid, then the Lock bit check is followed. If both are not locked, then the LRU bit (one bit only) associated with the Set is tested. If it is a zero, the Cache Line in Way 0 is replaced; otherwise, Way 1. If both are locked, no replacement can be performed and the Cache Controller will convert the cycle from instruction fetch to data fetch and forward the cycle to SIU for the requested instruction. If only one of the two lines is locked, the unlocked line will be chosen for the replacement.

4.2.1.8   SIU interface

The ARM to SIU interface behaves differently depending on whether the Cache is enabled or not. If the Cache is disabled, the only affect that the Cache Controller adds to the ARM/SIU interface is a small propagation delay for those signals that pass through the Cache Controller (refer to the block diagram for the pass-through signals). On the other hand, if the Cache is enabled, the Cache controller will response to an instruction cycle by either providing data to the ARM in a cache hit, or, converting the instruction cycle to a series of burst data cycle(s) and forwarding them to SIU in a cache miss.

4.2.2   Register Description

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Cache Control Register **$01FF8800**	N/A	Flush Cache	Global Lock	Lock Mode	Test Mode	Cache Enable	N/A	N/A	Rst Value 00h

This register resides in the SIU block. The SIU, upon detecting the set condition for a given bit(s) in this register, asserts the corresponding control signal(s) to the Cache controller.

Bit 7   **Reserved**

Bit 6   **Flush Cache**   Read/writable bit. Writing a 1 to this bit flushes all Valid bits, LRU bits, and Lock bits in the Cache Tag to zero. It requires 128 cycles to flush the entire Tag memory area. This bit will be automatically reset upon completion of the flush operation. Flush does not reset Global Lock or Cache Enable condition if present.

Bit 5   **Global Lock**   Read Writable bit. Writing a 1 to this bit locks the entire Cache memory; a 0 unlocks the Cache. This bit provides a quick way to lock the entire cache memory.

Bit 4   **Lock Mode**   Read/writable bit. Writing a 1 to this bit and a 0 to the Cache Enable bit places the Cache in the Lock Mode. The Cache stays in Lock Mode until a 0 is written to this bit.

Bit 3   **Test Mode**   Read/writable bit. Writing a 1 to this bit and a 0 to the Cache Enable bit sets the Cache into test mode. In test mode, the Cache RAMs and Tags can be accessed in the same manner as regular memory. Certain Tag bits are readable only. The Cache stays in Test Mode until a 0 is written to this bit.

Bit 2   **Cache Enable**   Read/writable bit. Writing a 1 enables the Cache and the Cache stays enabled until a 0 is written or a reset is received. Power-up resets to 0 and disables the Cache.

Bit 1   **Reserved**

Bit 0   **Reserved**

4.2.3   Firmware Operation

4.2.3.1   Enabling the Cache

The Cache Enable bit in the SIU Cache Control register determines if the Cache is enabled or not. In power-up reset state, the Cache is disabled. If disabled, all CPU accesses go directly to the SIU and the Cache Controller passes through all signals from ARM to SIU. After the power-up reset, all Cache Tag entries are flushed after 128 clocks. If the Cache is enabled after power-up, the Cache Controller starts to update the Cache memory and Cache tag after the flush operation is completed.

4.2.3.2   Locking the Cache

The system can lock the entire cache memory by setting GLOBAL LOCK bit to 1 in Cache Control register. The Cache remains locked until the bit is reset. Setting and resetting the Global _Lock bit has no effect on the individual lock bit set during the Lock mode, the individual lock bit can be cleared by setting Flush_Cache bit to 1.

The system can also lock an individual Cache Line by placing the Cache in the Lock Mode. Once in the Lock mode, the system can access the Cache RAM and Tag RAM as if they were regular memory. A write to the Tag entry sets both the Lock bit and Valid bit for the corresponding Tag. The software is responsible for properly mapping the instructions from ROM ('where to be cached in') into Cache RAM and Tag RAM ('where to be locked down') based on the modular of 2048 bytes. In other words, the A10-A2 of address lines used for the ROM code and Cache/Tag RAM's entry must be identical, and the A31-A11used for the ROM code becomes the data entry for the corresponding Tag entry. Caching is disabled during lock mode; the system must exit the lock mode before enabling the Cache.

Once a Cache line is locked, LRU replacement policy prevents the replacement of the locked Cache Line. If both Cache Lines in a Set (two Ways) are locked, the LRU algorithm is not able to replace either Line; thus, no Cache Line fill is performed; instead, a data fetch N (non-sequential) cycle is generated by the Cache Controller and sent to the SIU. A minimum of 5 wait states is expected.

4.2.3.3   Flushing the Cache

The system can clear the Cache by activating the Flush input. This signal is generated by the SIU when the Flush bit in the Cache Control register is set by the system. Upon receiving the Flush input, the Cache Controller starts the flush operation. The operation takes 128 clocks to resets all the Tag Valid bits, LRU bit, and Lock bit. During the operation the cache, if enabled, is temporarily disabled until the flush is completed. The Flush bit is cleared automatically at the end of the flush operation.

4.2.3.4   Testing the Cache and Tag Memories

The system can access the Cache memory and Tag RAM as regular memory does when in Test mode. Test mode is entered after setting the Test bit in the Cache Control register. In Test mode, the Tag RAM and Cache RAM behave like an ordinary memory for read/write cycles. This test feature is not only required for the power-up self test, but also is important for diagnostics when a system problem develops. The contents of the TAG and Cache RAM are essential to the investigation of the problem.

> *Note*:   *The Valid bit, Lock bit, and LRU bit can only be read, not written, and LRU is only available through Way 0 access.*

> *Note*:   *It is important to flush the Cache upon completion of the Test Mode.*

4.3   SIU

4.3.1   Functional Description

The System Integration Unit (SIU) is responsible for interfacing between the ARM7TDMI core, the Cache Memory Controller, the Internal Peripheral Bus (IPB), and the External Bus (EB). The ARM7TDMI core and the Cache Memory Controller are on the Internal System Bus (ISB). The ISB data bus is 32-bits wide and the IPB and EB data buses are 16-bits wide. The SIU generates the external chip selects along with chip selects to all the internal peripherals. It provides the following functions:

1.   Interfaces between the Internal Peripheral Bus (IPB), the Internal System Bus (ISB) and the External Bus (EB). The SIU allows bus master devices on the IPB and ISB.

2.   Control the chip selects to devices on the IPB, the ISB, and the EB.

3.   Address multiplexing for DRAM access.

4.   ROM interleave control (including wait state control for the interleave mode): no interleave and 2-way interleave with external Q-switch.

5.   Fast page mode ROM operation.

6.   Even though ARM7TDMI is fixed to the little endian in this MFC2000 chip, the SIU can support the little endian or big endian for the DMA operation.

7.   Support Arm and Thumb mode operations.

4.3.1.1   IPB, ISB and External Bus

IPB Bus

The IPB Bus supports both 8-bit and 16-bit peripherals. The ARM or an internal bus master such as DMA can access a device residing on the IPB bus.

The SIU provides the chip selects to each of the internal peripheral devices. The chip selects are driven in the second clock cycle of an IPB bus cycle. The peripheral device needs to decode only the address lines required to access the specific registers within the block.

Transactions on the IPB bus only occur when a device on the IPB bus is being accessed. All accesses on the IPB bus require two peripheral bus clock cycles (2 SIUCLK's). During the first cycle, the address is decoded and determined if an access to an internal peripheral is occurring. During the second cycle the peripheral chip select is asserted, and the access occurs.

During Write operations to peripherals, signals BS[1:0] are used to signal which bytes are valid on the data bus. 8-bit peripherals can ignored these signals. 16-bit peripherals MUST use these signals to allow each 8-bit half of the peripheral registers to be written independently. This is due to the fact that the ARM compiler may generate two byte transactions when accessing a 16-bite register on the IPB instead of a single halfword transaction.

During Read operations, the peripherals must fill the 16-bit IPB data bus. If the peripheral is less than 16-bit wide, it should fill the empty bits with 0.

ISB Bus

The ISB Bus is used for connecting ARM and Cache Controller to the highest performance. The 32-bit ISB bus directly interfaces with the ARM core 32-bit data bus. The cache memory controller resides on this bus.

All control signals to and from the ARM, and its address bus go through the cache memory controller regardless of the cache enable bit. The cache controller control register resides in the SIU.

When the ARM is fetching instructions, and it is a cache hit situation, the ARM gets the instructions from the cache. The SIU lets the other bus masters have the bus.

When the ARM is fetching instructions, and it is a cache miss situation, the SIU must perform a burst read of eight halfwords (4 words) to fill up the cache line if the cache line is not locked. If the cache line is locked, then the SIU reads in two halfwords (one word) of instruction.

When the ARM is fetching data, data is passed directly from the SIU to the ARM.

EB Bus

The EB Bus is used for connecting external memories and devices. The width of the selected slave device is programmable in the chip select configuration register. The external A[23:12] and A[11:0] addresses are multiplexed through A[11:0] pins. The ALE signal is provided to latch A[23:12] addresses externally.

External Chip Selects

The SIU provides programmable external chip selects. Each chip select is programmable through the chip select configuration register.

- Each chip select can be configured to be:
- enabled or disabled (default = enabled).
- programmable from 0 up to 7 wait states.
- programmable read/write delay-on (write strobe activated one or two SIUCLK cycles later).
- programmable write early-off (read or write strobe deactivated one SIUCLK cycle earlier).
- programmable to allow for 8 or 16 bit devices. The SIU will automatically perform the necessary transaction to access any size data as long as the source of the transaction is internal.

> **Note:**   The RD/WR-delay-on and WR-early-off settings should be disabled for zero wait state access. For other wait state settings ($\geq$ 1wait state), the delay-on and early-off will shorten the width of read/write strobe. Firmware has the responsibility to set RD/WR-delay-on and WR-early-off bits correctly. Otherwise, read or write strobes may disappear. For example, if 1 wait state and 1 RD/WR-delay-on are set for a chip select, the read strobe is not suppressed when firmware tries to do a read operation. If 2 wait states, 1 WR-early-off and 1 RD/WR-delay-on are set for a chip select, the write strobe is not suppressed when firmware try to do a write operation.

ROM Interface

The SIU supports non-interleave, 2-way interleave and fast page mode access to ROM, depending on the ROM Access Configuration pins (AE[2]/ROM_CFG[0] and AO[2]/ROM_CFG[1] pins). Following are the four configurations supported by the SIU.

ROM Access Configuration[1:0]	ROM Mode
00	8-bit non-interleave
01	16-bit non-interleave
10	16-bit 2-way interleave
11	16-bit fast page mode

The MFC2000 assigns 4 multi-function pins (AE[2], AO[2], AE[3], and AO[3]) to facilitate the interleave access. SIU generates 4 signals on these 4 pins to control ROMs for the following types of interleave accesses.

- In 2-way interleave mode, MFC2000 pins AE[2], AO[2], AE[3], and AO[3] are connected to pin A[1:0] of the even and odd external ROMs. MFC2000 pins A[1:0] are used to enable the ROM's data busses.
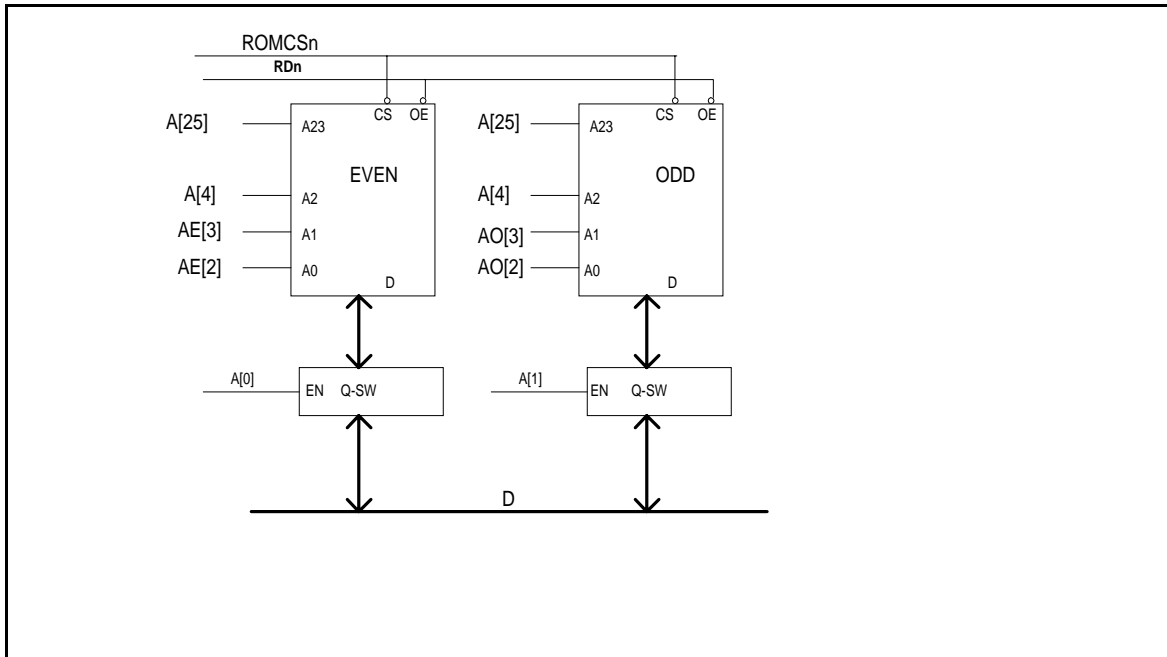
Figure 4-4. 2-Way Interleave ROM Connection

- The ROMCSn can be programmed to have up to 7 wait states for the initial access, and up to 1 wait state for the sub-sequential access.
- The write access to the ROM chip select area (ROMCSn) is allowed. It is customer's choice to use it or not.
- To use NOR-type flash memory in the ROM chip select area, WREn and WROn are designed to be used as the write strobes for the different banks in the interleave mode (not for the higher and lower bytes). Therefore, the 16-bit wide flash memory should be used in order to avoid the extra glue logic.
- For non-interleave mode flash memory in the ROM chip select area, the WROn is used to access the higher byte of the 16-bit data bus and the WREn is used to access the lower byte of the 16-bit data bus.

Assume that W wait states are needed for the initial access to ROM and S wait states (S = 0 or 1) for the sub-sequential access according to the CPU clock frequency and the ROM speed. For a burst of 8 half-words interleave access, the wait state of each half-word access, assuming the starting address's A[3:1]=000. We can have the following table show all different access modes for reading from ROM.

Table 4-5. Access Modes for Reading ROM

ROM Access Mode	Data Type	Cache Memory	Wait States	Notes
8-bit non-interleave	instruction	Cache enabled and Cache miss	w,w,w,w,w,w,w,w w,w,w,w,w,w,w,w (16 sequential accesses)	This is cache burst access. Save the address decoding cycle for all accesses except the first access.
16-bit non-interleave	instruction	Cache enabled and Cache miss	w,w,w,w,w,w,w,w (8 sequential accesses)	This is cache burst access. Save the address decoding cycle for all accesses except the first access.
8-bit non-interleave	instruction	Cache disabled	w	If the sequential access occurs, save the address decoding cycle for all accesses except the first access.
16-bit non-interleave	instruction	Cache disabled	w	If the sequential access occurs, save the address decoding cycle for all accesses except the first access.
8-bit non-interleave	data	Not applied	w	If the sequential access occurs, save the address decoding cycle for all accesses except the first access.
16-bit non-interleave	data	Not applied	w	If the sequential access occurs, save the address decoding cycle for all accesses except the first access.
16-bit 2-way interleave	instruction	Cache enabled and Cache miss	w, s, w-s-1, s, w-s-1, s, w-s-1, s (8 sequential accesses)	This is cache burst access. 0 or 1 wait state is programmable and depends on the CPU clock frequency and the ROM speed. If 'w-s-1' is less than 1, it will be forced to 1.
16-bit 2-way interleave	instruction	Cache disabled	w, s, w-s-1, s, w-s-1, s, w-s-1, s (one interleave access sequence) The actual sequential access length is dynamic.	0 or 1 wait state is programmable and depends on the CPU clock frequency and the ROM speed. If the starting address of the sequential access is not lined up with the octal address boundary of the interleave access, the partial interleave access sequence should be done. Then, restart the interleave access sequence at the octal address boundary of the interleave access. Even if the stopping address of the sequential access is not lined up with the octal address boundary of the interleave access, the access sequence must be stopped immediately at anywhere. If 'w-s-1' is less than s, it will be forced to s.
16-bit 2-way interleave	data	Cache enabled and Cache miss	w, s, w-s-1, s, w-s-1, s, w-s-1, s (one interleave access sequence) The actual sequential access length is dynamic.	0 or 1 wait state is programmable and depends on the CPU clock frequency and the ROM speed. If the starting address of the sequential access is not lined up with the octal address boundary of the interleave access, the partial interleave access sequence should be done. Then, restart the interleave access sequence at the octal address boundary of the interleave access. Even if the stopping address of the sequential access is not lined up with the address boundary of the interleave access, the access sequence must be stopped immediately at anywhere. If 'w-s-1' is less than s, it will be forced to s.

Conexant

If the ROM write access (Flash memory in the ROM address range) is performed in the interleave access mode, the SIU still generates those signals to control ROMs and external multiplexes to perform the interleave access. But, all the access (no matter if it is the sequential access or not) have W wait states.

External DRAM Interface

When the decoded address from any bus master matches external DRAM address, the SIU will issue a DRAM request to the DRAM controller and start the transaction . First, it routes the DRAM row address to the address bus. After receiving the column enable signal from the DRAM controller, the SIU multiplexes the DRAM column address to the same address bus according to the size of the DRAM . The SIU will perform the next transaction after receiving the DRAM ready back from the DRAM controller signaling the DRAM transaction is complete.

The SIU also looks at the burst request signals from the bus master who owns the bus to generate the BURST signal to the DRAM Controller indicating a burst access.

Bus Arbitration

The bus arbitrator block arbitrates control of the internal and external busses between the ARM7TDMI core and any bus master devices (such as DMA) residing on the IPB or ISB. The ARM core is the default bus master and has control of the bus whenever no other bus master requests it . In arbitrating control of the bus, the arbitrator gives highest priority to the DMA Controller and then, the ARM core.

In burst mode access (both DMA and CPU), the bus is not arbitrated within the burst access. In order to prevent a bus master from hogging the bus. The maximum burst length allowed is eight halfword access. The DMA of internal peripherals only bursts a maximum of five halfwords.

A bus master requests the bus by asserting request. The arbitrator grants the bus to the requesting bus master by asserting grant. The requesting bus master must continue to assert request for as long a bus ownership is required and release the bus by de-asserting request. The arbitrator always inserts a single dead cycle before granting the bus to another bus master.

Little Endian and Big Endian

The little endian and big endian control is only for internal DMA (The DMA request is from an internal peripheral). When a DMA access requires different endian format. the corresponding bit of the DMAEndian register needs to be set. The SIU will transform the endian format; from little endian DMA address and data into big endian format or from big endian DMA address and data into little endian format. The even and odd write signals (WREn, WROn) also change accordingly. The following tables show the final addresses and data at the ASIC pins, and the resulting DMA read or write data. Internal DMA data size is always 16 bits (a halfword).

Table 4-6. Read Operation (Internal Peripheral Gets Data From Memory)

BIG ENDIAN				LITTLE ENDIAN	
DMA SIZE	MEM SIZE	DMA_AD	DMA DATA (output)	MEM ADDR	MEM DATA (input)
Half Word	Byte	0000	Byte 2, Byte 3	0011,0010	Byte 3, Byte 2
Half Word	Byte	0010	Byte 0, Byte 1	0001,0000	Byte 1, Byte 0
Half Word	Half Word	0000	Byte 2, Byte 3	0010	Byte 3, Byte 2
Half Word	Half Word	0010	Byte 0, Byte 1	0000	Byte 1, Byte 0

Table 4-7. Write Operation (Internal Peripheral Puts Data Into Memory)

LITTLE ENDIAN				BIG ENDIAN			
DMA SIZE	MEM SIZE	DMA_AD	DMA DATA (input)	MEM ADDR	MEM DATA (output)	WRON	WREN
Half Word	Byte	0000	B1,B0	0011,0010	B0,B1	0	0
Half Word	Byte	0010	B3,B2	0001,0000	B2,B3	0	0
Half Word	Half Word	0000	B1,B0	0010	B0,B1	0	0
Half Word	Half Word	0010	B3,B2	0000	B2,B3	0	0

4.3.2   Register Description

SIU Control

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
SIU Configuration (SIUConfig) **01FF8801**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	CS4n Read Only	Rst. Value xxxxxxx0b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
SIU Configuration (SIUConfig) **01FF8800**	CS3n Write Only	Flush Cache	Global Lock	Cache Lock Mode	Cache Test Mode	Cache Enable	Disable Force External	Disable Abort	Rst. Value 00h Read Value 00h

Bit 8   CS4n Read only   Writing a 1 makes CS4n a read only CS. Default is 0.

Bit 7   CS3n Write only   Writing a 1 makes CS3n a write only CS. Default is 0.

Bit 6   Flush: Write only bit.   Writing a 1 generates a pulse which flushes all Valid bits, LRU bits and Lock bits in the Cache Tag. Default is 0

Bit 5   Global_Lock: Read/writable bit   Writing a 1 locks the whole Cache. The Cache stays in Lock Mode until a 0 is written. Default is 0.

Bit 4   Cache_Lock: Read/writable bit   Writing a 1 places the Cache in the Lock Mode and the Cache stays in Lock Mode until a 0 is written. Each cache line is locked individually . Default is 0

Bit 3   Cache_Test: Read/writable bit   Writing a 1 sets the Cache into Test Mode and the Cache RAM and Tags can be accesses as regular memory. Note that certain Tag bit only readable . The Cache stays in Test Mode until a 0 is written to this bit. Default is 0.

Bit 2   Cache_Enable: Read/writable bit

Writing a 1 enables the Cache and the Cache stays enabled until a 0 is written or a reset is received. Power-up resets to 0, so the Cache is disabled.

Bit 1   Force_external   This signal will disable the forcing of all accesses to be visible on the external bus regardless of destination. If this signal is enabled, only external transactions will be visible on the external bus. 1 is disabled, 0 is enabled. Default is 0.

Bit 0   Disable_abort: This signal disables abort generation for internal and external access. If this signal is a 1, all transactions to internal and external address space will be allowed to occur regardless of if an valid internal or external peripheral exists. If this signal is 0, then accesses must be to valid peripheral locations or an abort signal will be generated. Default is 0.

External Chip Select Control Registers

Associated with each external chip select pin is a register to control automatic functions that will be executed when a location within the chip select range is accessed. All bits default to 0 unless otherwise specified.

Several control functions are common between the various chip select register.

A chip select is enable when the enable bit is set to 1.

Wait states defines the number of wait states that are added to the associated bus cycle, in increments of SIUCLK cycle. A bus cycle is 1 SIUCLK.

Size is the width of the external devices peripherals using the chip select. The allowable widths are 8 and 16 bits. Size are coded : 0=byte, 1 = half-word. If the size of the data is larger than the size of the peripheral, the SIU will automatically perform multiples accesses to complete the transaction. Default to 0.

Where applicable, Strobe Delay On = 1 delays the activation of RDn or WRn by 1 clk. Write Early Off deactivates the WRn strobe by 1 clk earlier.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
ROMCS Control (ROMCtrl) **01FF8803**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
ROMCS Control (ROMCtrl) **01FF8802**	Read/Write Strobe Delay On	Mode[1] (read only)	Mode[0] (read only)	Subsequent Wait	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 1??11111b Read Value 1??11111b

Bit 7   Read/Write strobe Delay On   (default = 1)

Bits 6-5   Mode[1:0] ROM interface mode (read only). These 2 bits are read in directly from 2 pins (the AE[2]/ROM_CFG[0] pin and the AO[2]/ROM_CFG[1] pin during reset).

00: 8-bit Non Interleave
01: 16-bit Non interleave
10: 16-bit 2way interleave
11: 16-bit fast page mode

Bit 4   Subsequent access wait states in interleave mode (default = 1)

Bits 3-1   Wait states or initial access wait states in interleave mode (default = 7)

Bit 0   ROMCSn Enable.   (default = 1)

> *Note:*   *These controls are also applicable to the optional CS5n.*

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
CS5 Control (CS5Ctrl) **01FF8805**	Write Strobe Early Off	Read/Write Strobe Delay On[1]	Read/Write Strobe Delay On[0]	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
CS0 Control (CS0Ctrl) **01FF8804**	Write Strobe Early Off	Read/Write Strobe Delay On	(Not Used)	Size	Wait[2]	Wait[1]	Wait[0]	0	Rst. Value 00x00000b Read Value 00h

Bit 7,15   Write Strobe Early Off   (default =0)

Bit 6,14-13 Read/Write Strobe Delay On   (default = 0)

Bit 5   (Not used)

Bit 4,12   Size   (default = 0 . Byte)

                                            Mode = 0: 8-bit access

                                            1: 16-bit access

Bit 3-1, 11-9 Wait states   (default =0)

Bit 8   CS5n Enable

> **Note:**   The enable control for CS0 is set by the Battery Control Logic.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
CS2 Control (CS2Ctrl) **01FF8807**	Write Strobe Early Off	Read/Write Strobe Delay On[1]	Read/Write Strobe Delay On[0]	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 01h Read Value 01h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
CS1 Control (CS1Ctrl) **01FF8806**	Write Strobe Early Off	Read/Write Strobe Delay On[1]	Read/Write Strobe Delay On[0]	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 01h Read Value 01h

Bit 15   Write Early Off strobe   (default =0)

Bit[14:13] Read/Write strobe Delay On   (default = 00)

Bit 12   Size   (default = 0. Byte)

                                            0: 8-bit access

                                            1: 16-bit access

Bit[11:9]   Wait states   (default =0)

Bit 8   CS2n Enable   (default = 1) .

Bit 7   Write Early Off strobe   (default =0)

Bit[6:5]   Read/Write strobe Delay On   (default = 00)

Bit 4   Size   (default = 0. Byte)
                                               0: 8-bit access
                                               1: 16-bit access

Bit[3:1]   Wait states   (default =0)

Bit 0   CS1n Enable   (default = 1)

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
CS4 Control (CS4Ctrl) **01FF880F**	Write Strobe Early Off	Read/Write Strobe Delay On[1]	Read/Write Strobe Delay On[0]	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 01h Read Value 01h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
CS3 Control (CS3Ctrl) **01FF880E**	Write Strobe Early Off	Read/Write Strobe Delay On[1]	Read/Write Strobe Delay On[0]	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 01h Read Value 01h

Bit 15   Write Early Off strobe   (default =0)

Bit[14:13]   Read/Write strobe Delay On   (default = 00)

Bit 12   Size   (default = 0. Byte)
                                               0: 8-bit access
                                               1: 16-bit access

Bit[11:9]   Wait states   (default =0)

Bit 8   CS4n Enable   (default = 1)

Bit 7   Write Early Off strobe   (default =0)

Bit[6:5]   Read/Write strobe Delay On   (default = 00)

Bit 4   Size   (default = 0. Byte)
                                               0: 8-bit access
                                               1: 16-bit access

Bit[3:1]   Wait states   (default =0)

Bit 0   CS3n Enable   (default = 1)

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Modem CS Control (MCSCtrl) **01FF8809**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Modem Interrupt Select	Rst. Value xxxxxxx0b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Modem CS Control (MCSCtrl) **01FF8808**	Write Strobe Early Off	Read/Write Strobe Delay On	(Not Used)	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 00x00001b Read Value 01h

Bit 7   Select P80 or external MIRQn interrupt (default =0, select P80)

Bit 7   Write Early Off strobe   (default =0)

Bit 6   Read/Write strobe Delay On   (default = 00)

Bit 5   (Not Used)

Bit 4   Size   (default = 0 . Byte)
                                              0: 8-bit access
                                            1: 16-bit access

Bit 3,2&1   Wait states (default =0)

Bit 0   MCSn Enable. (default = 1)

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Flash Memory Control *(FlashCtrl)* **01FF880B**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	FCS1n value for NAND type	FCS0n value for NAND type	FCS1n disable	Rst. Value x8h Read Value 08h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Flash Memory Control *(FlashCtrl)* **01FF880A**	Write Strobe Early Off	FCSn NAND-type	(Not Used)	Size	Wait[2]	Wait[1]	Wait[0]	FCS0n disable	Rst. Value 00x00000b Read Value 00h

Bit 10   Output value of FCS1n when NAND-type memory is used. Not applicable for NOR-type memory.

Bit 9   Output value of FCS0n when NAND-type memory is used. Not applicable for NOR-type memory.

Bit 8   1= Disable FCS1n   (default = 0: Enable)
                                              When this bit is set to 1, pin PWM2/FCS1n is used as PWM2.

Bit 7   Write Early Off strobe   (default =0)

Bit 6   NAND-type memory is used when this bit is set to 1. (default =0 . NOR type).

Bit 4   Size   (default = 0 . Byte)
                                              0: 8-bit access
                                            1: 16-bit access

Bit[3:1]   Wait states   (default =0)

Bit 0   1= Disable FCS0n   (default = 0: Enable)
                                              When this bit is set to 1, pin PWM0/FCS0n is used as PWM0.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
RotPacked Data register Access Control (RotPackCtrl) **01FF880D**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
RotPacked Data register Access Control (RotPackCtrl) **01FF880C**	Write Strobe Early Off	Read/Write Strobe Delay On[1]	Read/Write Strobe Delay On[0]	Size	Wait[2]	Wait[1]	Wait[0]	Enable	Rst. Value 01h Read Value 01h

Bit 7   Write Early Off strobe (default =0)

Bit[6:5]   Read/Write strobe Delay On (default = 00)

Bit 4   Size (default = 0. Byte)

                    0: 8-bit access
                    1: 16-bit access

Bit[3:1]   Wait states (default =0)

Bit 0   RotPackedData register access enable. (default = 1)

> *Note: This register is used to set up the access timing for the DMA read from RotPackedData register to the external PIF device. It provides the control of wait states and RDn width when accessing the RotPackedData.*

4.3.3   Timing
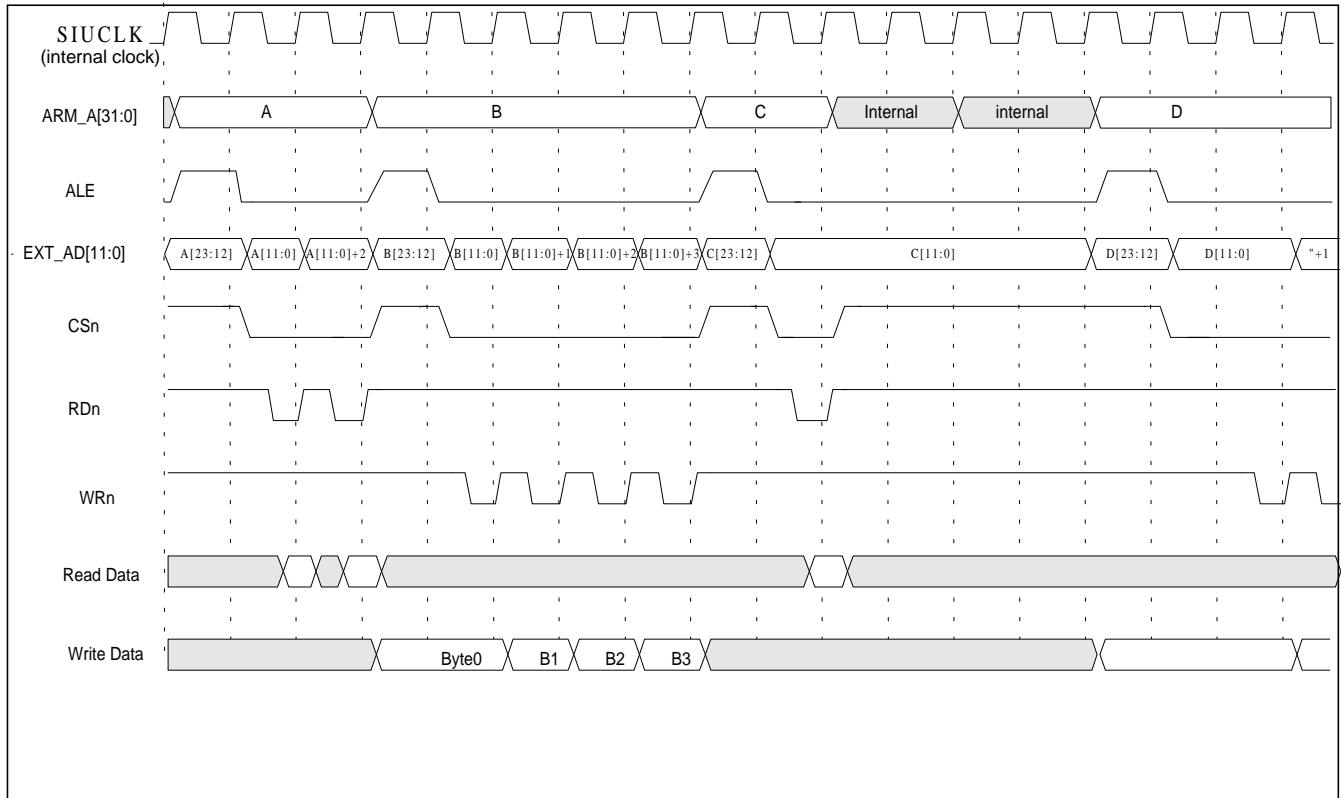
Figure 4-5. Zero Wait State, Single Access, Normal Read, Normal Write
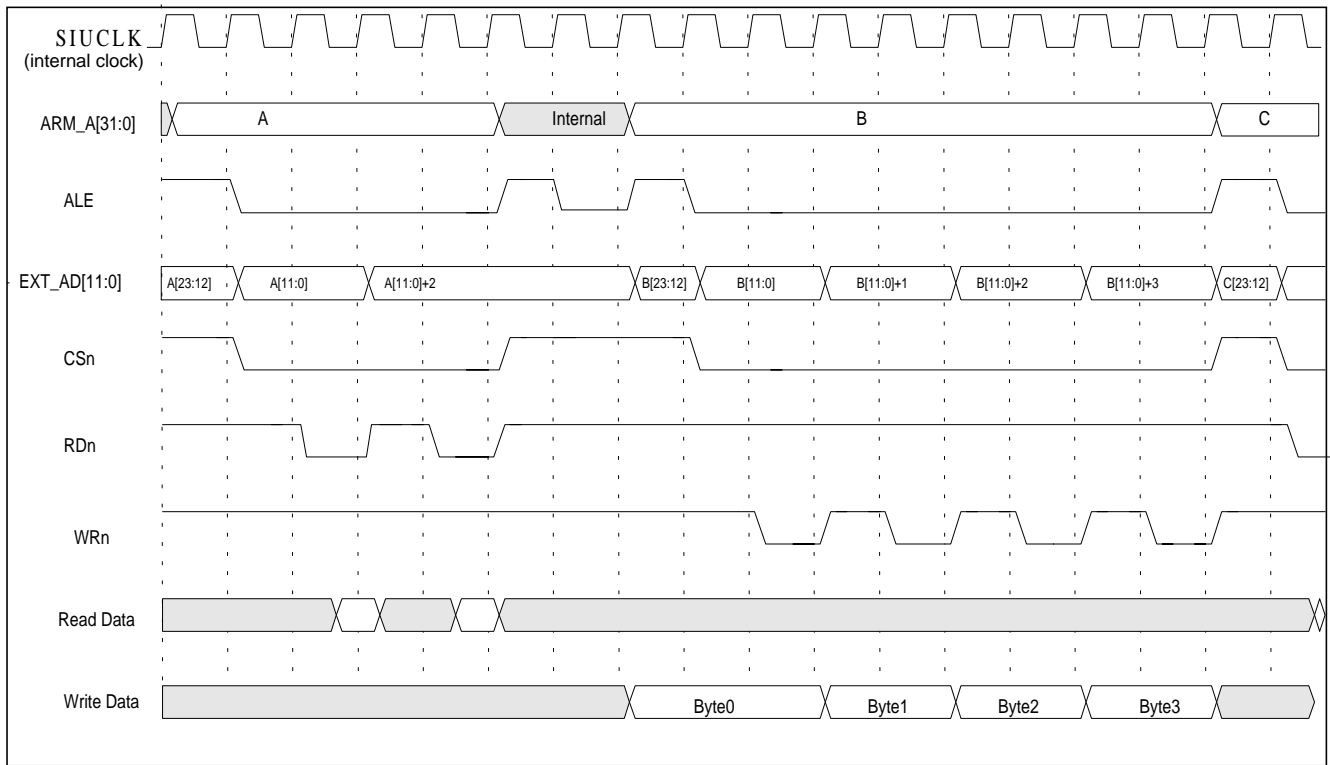
Figure 4-6. One Wait State, Single Access, One Read, One Write
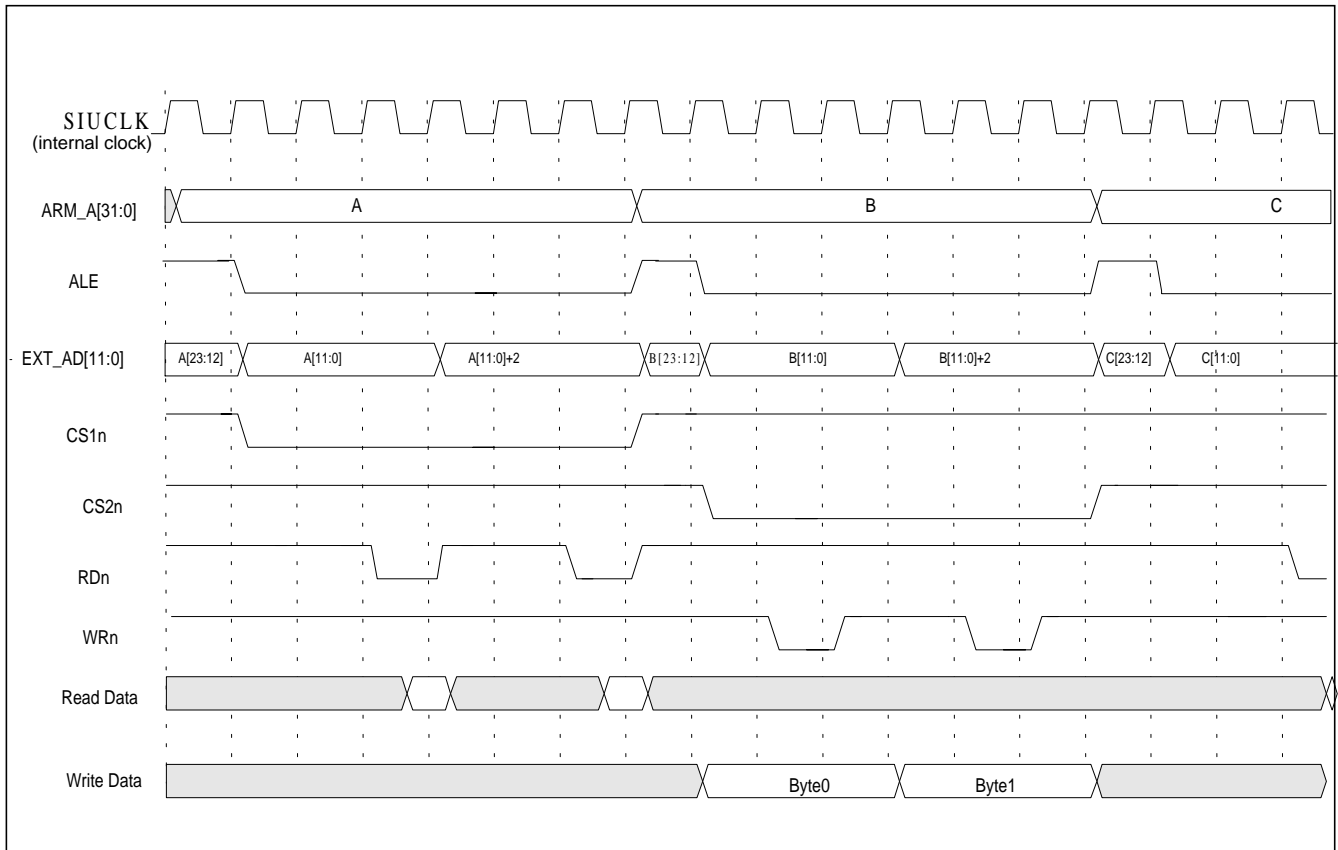
Figure 4-7. Two Wait States, Single Access, Read On Delayed (CS1n), Write Early Off (CS2n)
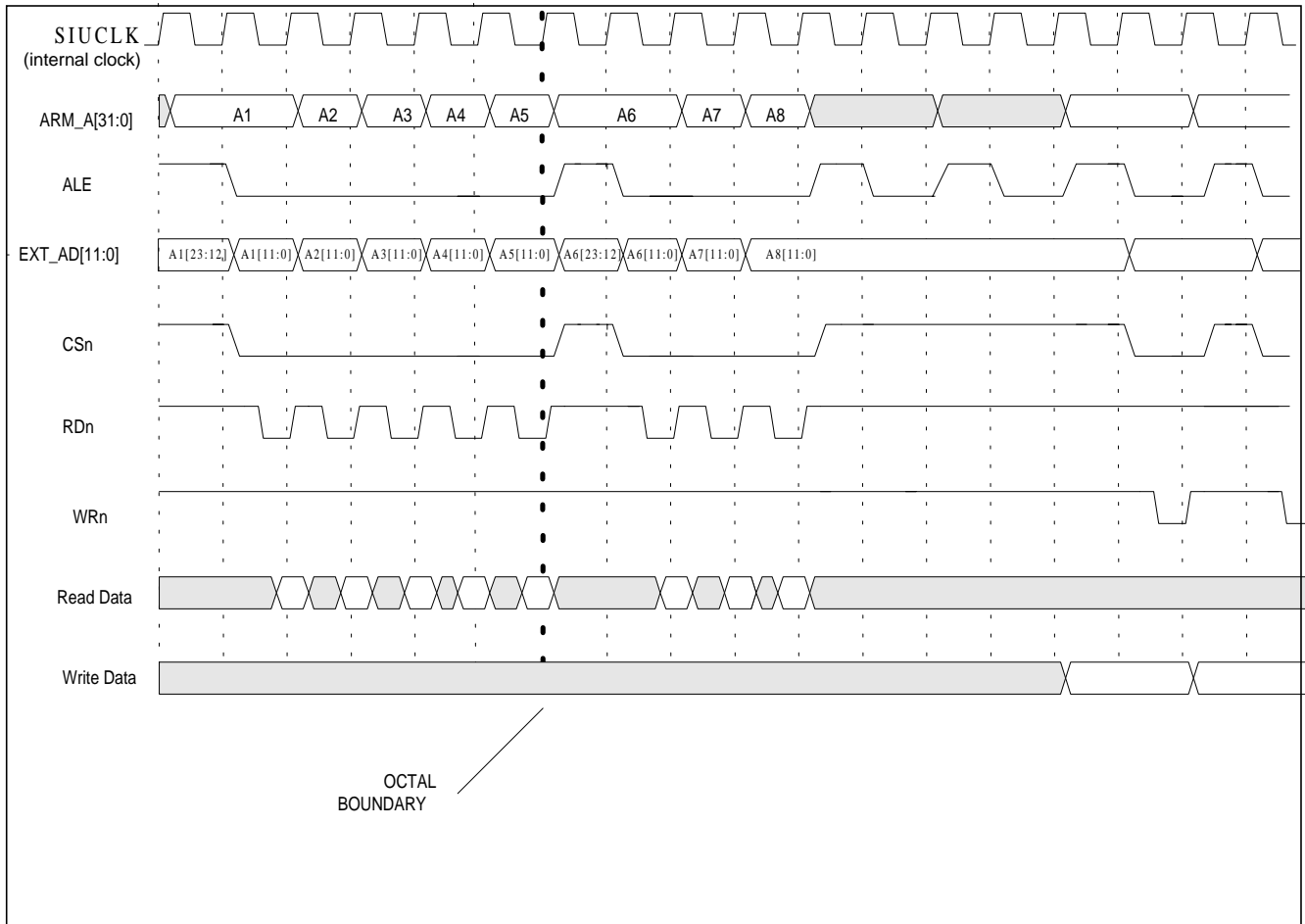
Figure 4-8. Zero Wait State, Burst Access, Normal Read, Normal Write
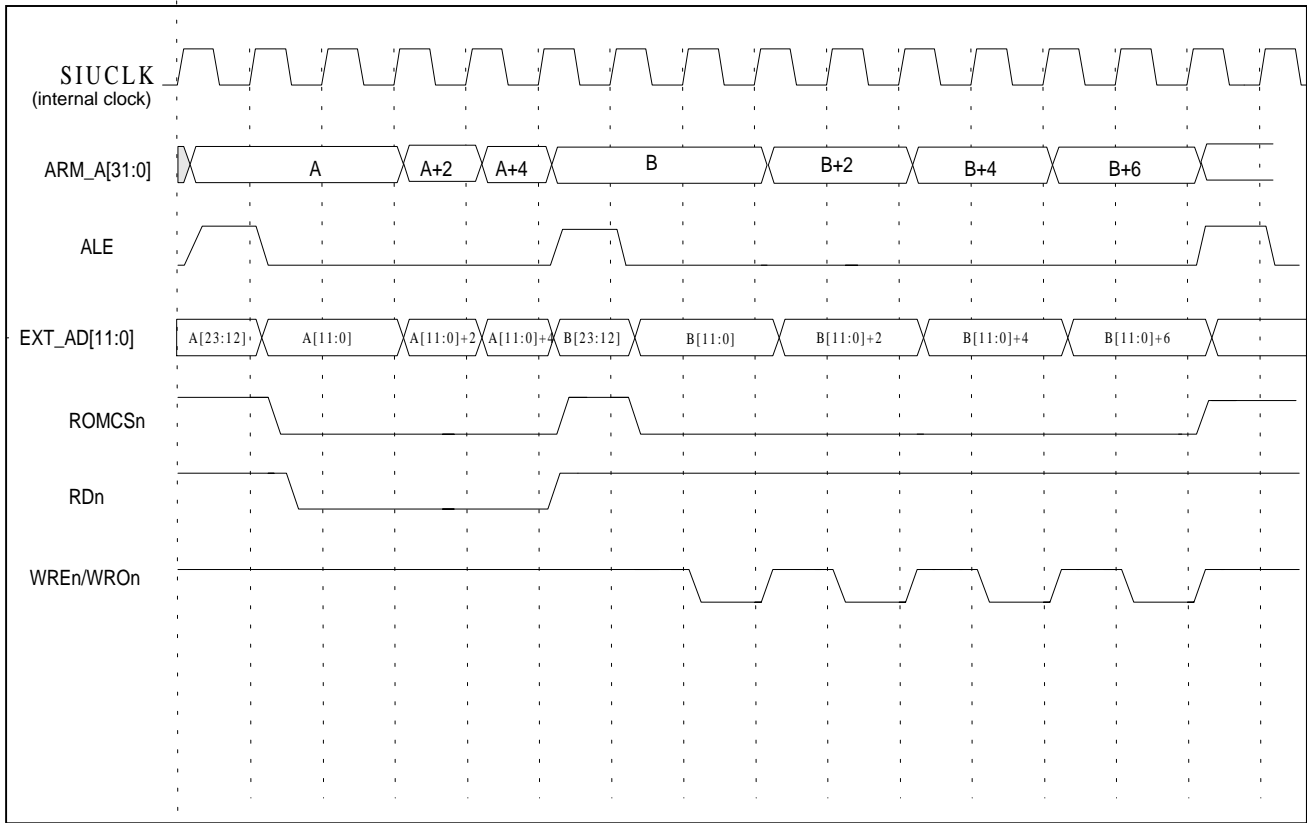
Figure 4-9. Fast Page Mode ROM Access—1,0,0 Read Access Followed by 1,1,1,1, Write Access

Detail External Bus Timing

3 wait states

SIUCLK
(internal clock)

A[11:0]   upper address   lower address

t_{AD} t_{AAD} t_{AAH}

ALE

t_{ALD}

Ext. CS's (romcsn, gpio21(mcsn),gpio7-4(cs5,2n), cs1n,cs0n)

t_{CSD}

RDn

t_{RD} t_{RD}

WREn, WROn

t_{WD} t_{WD} t_{WD} t_{WD}

GPIO[5] (CS3n), GPIO[6] (CS4n)

t_{CGD}

D[15:0] (read)

t_{DIS} t_{DIH}

D[15:0] (write)

t_{DOD} t_{DOH}
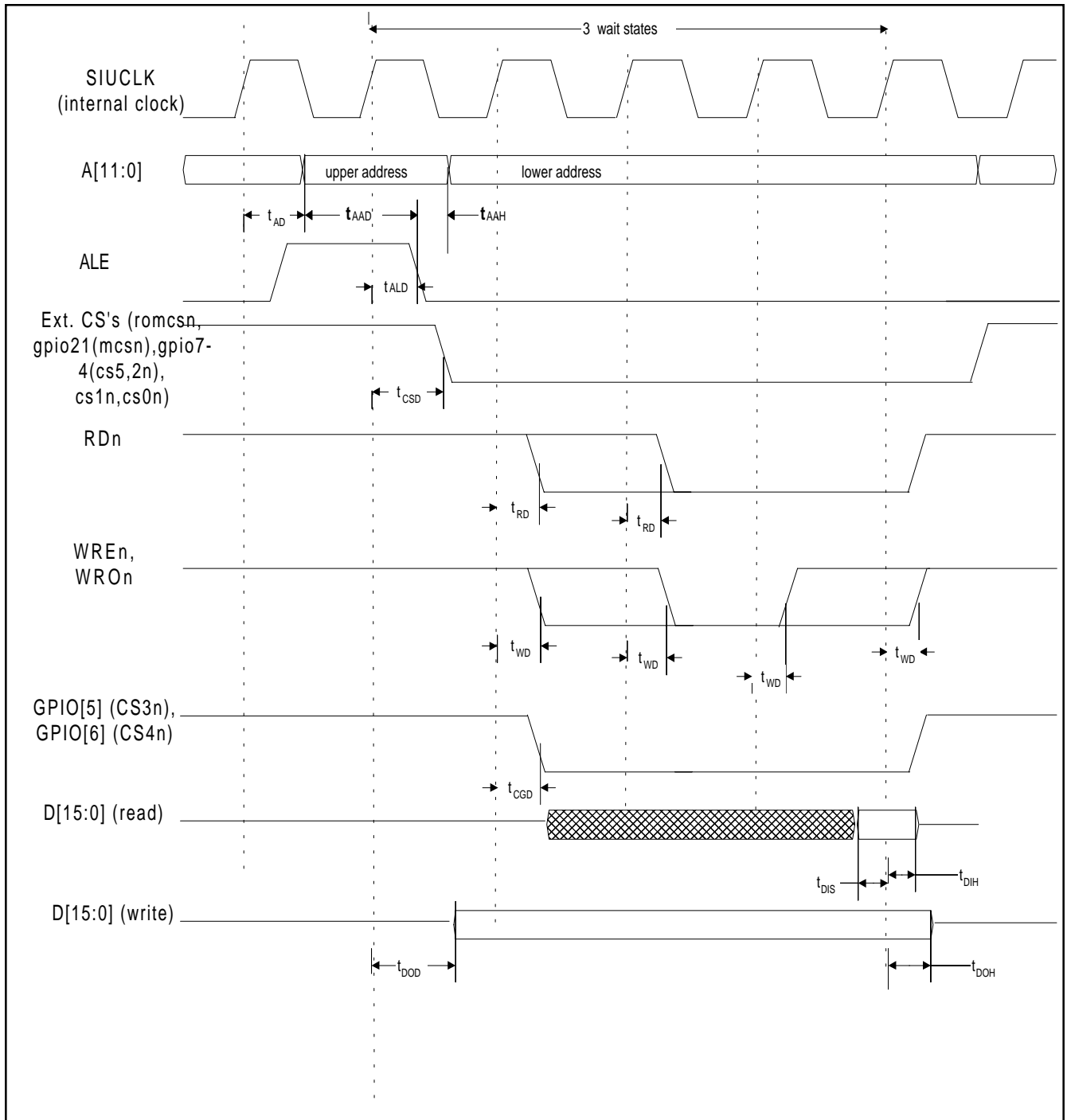
Figure 4-10. System Bus Timing—Read/Write with Wait States
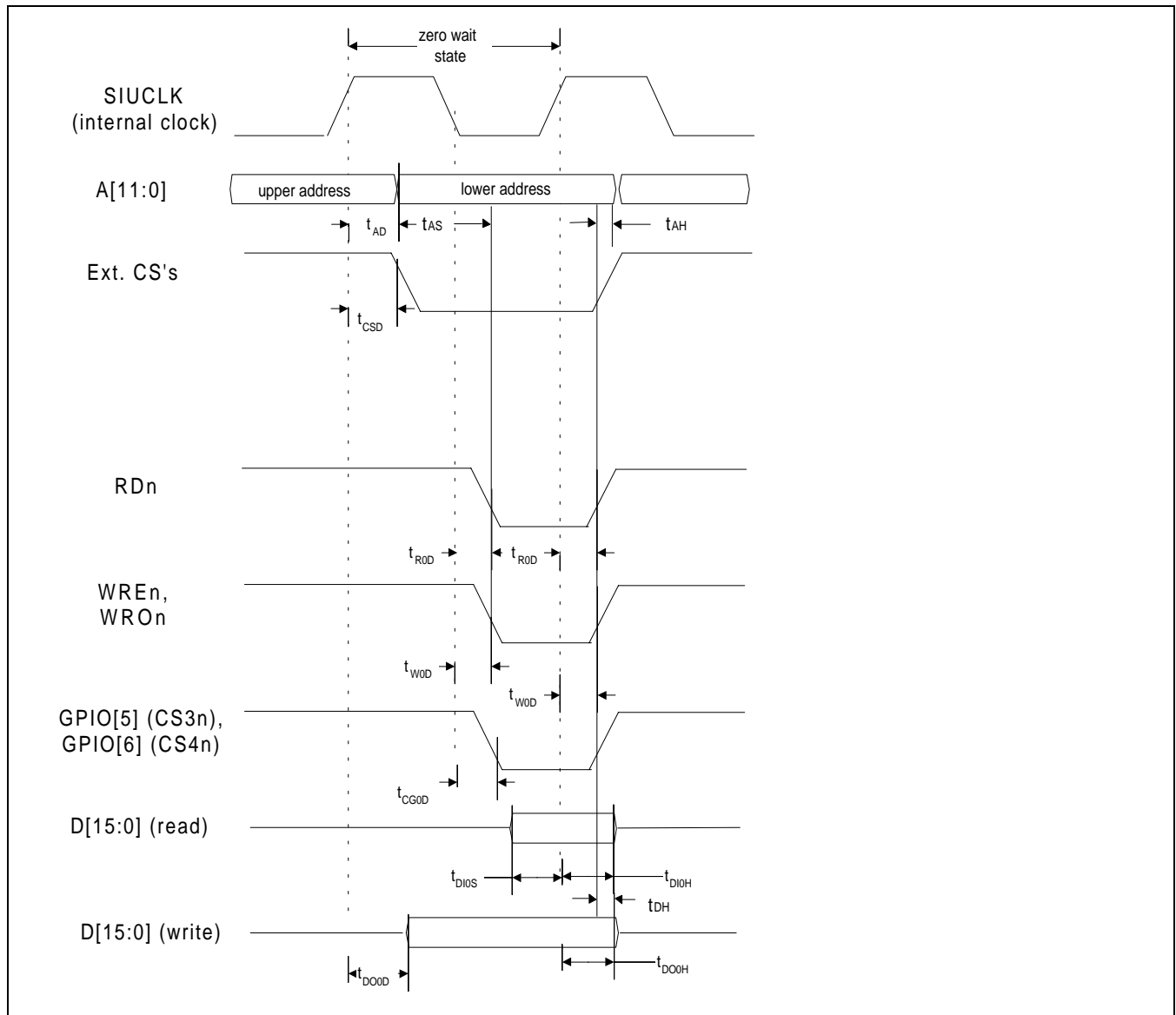
Figure 4-11. System Bus Timing—Zero-Wait-State Read/Write
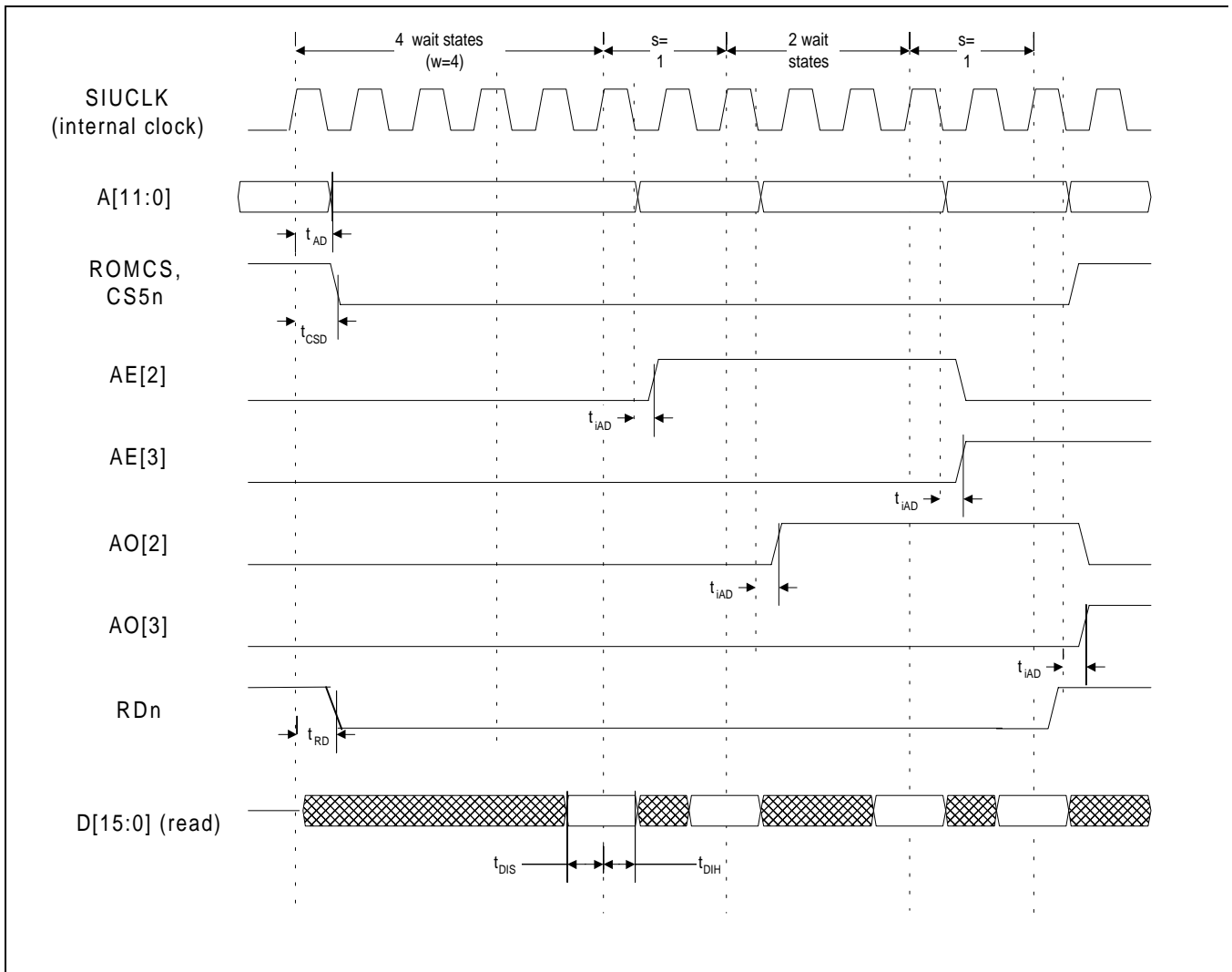
Figure 4-12. System Bus Timing—2-Way Interleave Read Timing (S = 1)
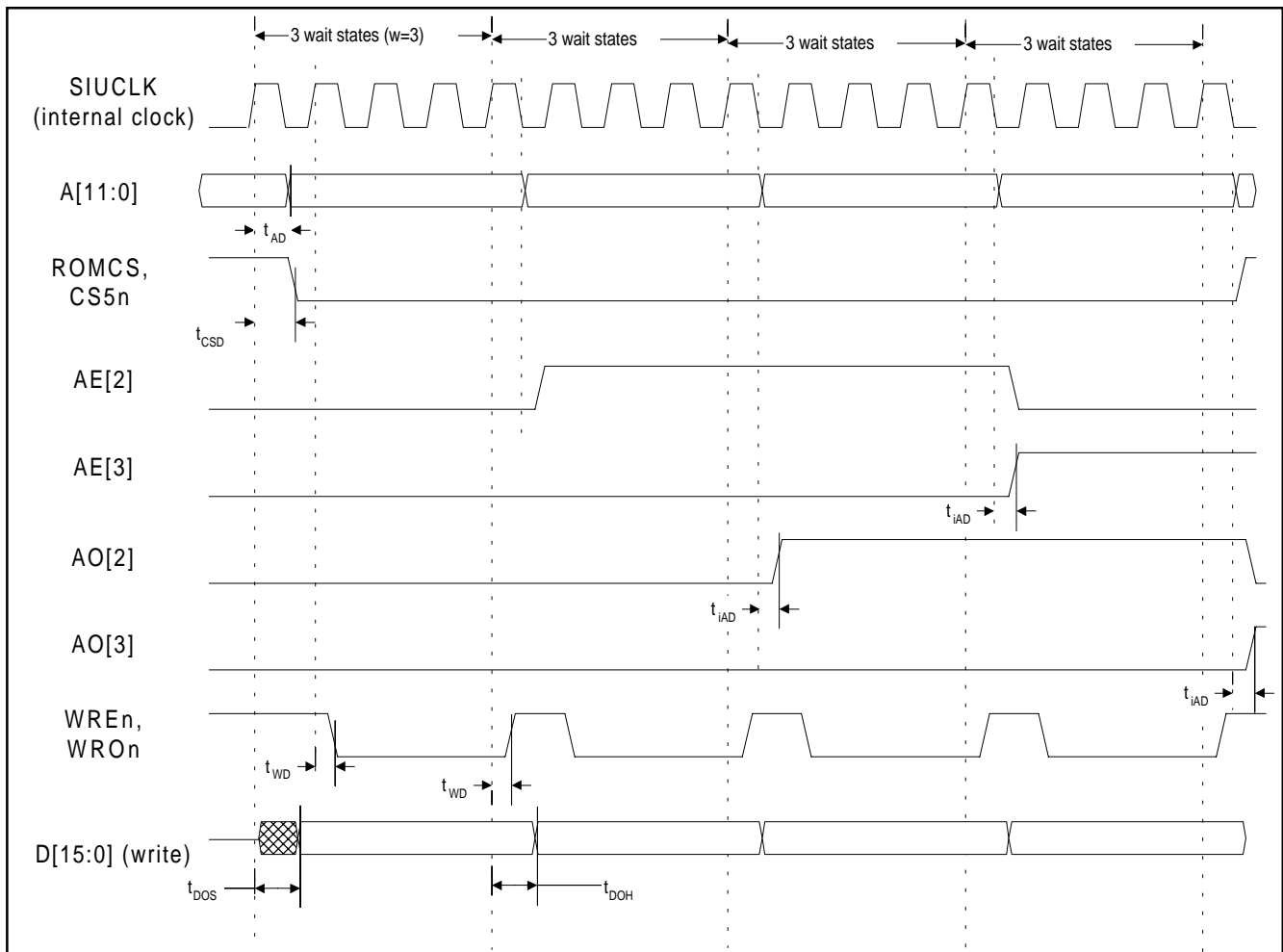
Figure 4-13. System Bus Timing—2-Way Interleave Write Timing (S = 0 or 1)

Table 4-8. Read/Write with Wait States Timing Parameters

Parameter	Symbol	Min.	Max.	Units
Address delay time	t_{AD}	5	20	ns
Chip select delay time	t_{CSD}	-	20	ns
Read delay time for the normal case and delay-on)	t_{RD}	5	18	ns
Write delay time (the normal case, delay-on, and early-off)	t_{WD}	5	12	ns
CS[4:3] delay time (gated with read or write strobe)	t_{CGD}	-	18	ns
Data input setup time	t_{DIS}	8	-	ns
Data input hold time	t_{DIH}	0	-	ns
Data output delay time	t_{DOD}	-	21	ns
Data output hold time	t_{DOH}	5	21	ns
Read delay time (for zero wait state)	t_{R0D}	5	11	ns
Write delay time (for zero wait state)	t_{W0D}	5	11	ns
CS[4:3] delay time (gated with read or write strobe for zero wait state)	t_{CG0D}	-	20	ns
Data input setup time (for zero wait state)	t_{DI0S}	8	-	ns
Data input hold time (for zero wait state)	t_{DI0H}	0	-	ns
Data output delay time (for zero wait state)	t_{DO0D}	-	21	ns
Data output hold time (for zero wait state)	t_{DO0H}	-	21	ns
2-way interleave address delay time	t_{IAD}	-	11	ns
ALE address setup time	t_{AAD}	10		ns
ALE address hold time	t_{AAH}	2		ns
ALE delay time	t_{Iald}	-	10	ns
Address setup time (read and write)	t_{IAS}	3	-	ns
Address hold time (read and write)	t_{IAH}	2	-	ns
Data hold time (write)	t_{IDH}	2	-	ns

Note: SIUCLK is the internal system interface clock. These values are for SIUCLK = 30 MHz. When S=0 in the 2-way interleave read operation, t_{IAD} parameter is still same.

4.3.4   Firmware Operation

Caution   Only word or half-word accesses that happen on their respective boundaries are valid. If the access is to a non-boundary address, the SIU ignores the last 2 LSBs (word access) or 1 LSB (half word access) and reset the address to the appropriate boundaries.

For 16-bit register, writing a byte to the even address (register address) will update the lower 8-bits of the register. Writing a byte to the odd address (register address + 1) will update the upper 8 bits of the register. BS[1:0] indicate which byte is written. Writing a halfword to either the even or odd address will update all 16-bits.

4.4   Interrupt Controller

4.4.1   Function Description

Table 4-9. MFC2000 Interrupt and Reset Signals

Description	External Source	Internal Source	IRQ Number
Modem Interrupt	MIRQn	P80 Core	IRQ0
Countach Bus System Interrupt (irqcbs)		Countach Imaging DSP Bus System	IRQ1
Print subsystem Interrupt	PRTIRQn	—	IRQ2
Scan Step Interrupt (irqsstep)		Motor Control Block	IRQ3
Vertical Print Step Interrupt (irqvpstep)		Motor Control Block	IRQ4
SASIF Interrupt (irqsasif)	—	SASIF Block	IRQ5
DMA ch.2 Interrupt (irqdma2)	—	DMA Control Block	IRQ6
Bi-level Resolution Conversion Interrupt (irqbrc)	—	Bi-level Resolution Conversion Block	IRQ7
DMA ch.10 Interrupt (irqdma10)	—	DMA Control Block	IRQ8
Reset	BATRSTn RESETn	Watchdog Timer & power-down lockout	N/A
VSC IF Interrupt (irqvsc)	—	VSC IF Block	IRQ9
Timer Interrupt 1 (irqtimer1)	—	Interrupt Controller	IRQ10
External Interrupt 1	IRQ11	—	IRQ11
PIO Interrupt (irqpio)	—	PIO Block	IRQ12
External Interrupt 2	IRQ13	—	IRQ13
T.4/T.6 Interrupt (irqt4)	—	T.4/T.6 Block	IRQ14
SOPIF Interrupt (irqsopif)	—	SOPIF Block	IRQ15
System Interrupt (irqsys)	IRQ16	Power Down Block	IRQ16
Software Interrupt (irqsw)	—	Interrupt Controller	IRQ17
SSIF Interrupt (irqssif)	—	SSIF	IRQ18
DMA ch.5 Interrupt (irqdma5)	—	DMA Controller	IRQ19
SmartDAA IF Interrupt (irqsdaa)	—	SmartDAA IF	IRQ20
Timer Interrupt 2 (irqtimer2)	—	Interrupt Controller	IRQ21
USB Interrupt (irqusb)	—	USB Block	IRQ22

This section describes the three methods of interrupting the CPU program flow, which are:

• Reset
• Interrupts for the normal functions (IRQs)
• Interrupt for the development system (through the IRQ16 pin)/Power Down (through the Power Down block)

The reset signal is controlled by the Prime Power Reset block. IRQs and SYSIRQn are managed by the Interrupt Controller and are sent to the ARM as either an IRQ or a FIQ if enabled. Table 4-9 summarizes the interrupts and their sources.

4.4.1.1   Reset

An active level on the CPU Reset input halts program execution and resets the CPU's internal registers. When the CPU's Reset input is released, the CPU begins program execution at the address located in the reset vector. This signal can be activated externally by putting low levels on the BATRSTn or RESETn pins, or internally by the Watchdog Timer or the Battery Power Control logic Lockout circuitry. (For more information, see Section 5-1.)

4.4.1.2   System Interrupt

The system interrupt can be activated externally by the programmable interrupt IRQ16 or by the power down signal from the Power Down Block. This interrupt is treated the same as other interrupts in the interrupt controller. Firmware has the responsibility to make it the highest priority and to use it as the NMI function which is provided by many other CPUs.

The input from the Power Down Block is detected and OR'ed with the programmable external IRQ16 pin. This combined signal is then synchronized to the rising edge of SIUCLK, and then clocked to the falling edge of SIUCLK before an interrupt will be operated in the interrupt controller.

For normal system operation, the system interrupt represents a loss of system power, indicated by Power Down signal going low. The system interrupt control firmware performs the necessary power-down maintenance operations, and then writes to the Lockout Enable register (LockEnn) to protect the battery backed-up registers during loss of power. (Note that activating lockout also generates a reset).

4.4.1.3   Interrupts for Normal Functions

The level-mode interrupt is provided for internal and external interrupts. All internal interrupts are high-level interrupts. The external Modem interrupt is a low-level interrupt. All other external interrupts are programmable to be either high/low/level/edge interrupts. There are only two kinds of registers needed for the interrupt controller; one is the interrupt enable register and another is the interrupt event register. The interrupt controller DOES NOT prioritize the multiple sources of interrupts and DOES NOT generate the interrupt addresses. It only provides interrupt masking for all of interrupts including the system interrupt (i.e., enable/disable control), and generates the interrupt request for the CPU.

When the bit corresponding to an interrupt in the interrupt enable register is set, it enables the interrupt request to cause an interrupt. When the bit is cleared, it masks the interrupt. When the event corresponding to an interrupt bit in the interrupt event register occurs, this bit needs to be set on the rising edge of SIUCLK whether it is enabled or not. On the falling edge of SIUCLK, the interrupt controller generates the interrupt (IRQn and FIQn) to CPU. This interrupt controller has two identical sets of interrupt logic and registers for IRQn and FIQn. Firmware needs to decide which interrupts trigger IRQn and which interrupts trigger FIQn. In the interrupt subroutine, the CPU needs to clear the interrupt event from the interrupt source. Then, this bit will be reset at the following rising edge of SIUCLK. For the software interrupt, the interrupt source is the interrupt bit in the interrupt event register itself. Therefore, the CPU needs to write a 1 to generate the software interrupt and write a 0 to clear the software interrupt.

The source of the IRQ is required to latch the interrupt signal and hold the signal active until the CPU processes the IRQ. The CPU firmware clears the source of the IRQ before exiting the IRQ's service routine. If any IRQ's are pending when new IRQ's are enabled by either setting the interrupt enable registers or the Interrupt Disable bit in the CPU Processor Status register, the enabled IRQ causes an almost immediate CPU interrupt [the CPU only acknowledges interrupts during the op code fetch of an instruction].

External Interrupts

The optional IRQ13 and IRQ11 external interrupt requests share pins with GPIO9 and GPIO8, respectively, and these interrupts are enabled by setting the corresponding bits in the IRQ ENABLE registers to 1. These interrupt enable bits must be set to 0 when using GPIO[9:8] as GPIO to prevent these pins from causing interrupts.

If an external interrupt source is connected to GPIO8 and/or GPIO9, the corresponding GPIO direction control register must remain set to 0 (GPI) [default] to avoid bi-directional conflicts with the GPIO output.

Dedicated external interrupt pins are provided for an active low modem interrupt (MIRQn). All other external interrupts (IRQ2, IRQ11, IRQ13, and IRQ16) are programmable to be either active low or high, edge or level triggered. All external interrupts are resynchronized in the ASIC.

Internal Interrupts

Internal interrupts are provided for the Countach Imaging DSP Bus System (irqcbs), the T.4/T.6 logic (irqt4), the vertical printer stepper motor (irqvpstep), the scan stepper motor (irqsstep), the parallel IO block (irqpio), USB interface (irqusb), the 50ms timer1 and timer2 (irqtimer1, irqtimer2), DMA Channel 2 (irqdma2), Bi-level Resolution Conversion (irqbrc), DMA Channel 10 (irqdma10), Scanner IF (irqvsc), SOPIF (irqsopif), SASIF (irqsasif), software interrupt (irqsw), SSIF (irqssif), DMA Channel 5 interrupt (irqdma5), and the SDAA Interface interrupt (irqsdaa).

4.4.2   Register Description

4.4.2.1   IRQ/FIQ Event1 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
IRQFIQEvent1 0x01FF8021	IRQ15 irqsopif Event Status	IRQ14 irqt4 Event Status	IRQ13 irqext2 Event Status	IRQ12 irqpio Event Status	IRQ11 irqext1 Event Status	IRQ10 irqtimer1 Event Status	IRQ9 irqvsc Event Status	IRQ8 irqdma10 Event Status	Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
IRQFIQEvent1 0x01FF8020	IRQ7 irqbrc Event Status	IRQ6 irqdma2 Event Status	IRQ5 irqsasif Event Status	IRQ4 irqvpstep Event Status	IRQ3 irqsstep Event Status	IRQ2 irqprt Event Status	IRQ1 irqcbs Event Status	IRQ0 MIRQ Event Status	Rst Value 00h Read Value 00h

Bit 15	Internal interrupt from SOPIF block. Read only.
Bit 14	Internal interrupt from T4/T6 block. Read only.
Bit 13	External interrupt 2. Programmable. Read only.
Bit 12	Internal PIO interrupt from PIO block. Read only.
Bit 11	External interrupt 1. Programmable. Read only.
Bit 10	Internal timer 1 interrupt up to 50 ms. Read only.
Bit 9	Internal video scan controller interrupt from VSC IF block. Read only.
Bit 8	Internal DMA channel 10 interrupt from DMA controller block. Read only.
Bit 7	Internal bi-level resolution conversion interrupt from BLRC block. Read only.
Bit 6	Internal DMA channel 2 interrupt from DMA controller block. Read only.
Bit 5	Internal SASIF interrupt from SASIF block. Read only.
Bit 4	Internal vertical print step interrupt from motor control block. Read only.
Bit 3	Internal scan step interrupt from motor control block. Read only.
Bit 2	External print subsystem interrupt. Programmable. Read only.
Bit 1	Internal Countach bus system interrupt from Countach Bus System. Read only.
Bit 0	External modem interrupt (active low) or the internal P80 core interrupt. Read only.

4.4.2.2   IRQ/FIQ Event2 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
IRQFIQEvent2 0x01FF8023	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
IRQFIQEvent2 0x01FF8022	(Not Used)	IRQ22 irqusb Event Status	IRQ21 irqtimer2 Event Status	IRQ20 irqsdaa Event Status	IRQ19 irqdma5 Event Status	IRQ18 irqssif Event Status	IRQ17 irqsw Event Status	IRQ16 irqsys Event Status	Rst Value x0000000b Read Value 00h

Bit 6   Internal USB interrupt from USB block. Read only.

Bit 5   Internal timer 2 interrupt up to 50 ms. Read only.

Bit 4   Internal SmartDAA interface interrupt from SmartDAA IF block. Read only.

Bit 3   Internal DMA channel 5 interrupt from DMA controller block. Read only.

Bit 2   Internal SSIF from SSIF block. Read only.

Bit 1   Internal Software interrupt. When CPU writes a 1, the software interrupt is issued. When CPU writes a 0, the software interrupt is cleared. R/W.

Bit 0   Internal system interrupt from programmable external interrupt 16 or power down circuit Read only.

4.4.2.3   IRQ Enable1 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
IRQEnable1 0x01FF8025	Enable IRQ15 irqsopif	Enable IRQ14 irqt4	Enable IRQ13 irqext2	Enable IRQ12 irqpio	Enable IRQ11 irqext1	Enable IRQ10 irqtimer1	Enable IRQ9 irqvsc	Enable IRQ8 irqdma10	Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
IRQEnable1 0x01FF8024	Enable IRQ7 irqbrc	Enable IRQ6 irqdma2	Enable IRQ5 irqsasif	Enable IRQ4 irqvpstep	Enable IRQ3 irqsstep	Enable IRQ2 irqprt	Enable IRQ1 irqcbs	Enable IRQ0 MIRQ	Rst Value 00h Read Value 00h

Bit 15 – 0:   When 1 will enable the corresponding interrupt and when 0 will mask that interrupt out. R/W.

4.4.2.4   IRQ Enable2 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
IRQEnable2 0x01FF8027	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
IRQEnable2 0x01FF8026	(Not Used)	Enable IRQ22 irqusb	Enable IRQ21 irqtimer2	Enable IRQ20 irqsdaa	Enable IRQ19 irqdma5	Enable IRQ18 irqssif	Enable IRQ17 irqsw	Enable IRQ16 irqsys	Rst Value x0000000b Read Value 00h

Bit 3 – 0:   When 1 will enable the corresponding interrupt and when 0 will mask
                                  that interrupt out. R/W.

4.4.2.5   FIQ Enable1 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
FIQEnable1 0x01FF8029	Enable IRQ15 irqsopif	Enable IRQ14 irqt4	Enable IRQ13 irqext2	Enable IRQ12 irqpio	Enable IRQ11 irqext1	Enable IRQ10 irqtimer1	Enable IRQ9 irqvsc	Enable IRQ8 irqdma10	Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
FIQEnable1 0x01FF8028	Enable IRQ7 irqbrc	Enable IRQ6 irqdma2	Enable IRQ5 irqsasif	Enable IRQ4 irqvpstep	Enable IRQ3 irqsstep	Enable IRQ2 irqprt	Enable IRQ1 irqcss	Enable IRQ0 MIRQ	Rst Value 00h Read Value 00h

Bit 15 – 0:   When 1 will enable the corresponding interrupt and when 0 will mask
                                  that interrupt out. R/W.

4.4.2.6   FIQ Enable2 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
FIQEnable2 0x01FF802B	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
FIQEnable2 0x01FF802A	(Not Used)	Enable IRQ22 irqusb	Enable IRQ21 irqtimer2	Enable IRQ20 irqsdaa	Enable IRQ19 irqdma5	Enable IRQ18 irqssif	Enable IRQ17 irqsw	Enable IRQ16 irqsys	Rst Value x0000000b Read Value 00h

Bit 3 – 0:   When 1 will enable the corresponding interrupt and when 0 will mask
                                  that interrupt out. R/W.

4.4.2.7   External Interrupt Configuration Register

Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
EIRQConfig **0x01FF802C**	irq16edge	irq13edge	irq11edge	irq2edge	irq16actlo	irq13actlo	irq11actlo	irq2actlo	Rst Value 00h Read Value 00h

Bit 7 – 4:   Write a 1 will configure the corresponding external interrupt to be edge triggered and write a 0 will configure it to be level triggered.

Bit 3 – 0:   Write a 1 will configure the corresponding external interrupt to be active low and write a 0 will configure it to be active high.

4.4.2.8   External Interrupt Clear Register

Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
EIRQClear **0x01FF802D**	(Not Used)	(Not Used)	irq21clr	irq10clr	eirq16clr	eirq13clr	eirq11clr	eirq2clr	Rst Value xx000000b Read Value 00h

Bit 3 – 0:   Firmware uses these bits to clear the corresponding edge triggered external interrupt which is latched in the interrupt controller. After writing a 1 to clear the interrupts, the bits reset themselves to 0.

Bit 5 – 4:   Firmware uses these bits to clear the corresponding internal timer interrupts. After writing a 1 to clear the interrupts, the bits reset themselves to 0.

4.4.2.9   Timer1 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Timer1 **0x01FF802F**				Timer 1 Value MSB					Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Timer1 **0x01FF802E**				Timer 1 Value LSB					Rst Value 00h Read Value 00h

Bit 15 – 0:   This is the timer value for the timer1 interrupt. This value will be loaded in a counter when the timer interrupt bit is enabled. The value loaded in this register is dependent on the SIUCLK frequency. This interrupt period can be programmed up to 50 ms with a programmable resolution. To write a new timer value into the register, the enable bit in the IRQ/FIQ Enable register must be disabled first; the new timer value is then written into the register and the enable bit is set to load the new timer value into the counter.

4.4.2.10   Timer2 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Timer2 **0x01FF8031**				Timer 2 Value MSB					Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Timer1 **0x01FF8030**				Timer 2 Value LSB					Rst Value 00h Read Value 00h

Bit 15 – 0:   This is the timer value for the timer2 interrupt. This value will be loaded in a counter when the timer interrupt bit is enable. The value loaded in this register is dependent on the SIUCLK frequency. This interrupt period can be programmed up to 50 ms with a programmable resolution (see Table 4-10). To write new timer value into the register, the enable bit in the IRQ/FIQ Enable register has to be disabled first; the new timer value is then written into the register and the enable bit is then set to load the new timer value into the counter.

The resolution of the timer1 and timer2 is dependent on SIUCLK and can be calculated as follows:

$$TMRCLK = (SIUCLK/B)/8 = ICLK/8 \text{ (value of B is programmable)}$$

Table 4-10. Programmable Resolution of Timer1 and Timer2

SIUCLK (MHz)	B	ICLK (MHz)	TMRCLK (MHz)	TMRCLK (uSec)
30	3	10	1.25	0.8
30	4	7.5	0.9375	1.067
37.5	4	9.375	1.171875	0.833
40	4	10	1.25	0.8

4.4.3   Timing

SIUCLK
(internal)

MIRQn

PRTIRQn(IRQ2)
or
GPIO[8](IRQ11)
or
GPIO[9](IRQ13)
or
IRQ16(SYSIRQ)
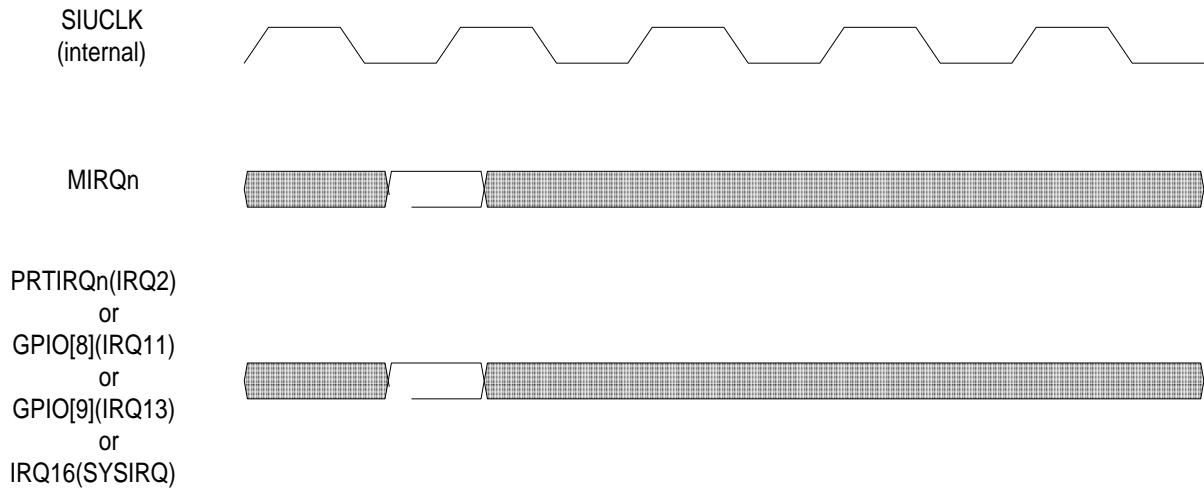
Figure 4-14. External Interrupt Request Timing

> **Note**:   The MFP2000 chip resynchronizes MIRQn, PRTIRQn, GPIO[8], GPIO[9], and
> IRQ16 signals internally. There are no setup time and hold time requirements for MIRQn,
> PRTIRQn, GPIO[8], GPIO[9], MIRQn, and IRQ16 signals with respect to SYSCLK. The four
> external interrupts PRTIRQn, GPIO[8], GPIO[9], and IRQ16 can also be programmed as edge
> triggered interrupts. In this case, the interrupt signals are implemented as clock into flip-flops
> with D-input either tied to high or low; again there is no setup and hold time requirements either.

4.5   DRAM Controller (Including Battery DRAM)

4.5.1   Functional Description

The DRAM Controller interfaces to external memory devices and to the internal ARM7 SIU block. The DRAM
memory space can be divided into two banks of memory which can be independently configured. The system
clock rate that is supported can be up to 40MHz and can support the DRAM characteristics that are listed in the
following tables

Addressing Size:	512K, 1M, 4M, 16M
Organization:	4 bits, 8 bits, or 16 bits
Access Speed:	50, 60, 70, 80 ns

The maximum memory size that is supported for two memory banks is 32M. The DRAM Chip sizes that are
supported go up to 16M, but are limited to the row/column configurations that can be accommodated from the
address multiplexing table (Table 4-12) and the DRAM row/column configuration Table 4-14).

The number of DRAM access and refresh cycle wait states can be programmed from the DRAMCtrl register.
Specifically, options to control the RAS precharge width, RAS low time, and CAS low time are provided. The drive
capability of the DRAM control signals can support a maximum of 50pF of loading capacitance.

Several types of external DRAM configurations can be supported: non-interleaved (8-bit or 16-bit data bus) and 2-
way interleaved (16-bit data bus) (See Table 4-11). Memory bank 0 can be configured independently from
memory bank 1.

If a burst of data is sent to the DRAM, the DRAM Controller will run in page mode once the initial access is completed. The maximum burst length is limited to 8 halfwords (i.e., the maximum burst length coincides with the CACHE line length) and is controlled by a burst signal that is generated from the SIU. If a burst of data is being sent to the DRAM, but an octal address boundary occurs, the burst signal will turn off causing a RASn precharge. It is impossible to go across a page boundary without precharging the RASn signal.

> **Note:** If a 16-bit wide memory structure is implemented, bursts of data must be 16-bit halfword bursts. 8-bit byte bursts are only allowed for an 8-bit wide memory structure.

Table 4-11. DRAM Wait State Configurations

Non-Interleaved Modes (8 or 16 bit interfaces)		
1 Cycle CASn		
30 MHz	-50, -60	3 wait state, PG = 1 wait state
2 Cycle CASn		
30 MHz	-50, -60, -70, -80	3 wait state, PG = 2 wait state (read)
		2 wait state, PG = 1 wait state (write)
37.5 MHz and 40 MHz	-50, -60	5 wait state, PG = 2 wait state (read)
		4 wait state, PG = 1 wait state (write)
37.5 MHz	-70	5 wait state, PG = 2 wait state (read)
		4 wait state, PG = 1 wait state (write)
Interleaved Mode (16-bit interface)		
30 MHz	-50, -60, -70, -80	3 wait state, PG = 0,1,0 wait state (read)-Even starting address, non octal boundary
		4 wait state, PG=1,0,1 wait state (read)- Odd starting address, non octal boundary
		2 wait state, PG = 1 wait state (write)
37.5 MHz and 40 MHz	-50, -60	5 wait state, PG = 0,1,0 wait state (read)
		4 wait state, PG = 1 wait state (write)
37.5 MHz	-70	5 wait state, PG = 0,1,0 wait state (read)
		4 wait state, PG = 1 wait state (write)

> **Note**: PG = page mode

4.5.1.1   Memory Bank Structure

DRAM address space can be selected in 2 separate memory blocks (Bank 0: RASn[0] and CASOn[0] (8-bit) or CASOn[1:0] (16-bit) or CASOn[1:0] and CASEn[1:0] (interleaved), Bank 1: RASn[1] and CASOn[0] (8-bit) or CASOn[1:0] (16-bit) or CASOn[1:0] and CASEn[1:0] (interleaved). Separate control bits are provided in the Backup Configuration register to enable and disable (default) each of the memory banks. Each bank has separate configuration controls and the address ranges of the two memory banks is continuous around the midpoint of the DRAM memory bank. The RASn[1] starting address is 03000000h and grows larger based on the size of the memory. The end of the RASn[0] bank ends at 03000000h and grows smaller from that point. The memory range is programmed through the address multiplexer selections for bank 0 and bank 1 in the DRAMCtrl register.

4.5.1.2   Non-Interleaved DRAM Accesses

Non-interleaved DRAM accesses are available for 8-bit or 16-bit data bus. Byte access is available for both 8-bit and 16-bit data bus and 16-bit halfword access is available for 16-bit data bus. DRAM early-write mode, normal read mode and page mode are supported. Read-modify-write is not supported.

> **Note**: *16-bit DRAMs must have upper and lower CAS's in order to work with the DRAM controller. 8-bit bursts of data will not work with a 16-bit wide memory structure.*
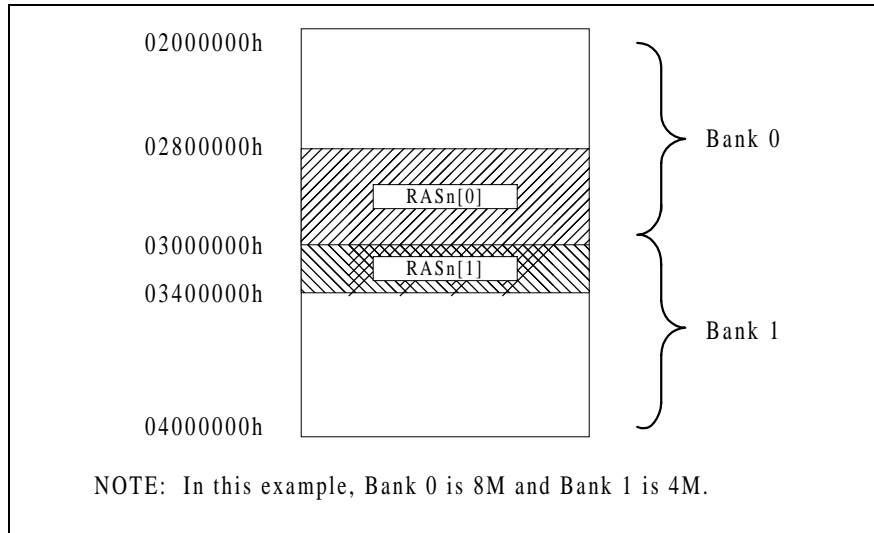
Figure 4-15. DRAM Bank/Address Map

4.5.1.3   2-way Interleaved DRAM Accesses

The two-way interleaved DRAM interface can support up to four 16-bit wide devices a maximum of 8M deep. Bank 0 is selected with RASn[0] and bank 1 is selected with RASn[1]. CASEn[1:0], CASOn[1:0], DWRn, DOEOn, DOEEn, ADDR, and DATA are common between the two banks. 2-way interleaving is limited to a 16-bit wide databus. 8-bit or 16-bit wide devices can be used. The ARM CPU can write 32-bit words, 16-bit halfwords or bytes to the memory banks. The addressing to the interleaved DRAMs starts with address bit 2. Bits 1 and 0 are used internally to generate the proper CASOn[1:0] and CASEn[1:0] signals. When the memory structure is configured for two-way interleaving, byte bursts are not allowed. Only bursts of 32-bit words or 16-bit halfwords are allowed. The maximum burst length is eight 16-bit halfwords. The burst length is controlled by the DRAMBURST signal that is sent from the SIU. The external memory structure can use the output enables directly to the memory device or for increased speed can use external bus transceivers. Bus contention must be considered when the output enables are tied directly to a DRAM memory.

Table 4-12. Address Multiplexing—Part 1

Address Multiplexing Register	Select Option 000		Select Option 001		Select Option 010		Select Option 011	
Physical Address	ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.
A[13]	A[22]	A[13]	A[23]	A[13]	A[24]	A[13]	A[25]	A[13]
A[12]	A[21]	A[12]	A[22]	A[12]	A[23]	A[12]	A[24]	A[12]
A[11]	A[20]	A[11]	A[21]	A[11]	A[22]	A[11]	A[23]	A[11]
A[10]	A[19]	A[10]	A[20]	A[10]	A[21]	A[10]	A[22]	A[10]
A[9]	A[18]	A[9]	A[19]	A[9]	A[20]	A[9]	A[21]	A[9]
A[8]	A[17]	A[8]	A[18]	A[8]	A[19]	A[8]	A[20]	A[8]
A[7]	A[16]	A[7]	A[17]	A[7]	A[18]	A[7]	A[19]	A[7]
A[6]	A[15]	A[6]	A[16]	A[6]	A[17]	A[6]	A[18]	A[6]
A[5]	A[14]	A[5]	A[15]	A[5]	A[16]	A[5]	A[17]	A[5]
A[4]	A[13]	A[4]	A[14]	A[4]	A[15]	A[4]	A[16]	A[4]
A[3]	A[12]	A[3]	A[13]	A[3]	A[14]	A[3]	A[15]	A[3]
A[2]	A[11]	A[2]	A[12]	A[2]	A[13]	A[2]	A[14]	A[2]
A[1]	A[10]	A[1]	A[11]	A[1]	A[12]	A[1]	A[13]	A[1]
A[0]	A[9]	A[0]	A[10]	A[0]	A[11]	A[0]	A[12]	A[0]

Table 4-13. Address Multiplexing—Part 2

Address Multiplexing Register	Select Option 100		Select Option 101		Select Option 110		Select Option 111	
Physical Address	ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.
A[11]	A[24]	A[13]	A[23]	A[13]	A[24]	A[12]	A[22]	A[12]
A[10]	A[23]	A[12]	A[22]	A[12]	A[23]	A[11]	A[21]	A[11]
A[9]	A[22]	A[11]	A[21]	A[11]	A[22]	A[10]	A[20]	A[10]
A[8]	A[21]	A[10]	A[20]	A[10]	A[21]	A[9]	A[19]	A[9]
A[7]	A[20]	A[9]	A[19]	A[9]	A[20]	A[8]	A[18]	A[8]
A[6]	A[19]	A[8]	A[18]	A[8]	A[19]	A[7]	A[17]	A[7]
A[5]	A[18]	A[7]	A[17]	A[7]	A[18]	A[6]	A[16]	A[6]
A[4]	A[17]	A[6]	A[16]	A[6]	A[17]	A[5]	A[15]	A[5]
A[3]	A[16]	A[5]	A[15]	A[5]	A[16]	A[4]	A[14]	A[4]
A[2]	A[15]	A[4]	A[14]	A[4]	A[15]	A[3]	A[13]	A[3]
A[1]	A[14]	A[3]	A[13]	A[3]	A[14]	A[2]	A[12]	A[2]
A[0]	A[13]	A[2]	A[12]	A[2]	A[13]	A[1]	A[11]	A[1]

Table 4-14. DRAM Row/Column Configuration

Memory Size	Address Multiplex Setting			Supported/Not Supported	Row/Column Configuration	
	8-bit	16-bit	2-wy intrl.		Row	Column
256K x 8	000	000	000	Supported	9	9
256K x 16 DRAMs	000	000	000	Supported	9	9
	---	---	---	Not Supported	10	8
512K x 8	000	000	000	Supported	10	9
1M x 8	001	001	001	Supported	10	10
1M x 16 DRAMs	001	001	001	Supported	10	10
	---	---	---	Not Supported	12	8
4M x 8	010	010	100	Supported	11	11
4M x 16 DRAMs	001	111	101	Supported	12	10
	---	---	---	Not Supported	13	9
16M x 8	011	110	---	Supported	12	12
	---	---	---	Not Supported	13	11

4.5.1.4   Refresh Operation

DRAM Refresh is performed automatically using the CAS-before-RAS method. Three different refresh speeds are supported: slow, normal and fast. These speeds are selected by bits in the backup configuration register. The refresh time is based on the crystal oscillator frequency and the refresh rate that is selected. During prime power when it is time to refresh the DRAM, a CAS-before-RAS refresh cycle will be inserted. If the DRAM is being accessed when a DRAM refresh is requested, the refresh cycle is not inserted until the access is complete. The maximum burst length that the DRAM Controller will see is 8 halfwords (i.e., the maximum burst length coincides with the CACHE line size). The maximum burst length is controlled by the SIU with the DRAMBURST signal that is sent to the DRAM Controller.

4.5.1.5   Power Down Mode

When the ASIC is powered down, the DRAMs cannot be accessed. Only DRAM refresh will continue on battery power (VDRAM). Refresh timing is generated from a one shot and an internal gate delay circuit during battery powered operation. To ensure a smooth transition from VDD refresh to battery powered refresh, a control signal from the power reset block allows the DRAM controller to switch from VDD refresh to battery refresh when power is down. When the prime power is reapplied, the refresh logic switches back to VDD refresh. The refresh speed selected using the BackupConfig register remains in force during the battery backed up mode. The DRAM Backup time duration is defined by the two DRAM Backup time bits in the BackupConfig register. No backup, 1-2 days, 2-3 days, and infinite are the backup options.

> **Note:**   The AMFPC ASIC uses a 3V process; therefore, if the DRAM memory structure is to be backed up, for the lowest power consumption 3 DRAMs should be used. Also, any external circuitry must also be battery powered. The output pads of the AMFPC are only 5V tolerant during high Z. The simplified block diagram for the DRAM controller is illustrated in Figure 4-16.
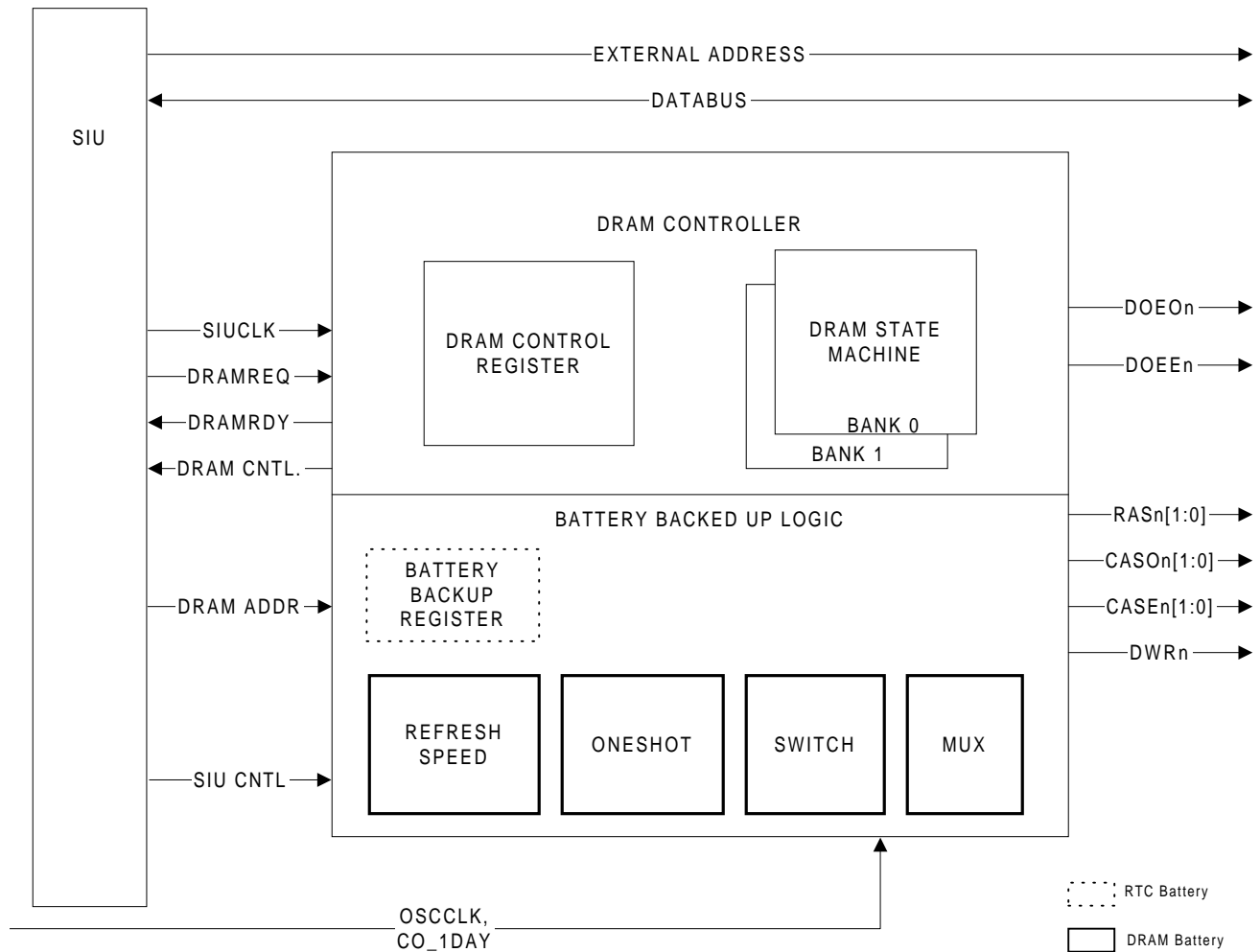
Figure 4-16. Simplified DRAM Controller Block Diagram
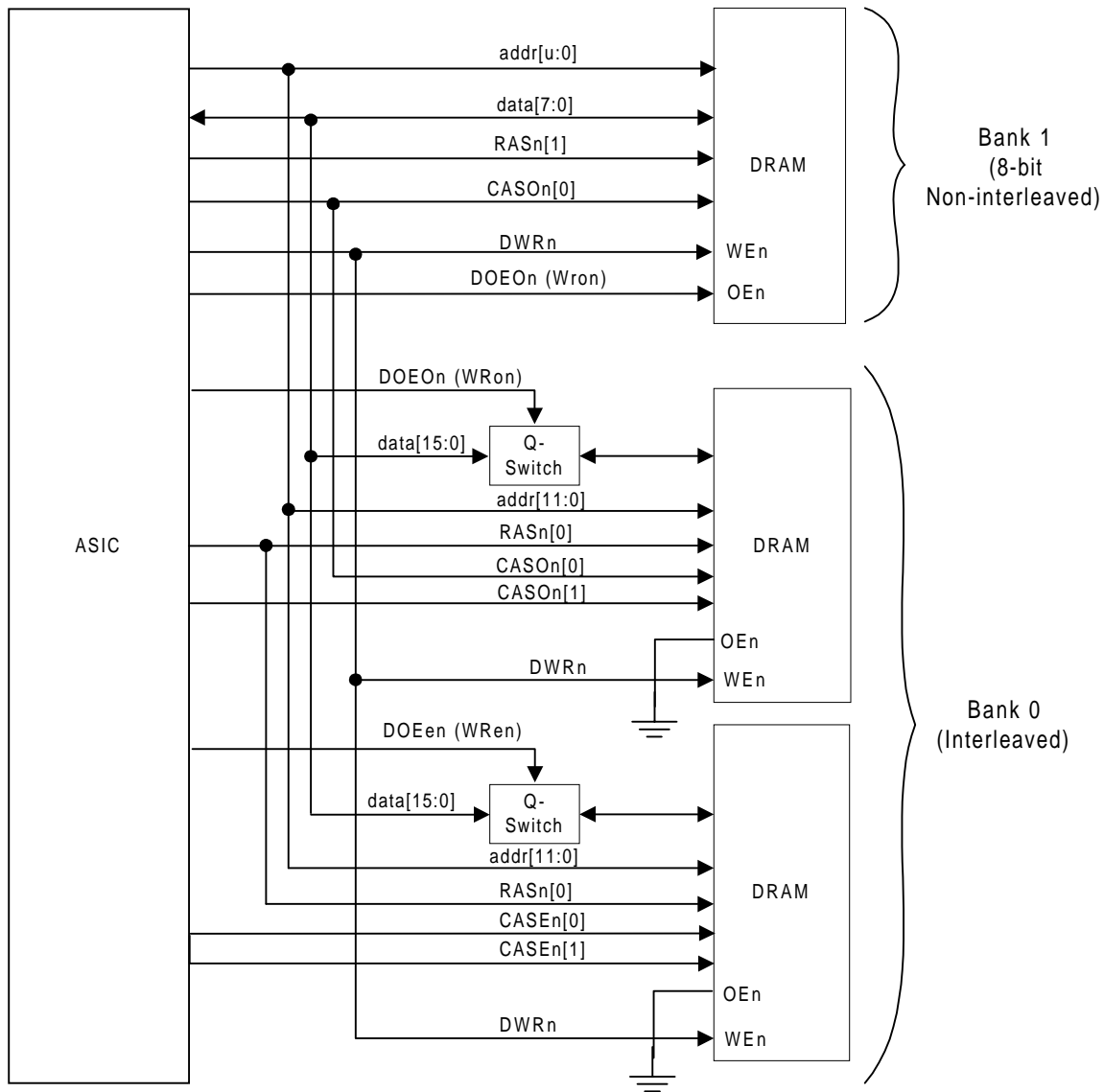
Figure 4-17. DRAM Interface Example

Figure 4-17 gives an example of how each bank of DRAMs might be setup for non-battery back-up DRAM system. In this example, Bank 0 is setup for an 8-bit non-interleaved memory bank and Bank 1 is setup with a 16-bit 2-way interleaved DRAM bank.

Note:

1. *DWRn is a battery backed-up signal and is 'high' during the battery back-up mode. All inputs of the prime powered logic will have 'no power' or 'low' during the battery back-up mode. Therefore, all external logic, which uses DWRn, should be battery backed-up logic and should be gated with WRPROTn to ensure that outputs are 'low' when the prime power is off for the battery back-up DRAM operation.*

2. *DWRn, CASO[1:0]n and CASE[1:0]n are shared by both DRAM banks (RAS[0]n and RAS[1]n). If you only want to back up one bank by battery power, all shared signals should be separated by the external logic and should follow the rule in note 2.*

Conexant

4.5.2   Register Description

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
DRAM Control Register *(DRAMCtrl1)* **$01FF8821**	Bank 1 Address Multiplexing			Bank 1 Increase RAS Cycle Time	Bank 1 2-cycle RAS Precharge	Bank 1 1-cycle RAH	Bank 1 Non-interleaved Speed Control 00 = Fast Mode 01 = Normal Mode 10 = Slow Mode 11 = N/A		Rst. Value x0000000b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
DRAM Control Register *(DRAMCtrl1)* **$01FF8820**	Bank 0 Address Multiplexing			Bank 0 Increase RAS Cycle Time	Bank 0 2-cycle RAS Precharge	Bank 0 1-cycle RAH	Bank 0 Non-interleaved Speed Control 00 = Fast Mode 01 = Normal Mode 10 = Slow Mode 11 = N/A		Rst. Value x0000000b Read Value 00h

Register Description: The DRAM Control register is used to program the two DRAM banks for the type of operation that is desired.

Bits 15-13: Bank 1 Address Multiplexing

This register controls the addressing multiplexing for Bank 1

Bit 12:   Bank 1 Increase RAS Cycle Time

This register will add one cycle after the refresh cycle prior to RAS precharge to meet T_{RC} (RASn cycle time). This is needed in order to use 70 ns DRAMs while running at 39MHz.

Bit 11:   Bank 1 2-cycle RAS Precharge

This register will increase the RASn[1] Precharge time from 1 to 2 clock cycles.

Bit 10:   Bank 1 1-cycle RAH

This register will increase the RASn[1] address hold time. When this bit is set, the address will be multiplexed 1 clock cycle after the falling edge of RAS[1]n. The default setting will multiplex the row/column address ½ clock cycle after the falling edge of RAS[1]n.

Bit 9-8:   Bank 1 Speed Control

These registers will control the speed of the DRAM interface for bank 1 when in the non-interleaved mode. This register controls the width of the CASn signal. These bits are ignored in interleaved mode.

Bit 9	Bit 8	Non-Interleaved Operation:
0	0	CASn is ½ clock cycle wide.
0	1	CASn is 1 clock cycle wide.
1	0	CASn is 2 clock cycles wide for Read and 1 clock cycle wide for write.
1	1	N/A

Bits 7-5:   Bank 0 Addressing Multiplexing:

This register controls the addressing multiplexing for Bank 0 (Table 4-12).

Bit 4: Bank 0 Increase RAS Cycle Time

> This register will add one cycle after the refresh cycle prior to RAS precharge to meet T_{RC} (RASn cycle time). This is needed in order to use 70 ns DRAMs while running at 39MHz.

Bit 3: Bank 0 2-cycle RAS Precharge

> This register will increase the RASn[0] Precharge time from 1 to 2 clock cycles.

Bit 2: Bank 0 1-cycle RAH

> This register will increase the RASn[0] address hold time. When this bit is set, the address will be multiplexed 1 clock cycle after the falling edge of RAS[0]n. The default setting will multiplex the row/column address ½ clock cycle after the falling edge of RAS[1]n.

Bits 1-0: Bank 0 Speed Control

> These registers will control the speed of the DRAM interface for bank 0 when in the non-interleaved mode. This register controls the width of the CASn signal. These bits are ignored in interleaved mode.

Bit 1	Bit 0	Non-Interleaved Operation
0	0	CASn is ½ clock cycle wide.
0	1	CASn is 1 clock cycle wide.
1	0	CASn is 2 clock cycles wide for Read and 1 clock cycle wide for write.
1	1	N/A

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Backup Configuration Register *(BackupConfig)* **$01FF8099**	(Not Used)	(Not Used)	Internal Power Down Select	Batrstn Detected	Lockenn Timeout Detected	SRAM Enable 0 = disable 1 = enable	Bank 1 Data interface size: 0 = 8-bit 1 = 16-bit	Bank 0 Data interface size: 0 = 8-bit 1 = 16-bit	Rst. Value xxxxx000b Read Value 00h

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Backup Configuration Register *(BackupConfig)* **$01FF8098**	DRAM Backup Time 0 = no backup 1 = 1-2 days 2 = 2-3 days 3 = infinite days		Refresh Rate 0 = normal 1 = slow	Oscillator Speed 0 = 32.768 kHz 1 = 65.536 kHz	Bank 1 Enable 0 = disable 1 = enable	Bank 0 Enable 0 = disable 1 = enable	Bank 1 Interleave Enable 0 = non interleaved 1 = 2 way interleaved	Bank 0 Interleave Enable 0 = non interleaved 1 = 2 way interleaved	Rst. Value 00h Read Value 00h

Register Description: This register is set to all zeros when first powered up and is battery backed up with the RTC Battery during power down. When a time out condition occurs, the RASn and CASn signals are tri-stated. When prime power has returned from a power down sequence, the user will have to perform a checksum on the DRAM data to know if a time out has occurred since there is no indication that the DRAM battery has lost power. The user will have to wait 1ms before accessing the DRAM after prime power has returned.

Bits 15-14: Not used

Bit 13: Internal Power Down Select

> 0 = PWRDWNn is generated by or-ing power_down1 with power_down2
> 1= PWRDWNn is generated by and-ing power_down1 with power_down2

> *Note*:   *Power_down1 and power_down2 are output signals from the power-down detection circuit 1 and 2.*

Bit12:   Betrstn Detected

This bit indicates that a betrstn occurred. To clear this bit, a 1 must be written to this bit.

Bit11:

This bit indicates that a power down occurred, but no lockout was set within the 1-2 second period. The lockout timer initiated the lockenn to create the lockout condition. Once the lockout condition occurs, if the power down signal is high, the chip will come out of reset after a pud1 delay. To clear this bit a 1 must be written to this bit.

Bit10:   SRAM Chip Select Enable

This bit enables the SRAM chip select CSN0.

Bit 9:   Bank 1 Interface Size

This register defines whether the data bus to the bank 1 DRAMs is 8 bits wide or 16 bits wide. An 8-bit wide DRAM interface uses RASn[1] and CASOn[0]. A 16-bit wide non-interleaved DRAM interface uses RASn[1] and CASOn[1:0]. A 16-bit wide interleaved DRAM interface uses RASn[1], CASEn[1:0], and CASOn[1:0].

Bit 8:   Bank 0 Interface Size

This register defines whether the data bus to the bank 0 DRAMs is 8 bits wide or 16 bits wide. An 8-bit wide DRAM interface uses RASn[0] and CASOn[0]. A 16-bit wide non-interleaved DRAM interface uses RASn[0] and CASOn[1:0]. A 16-bit wide interleaved DRAM interface uses RASn[0], CASEn[1:0], and CASOn[1:0].

Bits 7-6:   DRAM Battery Backup Time

These bits control the amount of time that the DRAM controller will spend refreshing the DRAMs when in the battery backup mode. After reset when the CPU is being released to run, the CPU will not be able to write data to bits 7 and 6 of the BackupConfig register immediately since the immediate write will not take effect. The CPU must wait at least one oscillator clock cycle before writing data into bits 7 and 6.

Bit 7	Bit 6	Battery Backup Duration:
0	0	No battery backup (default)
0	1	1-2 days
1	0	2-3 days
1	1	Infinite

Bit 5-4:   DRAM Refresh Rate

These bits are used to set up the DRAM refresh rate. See the following table:

Refresh Speed: (bit 5)	Oscillator Speed: (bit 4)	Refresh Speed:	Description:
0	0	normal	RTC crystal frequency = 32.768 kHz, Refresh clock = the crystal frequency = 32.768 kHz, The refresh cycle time = 15.625 ms/1024 cycles.
0	1	fast	RTC crystal frequency = 65.536 kHz, Refresh clock = the crystal frequency = 65.536 kHz, Refresh cycle time = 7.8125 ms/1024 cycles.
1	0	slow	RTC crystal frequency = 32.768 kHz, Refresh clock = the crystal frequency/8 = 4.096 kHz Refresh cycle time = 125 ms/1024 cycles.
1	1	slow	RTC crystal frequency = 65.536 kHz, Refresh clock = the crystal frequency/16 = 4.096 kHz Refresh cycle time = 125 ms/1024 cycles.

Bits 3:   Bank 1 Enable   This bit controls whether or not the Bank 1 DRAMs will be enabled. The Enable signal will allow CAS before RAS refresh to occur based on the non-interleave or interleave setting. If the bank setting indicates a non-interleaved mode, RASn[1] and CASOn[0] (8-bit mode) or CASOn[1:0] (16-bit mode) will refresh the DRAM. If the bank setting indicates an interleaved mode, RASn[1], CASOn[1:0] and CASEn[1:0] will refresh the DRAM. DWRn will be high during refresh. If bank 1 is disabled, RAS[1]n will be tri-stated and all appropriate CASn's will be tri-stated based on mode settings.

Bit 2:   Bank 0 Enable   This bit controls whether or not the Bank 0 DRAMs will be enabled. The Enable signal will allow CAS before RAS refresh to occur based on the non-interleave or interleave setting. If the bank setting indicates a non-interleaved mode, RASn[0] and CASOn[0] (8-bit mode) or CASOn[1:0] (16-bit mode) will refresh the DRAM. If the bank setting indicates an interleaved mode, RASn[0], CASOn[1:0] and CASEn[1:0] will refresh the DRAM. DWRn will be high during refresh. If bank 0 is disabled, RAS[0]n will be tri-stated and all appropriate CASn's will be tri-stated based on mode settings.

Bit 1:   Bank 1 Interleave Enable   This register defines whether the bank 1 DRAMs are to be accessed using 2-way interleave or non interleaved access. 2-way interleaved access is only valid with a 16-bit interface (the 16 bit vs. 8 bit interface size bit for bank 1 is ignored by the DRAM controller, but is used by the SIU to output the data correctly).

Bit 0:   Bank 0 Interleave Enable   This register defines whether the bank 0 DRAMs are to be accessed using 2-way interleave or non interleaved access. 2-way interleaved access is only valid with a 16-bit interface (the 16 bit vs. 8 bit interface size bit for bank 1 is ignored by the DRAM controller, but is used by the SIU to output the data correctly).

4.5.3   Brief Timing

No matter which mode you use and which address you access, the DRAM access timing is lined up with the octal halfword boundary.
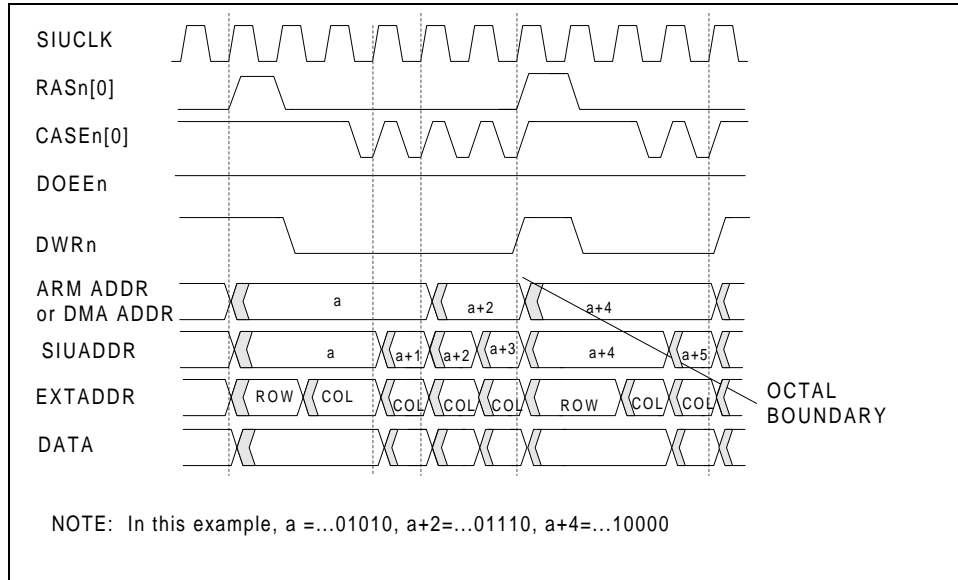
Figure 4-18. 8-bit Memory Data Bus

The timing diagram illustrates an 8-bit memory data bus, a burst of halfword transfers (3 halfwords) from the ARM7 or DMA, for ½ cycle CASn and PG = zero wait states (non-interleaved). It also illustrates the octal halfword boundary that will cause the SIU to terminate the burst and cause the DRAM Controller to regenerate the RASn precharge time. This interface speed can only be used at slow frequencies.
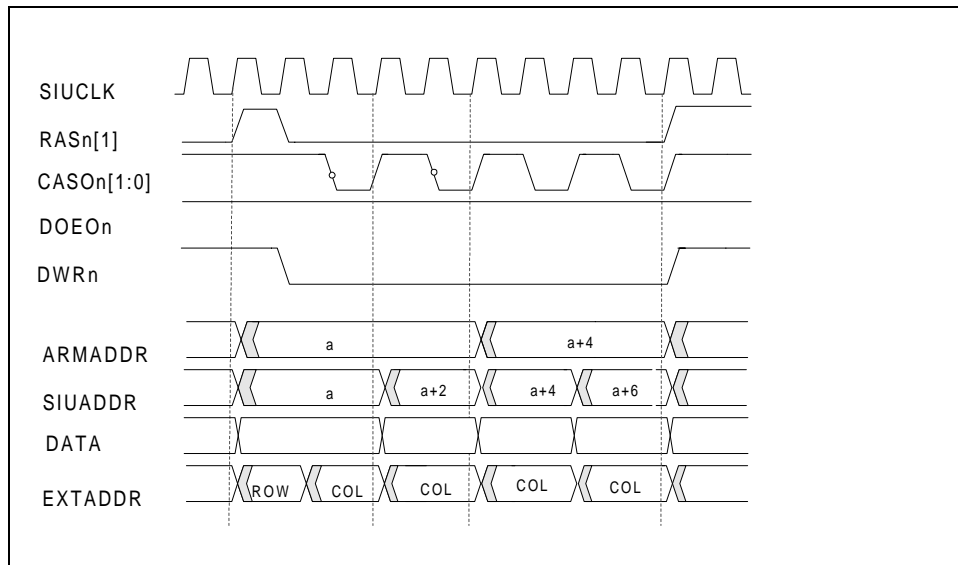
Figure 4-19. 16-bit Memory Data Bus

The timing diagram illustrates a 16-bit memory data bus, a burst of word transfers (2 words) from the ARM7 for full clock width CASn and PG = one wait state (non-interleaved). It also illustrates that the octal halfword boundary doesn't occur in the middle of the burst of word transfers.
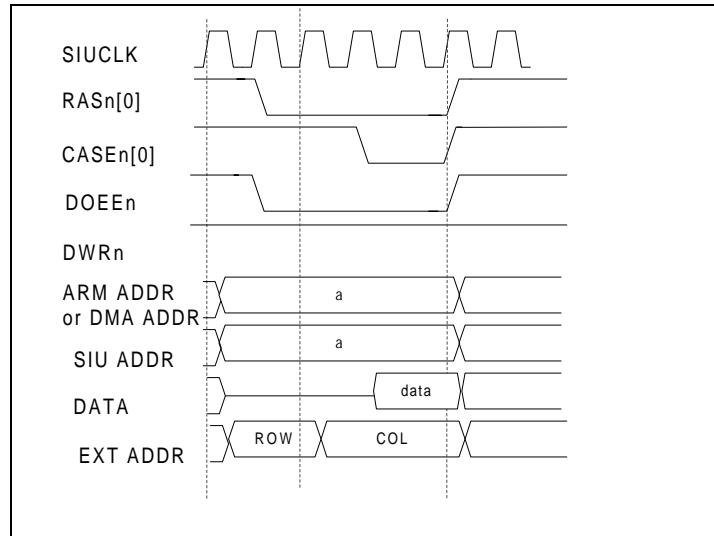
Figure 4-20. CASn Non-Interleaved 8-bit DRAM Read

The timing diagram illustrates a two clock cycle CASn non-interleaved 8-bit DRAM read (Non burst mode). This configuration illustrates the row/column address multiplex occurring 1 cycle after RASn.
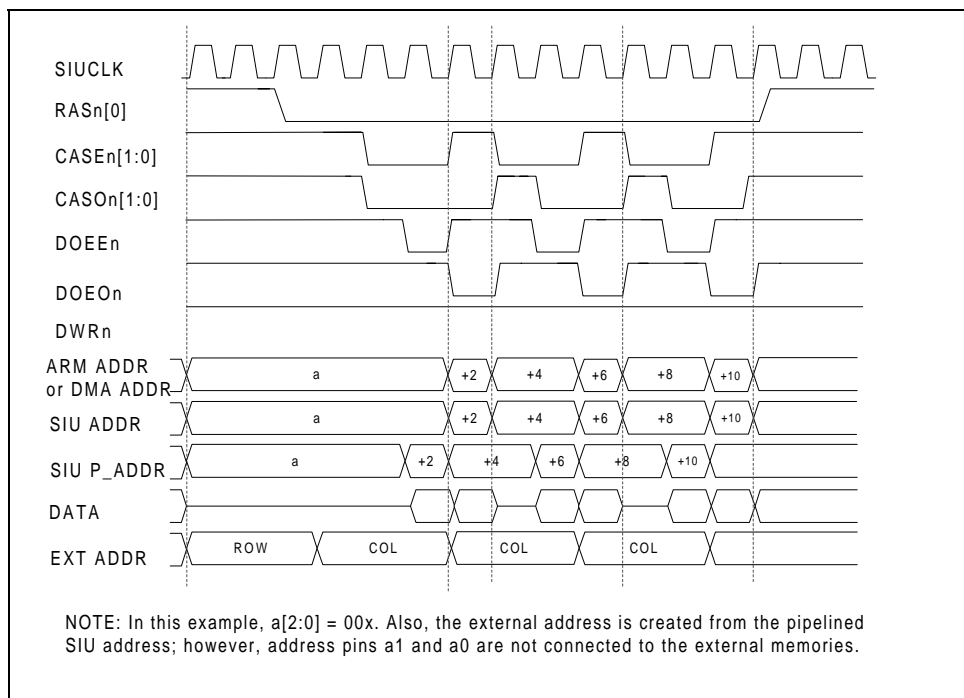
Figure 4-21. 2-Way Interleaved Memory with Halfword Bursts of Data

This example illustrates a read of two-way interleaved memory with halfword bursts of data (6 halfwords). It also illustrates that the octal halfword boundary doesn't occur in the middle of the burst of halfword transfers. It assumes external drivers to minimize the data bus contention and to insure that the data has enough setup time to CLK. This waveform also illustrates, increased RASn precharge time and increased address multiplexing time. Byte bursts of data are not allowed when a 16-bit memory structure is used.
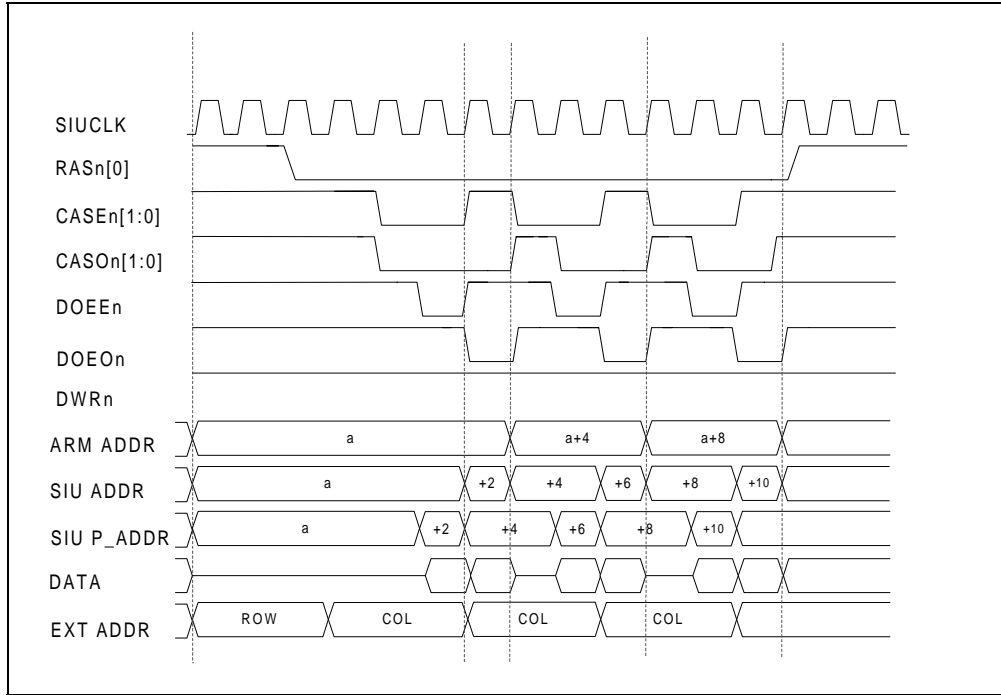
Figure 4-22. 2-Way Interleaved DRAM Read (3 words)

The timing diagram illustrates a two-way interleaved DRAM read (3 words). It assumes external drivers to minimize the data bus contention and to insure that the data has enough setup time to CLK. This waveform also illustrates, increased RASn precharge time and increased address multiplexing time.
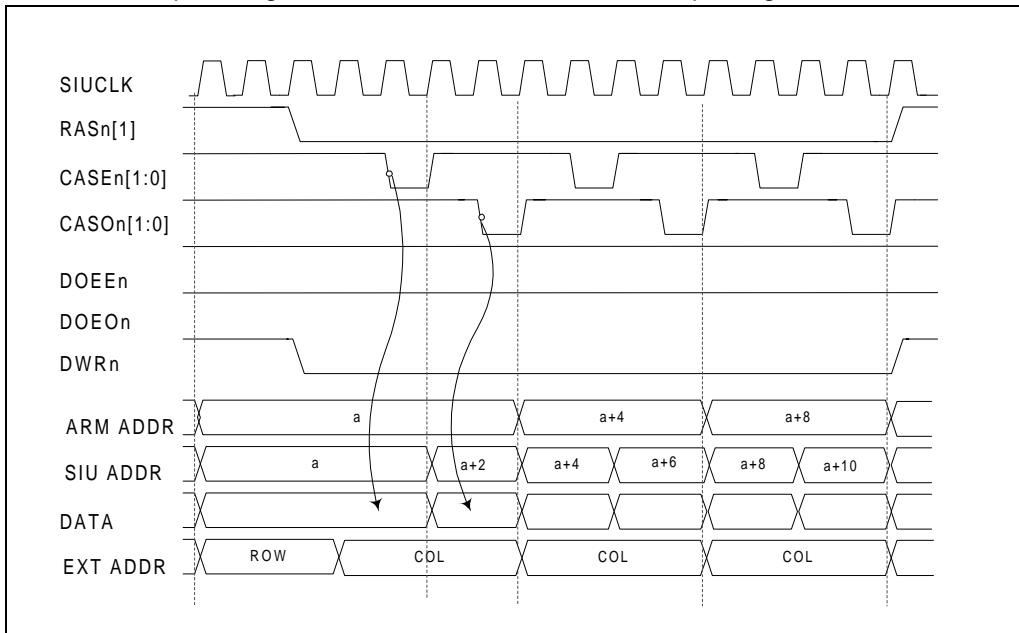
Figure 4-23. 2-Way Interleaved DRAM Write

The timing diagram illustrates two-way interleaved DRAM write. This configuration assumes that the data bus is available to the DRAM with enough setup time to the CASn falling edge. In addition, this configuration has a 2-cycle wide RASn and allows one cycle after the falling edge of RASn before the row/column address mux.
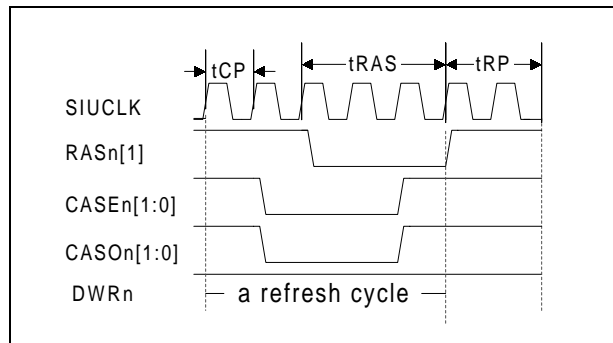
Figure 4-24. Refresh Cycle

The timing diagram illustrates a refresh cycle. t_{RAS} is three cycles wide to accommodate frequency ranges up to 40 MHz. This timing is used during prime power refresh. During battery backup, a custom refresh circuit is used to generate refreshed timing based on the oscillator clock.

4.5.4 Detailed Timing Measurements
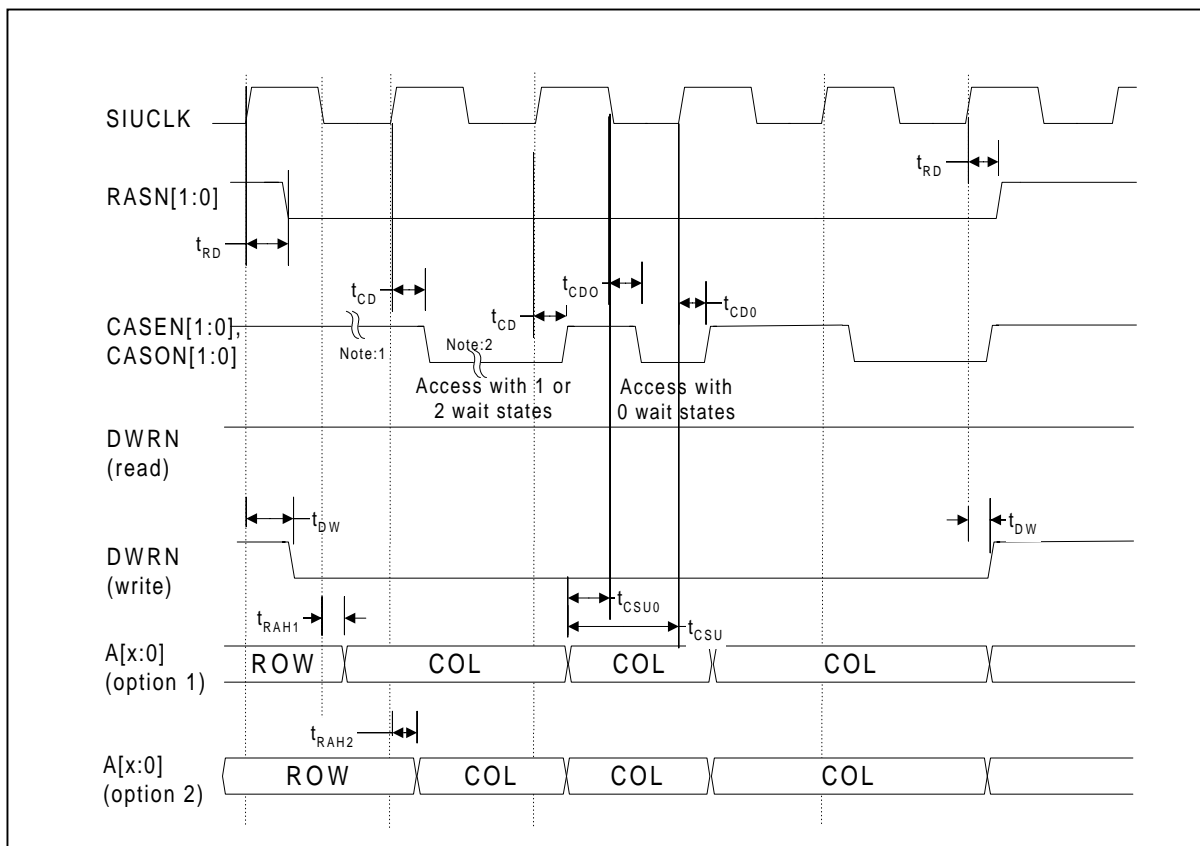
Figure 4-25. DRAM Timing—Read, Write and Wait States for Non-interleave Mode
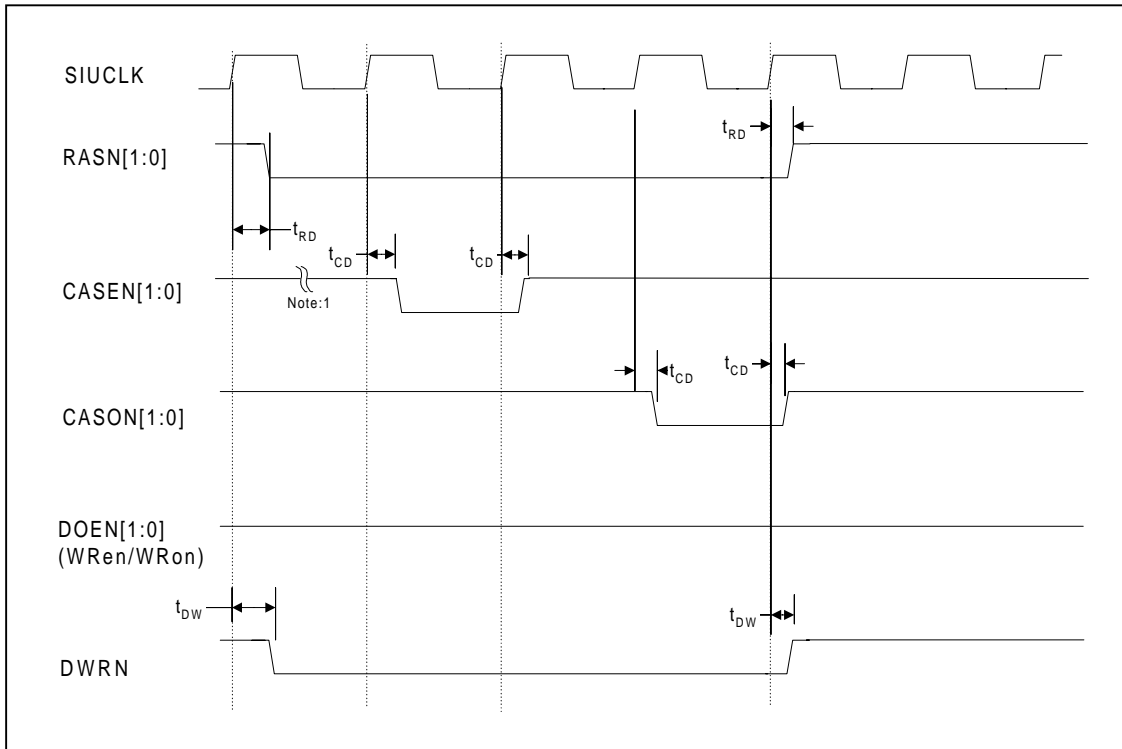
Figure 4-26. DRAM Timing for 2-way Interleave Write
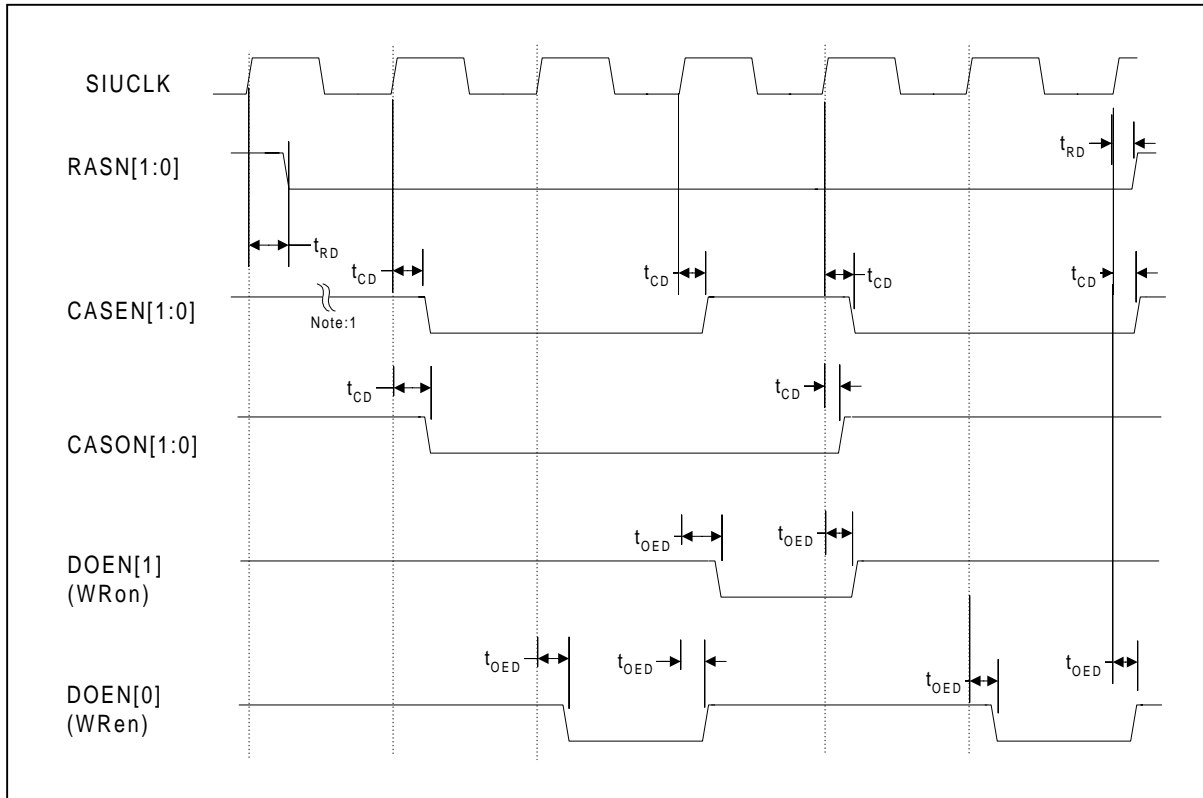
Figure 4-27. DRAM Timing—Read for 2-way interleave mode
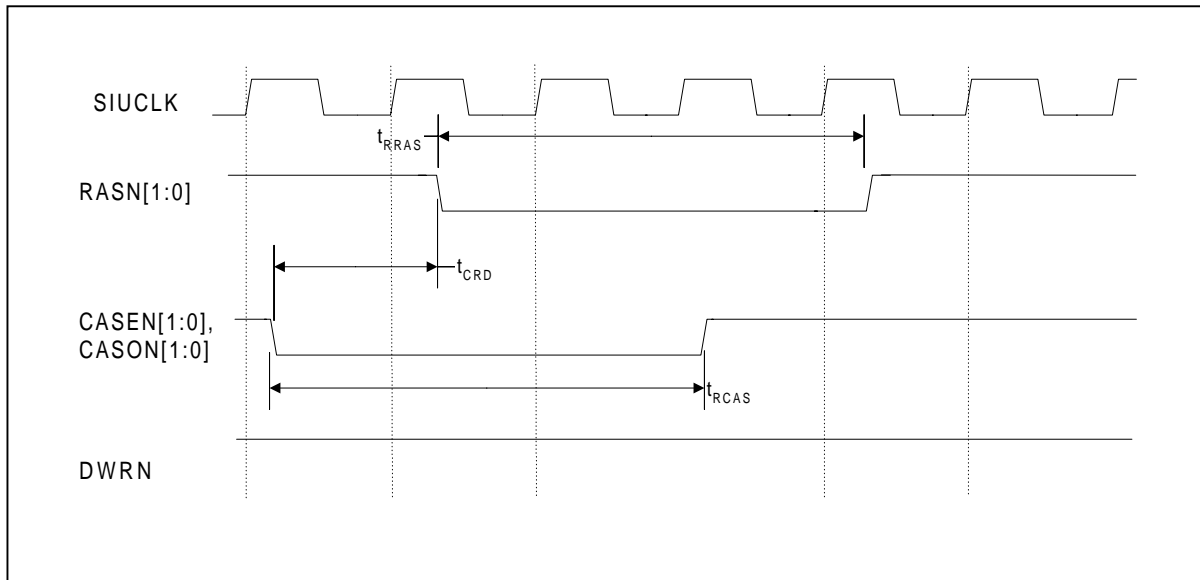
Figure 4-28. DRAM Refresh Timing
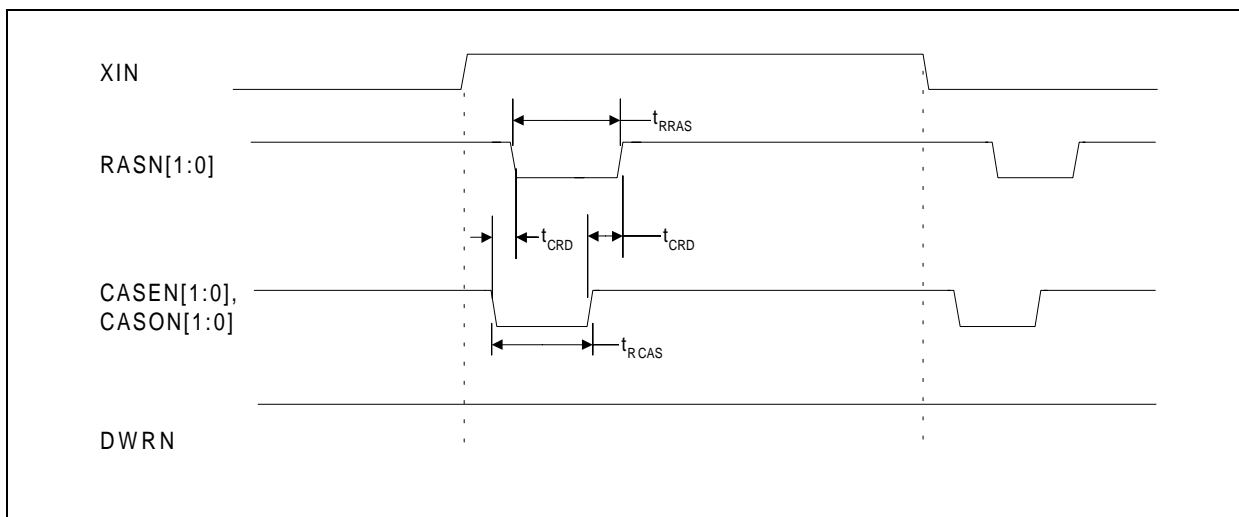
Figure 4-29. DRAM Battery Refresh Timing

Table 4-15. DRAM Timing Parameters

Parameter	Symbol	Min.	Max.	Units
RASN delay	t_{RD}		20	ns
CASN delay (0 wait state)	t_{CD0}		20	ns
CASN delay	t_{CD}		20	ns
CASN address setup (0 wait state) (10Mhz)	t_{CSU0}	7		ns
CASN address setup (30 MHz)	t_{CSU}	7		ns
RASN address hold 1	t_{RAH1}	3		ns
RASN address hold 2	t_{RAH2}	3		ns
DWRN delay	t_{DW}		20	ns
OE delay	t_{OED}		20	ns
CASN pulse width	t_{RCAS}	30	500	ns
RASN pulse width	t_{RRAS}	30	500	ns
CASN to RASN delay	t_{CRD}	5	200	ns

> *Notes*: 1. *Bit 2 of the DRAM Control Register sets this time to ½ or 1 clock cycle.*
> 2. *Bit 8 and 9 of the DRAM Control Register sets this time to 1 or 2 clock cycles.*

4.5.5   Software Operation

Special Cautions

- The Backup Configuration register is set to 0000h the initial power up only. This register is backed up during battery power mode.
- DRAM control signals are tri-stated during battery power if the backup timer is set to 0 or the backup time expires, regardless of how the enable bits are set.
- After a reset occurs and the CPU is released to run, the CPU cannot write to the Backup Configuration register until one OSCCLK cycle has passed.
- During power-up, the firmware must pay attention to the power up DRAM requirements specified in the DRAM data books. Typically, DRAMs cannot be accessed for 200μs and then must have 8 CASn before RASn refreshes occur before accessing the DRAMs.
- Within the DRAM Memory address space memory bank mirroring is allowed. Firmware will have to determine the memory size.

4.6   Flash Memory Controller

The Flash memory controller provides an interface for both NAND and NOR type flash memory devices. Up to two NOR type Flash memory devices or two NAND type devices can be connected to the ASIC. FCS0n and FCS1n pins are the flash memory control pins. FWRn and FRDn are the flash read and write signals for NAND type devices. Additional pins are required to interface to NAND type flash memory devices. GPIO pins can be used as NAND type flash memory control signals.

Figure 4-30 shows the structure of the flash controller. It consists of an address and chip select generator block and a read/write control block. The address and chip select generator block provides up to 21 address lines and 2 chip select signals for the flash memory devices. The read/write control block provides read and write strobes for NAND type devices.

4.6.1   Supported Flash Memory

The flash memory controller supports the following types of Flash memory and their equivalents:

Flash Memory		Size	Type
INTEL	28F400BL/28F004BL-150	512 kB	NOR type
	28F400BX/28F004BX-80, -120	512 kB	NOR type
AMD	Am29F040-70, -90, -120, -150	512 kB	NOR type
Samsung	KM29N040/080-150	512 kB/1 MB	NAND type

> **Note:**   This ASIC also supports 1MB and 2MB NOR-type flash memory if the specified timing can be matched.

4.6.2   Functional Description

4.6.2.1   Interfacing Flash Memory

The two types of flash memory that are supported require separate interface control options. NOR type devices are bus oriented and can be connected to the CPU bus. The ASIC provides address signals to access the memory space and provides the chip select signals. NAND type devices are special purpose peripherals with a specialized interface requiring no address bus signals. The ASIC will use dedicated pins and GPIO pins to interface with the NAND flash control and status lines.

4.6.2.2   NOR Type Flash Memory

NOR type devices can be used for both firmware code memory and for data memory. Accesses are performed using normal bus operations. Reading data is performed with a bus read. Programming bytes or erasing sectors requires multiple bus cycles. The CPU writes the command sequences required by the flash memory; then it polls the flash memory's status until the operation is complete.

The data address space is available in two separate blocks; the first block of memory is from 00800000h to 009FFFFFh (2Mbyte block) and the other is from 00A00000h to 00BFFFFFh (2Mbyte block). When the CPU accesses the address range 00800000h - 009FFFFFh, FCS0n is activated. When the CPU accesses the address range 00A00000h - 00BFFFFFh, FCS1n is activated. When using FCS0n and FCS1n for NOR-type flash memory, the NAND-type bit (bit 6) of the FlashCtrl register must be set to 0. The FlashCtrl register is described in the SIU section of the Hardware Description.

The NOR type flash interface consists of:

- **FCSn[1:0]:** The two flash device selection signals. To use FCSn[1:0], bit 6 of the FlashCtrl register must be set to 0.
- **A[20:0]:** The external address bus for 2MB address range.
- **RDn and WREn/WROn:** The external bus read and write strobes.
- **D[15:0]:** The external data bus.

4.6.2.3   NAND-Type Flash Memory

NAND type devices can only be used for data memory. NAND type flash devices have no address bus. Command, address and data are passed through the external data bus. Accesses are accomplished by first setting the appropriate flash CS and control signals through the FCSn[1:0] and GPIO pins. The actual bus transfer is performed by accessing the *NANDFLSH* register. When accessing the *NANDFLSH* register, the appropriate flash data strobe is activated, FRDn for read operations and FWRn for write. FRDn and FWRn are also controlled by the wait state setting in the *FlashCtrl* register. The NAND-type bit in the *FlashCtrl* register must be 1.

- **FCSn[1:0]:** The device selection signals for the NAND type flash devices. FCS0n and FCS1n pins act as GPO pins. FCSn[1:0] pins output values which are set to bit 8 and bit 9 of the FlashCtrl register.

> **Note:**   If FCS0n/PWM[1] and FCS1n/PWM[2] pins are programmed as PWM, any two available GPO's or GPIO's can be used to generate FCS[1:0]n for NAND-type flash memory.

- **FCLE:** The Flash Command Latch Enable. It is programmable via a GPIO pin. FCLE activates the commands sent to the command register.
- **FALE:** The Flash Address Latch Enable. It is programmable via a GPIO pin. FALE activates the controls for address or data to the internal address or data registers of the flash device.
- **FRDY:** The Flash Ready signal. It can be read from a GPIO pin. FRDY indicates the status of the flash device operation.
- **FRDn:** The Flash Read Enable signal. It is a hardware generated signal. It is multiplexed on GPIO[1] and configured in the GPIOConfig register. FRDn enables the data from the flash memory device onto its I/O bus. This signal is active when the CPU reads the NAND flash location of 01FF8824h.
- **FWRn:** The Flash Write Enable signal. It is a hardware generated signal. It is multiplexed on GPIO[0] and configured in the GPIOConfig register. FWRn controls writes to the I/O bus. This signal is active when the CPU writes to the NAND flash location of 01FF8824h.
- **WRPROTn:** The write protect signal is driven low to protect the flash devices from inadvertent writes during power transitions. It is also controlled by writing to the 'Prime Power write protect' bit (bit 11) of the FlashCtrl register.
- **D[15:0]:** The external data bus.
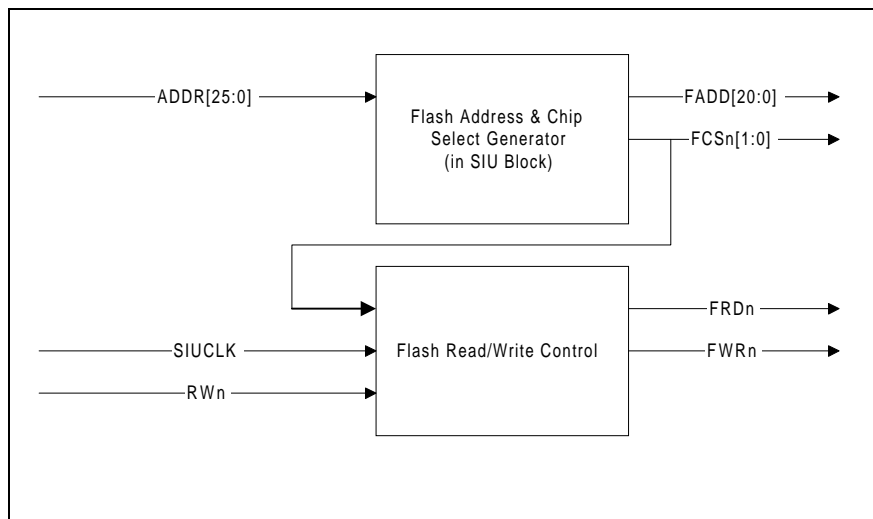
Figure 4-30. Flash Control Block Diagram

4.6.3   Timing

All accesses can be performed from zero to seven wait states. The number of wait states is programmable in the FlashCtrl register. During write operations, if one to seven wait states is used, the EarlyOff option must be set in the FlashCtrl register. The EarlyOff option is ignored with zero wait state.

Wait states are generated to accommodate slow memory devices, for more details of how the control signals are changed with an inserted wait state, refer to the timing diagram Figure 4-31.

- **NAND-Type flash** For this type of flash, addresses and data are passed through the external data bus. During read operations, the device selection is one of the two Flash Chip Select signals FCSn[1:0], FRDn is the Flash Read Enable signal from the bus interface. FCLE and FALE must be low (inactive) during reads. For zero wait state access, FWRn and FRDn are only half SIUCLK cycle (like RDn and WRn of the normal bus operation).

    During program/erase operations, the device selection is one of the FCSn[1:0] signals. Command, address and data are all written through the external data bus. The Flash Command Latch Enable (FCLE) and the Flash Address Latch Enable (FALE) signals are the controls for writing to the Command or the Address register respectively. FCLE and FALE must be low (inactive) during a data write to the flash devices.

- **NOR-Type Flash** Accesses are performed using normal bus operations.

    During program/erase operations, the device selections are the two Flash Chip Select signals FCSn[1:0], addresses are latched on the falling edge of the Flash Write Enable signal WREn/WROn, and data is latched on the rising edge of WREn/WROn.
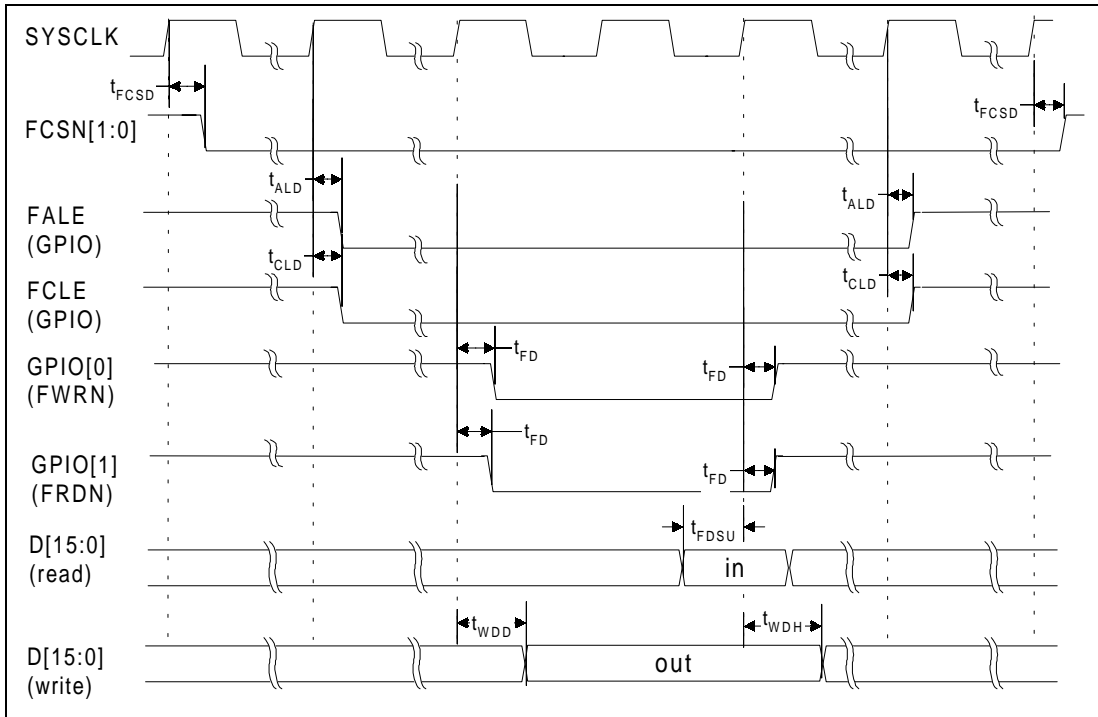
Figure 4-31. NAND-Type Flash Memory Access with Two Wait States

Parameter	Symbol	Min.	Max.	Units
FCSN delay	t_{FCSD}		16	ns
Flash read/write delay	t_{FD}		16	ns
Flash read data setup	t_{FDSU}	8		ns
Flash write data delay	t_{WDD}		21	ns
Flash write data hold	t_{WDH}	0		ns
Flash address latch delay	t_{ALD}		16	ns
Flash command latch delay	t_{CLD}		16	ns

4.6.4 Register Description

Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
NANDFlash *(NANDFLSH)* **00C00001h**	(Not Used)								Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
NANDFlash *(NANDFLSH)* **00C00000h**	Reading this location activates the FRDn signal. Writing to this location activates the FWRn signal. No physical register exists for this location.								Rst. Value xxh Read Value 00h

Register description: This register is the IO address space for the NAND type flash. Reading this register activates the FRDn signal. Writing to this register activates the FWRn signal. FCLE, FALE, and FRDY must be setup through the GPIO pins and FCSn[1:0] must be setup through the FlashCtrl register prior to accessing the NANDFlash register. FCLE, FALE, and FRDY must be cleared through the GPIO pins and FCSn[1:0] must be cleared through the FlashCtrl register after the register access.

Data written to this register is output on the data bus. When reading this register the flash memory data is placed on the internal CPU bus. The *NANDFlash* register is only an address location; no register actually exists.

4.7 DMA Controller

4.7.1 General Description

- The AMFPC is equipped with thirteen physically arbitrated DMA Channels assigned to either internal or external requests.
- Each channel provides its own 26-bit address for data access
- The DMA channel address registers are made up of two programmable half word read/write registers, making up a 26-bit counter (64MB).
- Minimum DMA acknowledge delay is 2 System Clocks.
- Maximum DMA acknowledge delay depends on the number of pending higher priority DMA requests and the length of associated bus cycles including wait states and halt states due to DMA/DRAM refresh cycle collisions.

- Channel Specific Unique Features

  ⇒ **Double Buffered Address and Block Size**

  This channel is equipped with a double buffered DMA address counter and Block Size register. This allows firmware to set up the DMA address and Block Size values for the next block access while the current one is active. When the DMA channel reaches it's Block Size limit it issues an interrupt, and if a new value has been written into the buffers (Buffer Loaded Flag is set), this value is transferred to the address counter and/or Block Size register. The interrupt is cleared upon writing to the Block Size Buffer register.

  ⇒ **DMA Block Size**

  An Access Block Size Counter is available to limit the number of DMA access in a given block. Once this counter is set, it will keep track of the number of DMA access. Once the limit has been reached a CPU interrupt will be set and no further accesses will be allowed until the register is reset. The IRQDMA is activated at the falling edge of DMAACK.

⇒ **Addressing Modes**

All of the DMA Channels increment by 2 after each access. The external channels 1 and 2, can be set to increment by 1 for byte wide peripherals. The count enable register allows the user to select which address will increment after each access. However, Channels 2 and 3 have the additional functionality allowing them to Increment or Decrement by a specified amount. Channel 2 is controlled by the values in it's control register. Channel 3 is controlled by control signals generated in the Bit Rotation Logic.

⇒ **Priority**

Shows the overall channel priority in the DMA request arbitrator, with 1 as the highest.

⇒ **Uninterrupted Burst mode**

Allows the requesting device to hold the Burst DMA Request active (PSEQ), to transfer multiple bytes of uninterrupted data. If higher priority requests occur during the burst operation they will be ignored.

This operation is mainly for high speed DRAM access. Also during this mode, the read/write control and the count controls must remain static.

⇒ **Throttle Control**

A Throttle Value can be set to allow 1 DMA access per a given time period. The throttle time period is a product of the value set and the system clock.

⇒ **Logical Channels**

This single physical DMA channel is equipped with multiple logical channels that operate independent of each other. Only one logical can be active at any time.

4.7.1   DMA Mode Summary

Table 4-16. Feature Matrix

Channel	Features							
	Double Buffered Control	Block Size Limit	Address Increment	Address Decrement	Address Jump	Priority	U-I Burst Mode	Other Features
0	√	√	√	√		1	√	Logical
1			√	√		2	√	
2	√	√	√	√	√	3		Byte
3			√	√	√	4		
4		√	√	√		7	√	
5		√	√	√		8	√	
6			√	√		9	√	Throttle
7			√	√		10	√	
8			√	√		11	√	
9		√	√	√		12	√	
10		√	√	√		13	√	Throttle
11		√	√	√		5	√	
12		√	√	√		6	√	Throttle

Each of the DMA channels, its function, and its characteristics are provided in Table 4-17.

Table 4-17. DMA Channel Functions and Characteristics

DMA Channel	Function	Characteristics
0[1]	USB data to/from memory for PC print, PC fax TX, PC Fax RX, PC scan	DMA Block Size Limit with CPU Interrupt via PIO/USB Interrupt, 4 logical DMA Channels, DMA address and Block Size double buffered, each Read/Write selectable, halfword DMA data access only.
1	External DMA access only.	Normal and delayed ACK, Read/Write selectable, byte and halfword DMA data access selectable.
2	External DMA Request only Bit Rotation Access (Output)	Address Jump Control, DMA Block Size Limit with Interrupt to IRQ Controller, Burst Mode, Double Buffered Control Registers Read/Write, External Request (Generates a DisDrive to disable the Bus IF output drivers to for external memory writes.) External Request to the DMA controller may be delayed from 1 to 2 clocks for synchronization.
3	Bit Rotation Access (Input)	Address Jump Control (Controlled by Bit Rotation Block see table4.8.2-1) Read Only, Internal Request. (MAS is always set to ½ word)
4	Countach to ARM Memory (c2a)	Data write only, halfword DMA data access only
5	ARM Memory to Countach (a2c)	Data read only, DMA Block Size Limit (16 kB), Interrupt to IRQ Controller halfword DMA data access only.
6	Read/Write T4 Uncoded data from/to Line Buffer to/from T4 Logic	Throttle Control Read/Write, Internal Request (MAS is always set to ½ word)
7	Read T.4 Reference Line from the Line Buffer to T.4 logic	Read Only, Internal Request (MAS is always set to ½ word)
8	T4 Resolution Converted data, Line Buffer Access	Disable Address Count for read modify write Read/Write, Internal Request, Read Modify Write Control (MAS is always set to ½ word)
9	Bi-level resolution conversion logic	DMA Block Size Limit. Interrupt to IRQ Controller (MAS is always set to ½ word)
10	Read/Write T4 Coded data from/to Page Memory to/from T4 Logic	Throttle control Address Block Size Limit, Interrupt to IRQ Controller Read/Write, Internal Request (MAS is always set to ½ word)
11	P1284 to Memory	DMA Block Size Limit with CPU Interrupt via PIO Interrupt, Read/Write (Control Bit), Internal Request
12	Memory to P1284	DMA Block Size Limit with CPU Interrupt via PIO Interrupt, Read/Write (Control Bit), Internal Request (MAS is always set to ½ word)
Notes:	1.	DMA Channel 0 has the highest priority.
	2.	"AD" means address.

4.7.2   DMA Operation and Timing

The DMA controller arbitrates DMA requests from all sources (internal and external), and then acknowledges the request of the source with the highest priority at the start of the next bus cycle. After each request has been issued, the associated acknowledge signal is activated with a minimum delay of 2 internal clock cycles. The maximum delay is dependent on both the number of pending higher priority DMA requests, and the length of the associated bus cycles (including wait states and halt states due to DMA to DRAM and refresh cycle collisions).

The DMA controller informs the bus control logic (SIU), and provides the address and read/write signal prior to the next bus cycle. At the time of the next bus cycle, the SIU routes the address and data onto the proper bus. If the external bus is required to complete the DMA transfer, the DMA controller notifies the external bus control logic, which in turn halts the CPU. After the completion of a DMA cycle, the DMA controller increments or decrements the DMA address counter.

A non-maskable interrupt (SYSIRQ) to the CPU allows the active DMA to be completed, but locks-out all other DMA acknowledge signals in the event of a power down condition.

DMA Channel 3 address progression is controlled by the Bit Rotation logic and is not effected by the DMA acknowledges like other channels. The Bit Rotation Logic provides an address progression value, an increment/ decrement (add/subtract) control and a count strobe. When the strobe is held active during the rising edge of the siuclk, the progression value is either added or subtracted from the current address value. Below is a table showing the control bit assignments.

Table 4-18 DMA Channel 3 Control Bit Assignment

Count Control Assignments		
countcntl(11)	Count Strobe	NA
countcntl(10)	Dec_Incn	NA
countcntl(9)	Cnt_By_32k	Add_value(15)
countcntl(8)	Cnt_By_16k	Add_value(14)
countcntl(7)	Cnt_By_8192	Add_value(13)
countcntl(6)	Cnt_By_4096	Add_value(12)
countcntl(5)	Cnt_By_2048	Add_value(11)
countcntl(4)	Cnt_By_1024	Add_value(10)
countcntl(3)	Cnt_By_512	Add_value(9)
countcntl(2)	Cnt_By_4	Add_value(2)
countcntl(1)	Cnt_By_2	Add_value(1)
countcntl(0)	Cnt_By_1	Add_value(0)

4.7.3   Timing

4.7.3.1   Internal DMA Requests

DMA requests are sourced by sub-blocks or peripherals within the ASIC or by an external device (Ch 0 or 1). Some logic blocks may source multiple DMA request lines. For example, the T4 Logic requires 3 request lines: input data, reference data and output data. These sources will issue DMA request signals, and the sequential access indicator, synchronized to the rising edge of *SIUCLK*. During the bus cycle that the requests are received, the controller will inform the SIU that the next bus cycle will be a DMA cycle. The DMA controller also provides the address, the *sequencial access*, and read/write control information to the SIU. The SIU will then activate the DMA bus acknowledge signal when the DMA controller becomes the bus master. On the first rising edge of the SIUCLK after the DMA bus acknowledge signal becomes active, the peripheral DMA acknowledge to the requesting peripheral will be set high indicating the start of the DMA cycle. For a DMA single cycle write, the requesting device must drive data onto the bus during activation of the peripheral DMA acknowledge signal. And for a single

cycle DMA read the requesting device must capture the data on the falling edge of the peripheral DMA acknowledge.

If a sequential DMA access burst is desired, the peripheral must also activate the sequential access indicator at the time of the initial request and must be cleared at the end of the next to the last DMA cycle. The DMA request must be deactivated during the final DMA cycle, on the 1st rising edge of the SIUCLK. The peripheral must also capture (read), or drive the next datum onto the data bus on each rising edge of the SIUCLK when the DMA acknowledge is active. During sequential burst accesses the Read/Write control and DMA address count controls must remain static.

4.7.3.2　External DMA requests

The external device activates the DMA request signal (DMAREQ) at any time when it wants to do the single DMA access operation. The MFC2000 chip double synchronizes the DMAREQ signal internally to avoid the meta-stable case. After external DMA request has been issued, the associated acknowledge signal is activated with a minimum delay of 5 internal SIUCLK cycles. The external device must deactivate DMAREQ after the associated acknowledge signal is activated and before the end of this DMA cycle. If the DMA request continues to be activated by the external DMA requesting device, the external DMA requesting device will get the second DMAACK signal for the next single data transfer, when system bus is ready.

The MFC2000 chip continues to perform single DMA access as long as the DMAREQ is kept active by the external DMA requesting device.

DMAACK is used similarly as chip select to indicate the DMA access cycle is ready for the external DMA requesting device. The delay path within the MFC2000 is longer for DMAACK than for RDn; however, the actual amount of skew between the two signals is dependent on their relative loading at the system board level. If RDn has excessive delay, the DMAACK can be programmed to extend it an extra half SIUCLK cycle.
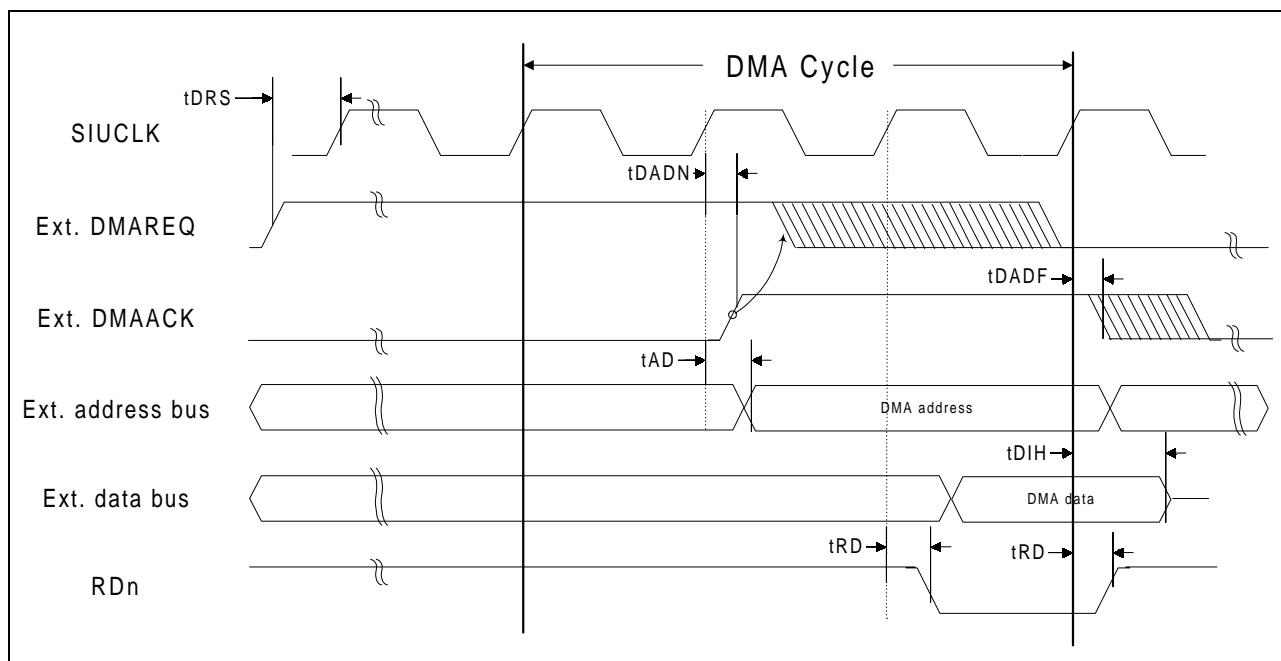
Figure 4-32: External DMA Read Timing (Single Access, One Wait State)
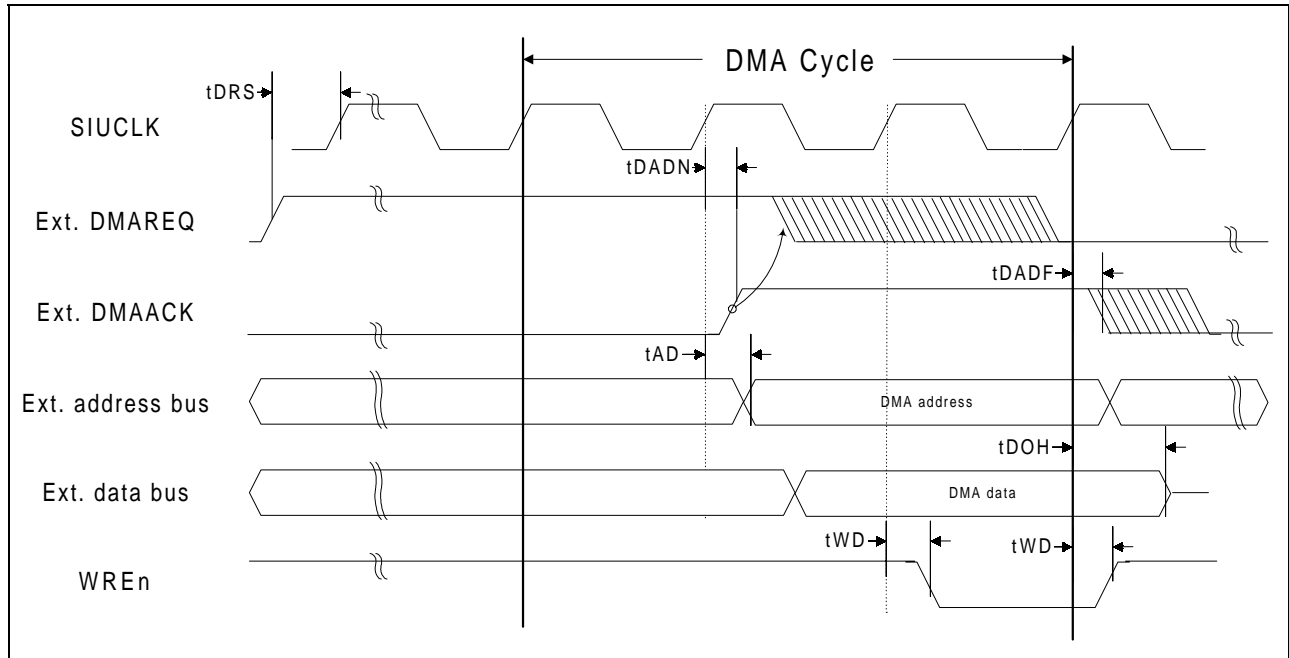
Figure 4-33. External DMA Write Timing (Single Access, One Wait State)

Parameter	Symbol	Min	Max	Units
RDn delay (ext. load = 50pF)	tRD	—	25	ns
WRn delay (ext. load = 50pF)	tWD	—	25	ns
Address delay (ext. load = 50pF)	tAD	—	24	ns
DMAACK on delay time (ext. load = 15pF)	tDADN	—	10	ns
DMAACK off delay time (ext. load = 15pF)	tDADF	—	5	ns
DMA Input Data Hold	tDIH	8	—	ns
DMA Output Data Hold (ext. load = 50pF)	tDOH	8	25	ns
RASn delay (ext. load = 40pF)	tRASD	—	25	ns
CASn delay (ext. load = 40pF)	tCASD	—	20	ns
DWRn delay (ext. load = 40pF)	tDWD	—	20	ns

Notes:

1. *SIUCLK is the internal system interface clock. These values is for SIUCLK = 30 MHz.*

2. *The external DMA request set-up time (tDRS) is not required because this input is double synchronized internally for the meta-stable case.*

4.7.4 USB Block Diagrams

Below is a block diagram depicting the four USB logical channels. Each logical channel shows the double buffering of the address and block registers. The right side of the diagram shows the common counters for the block and address. Also shown is the feedback path from the counters to the registers used upon USB ACKs. The select signal for the logical channels are shown at the bottom of the diagram.
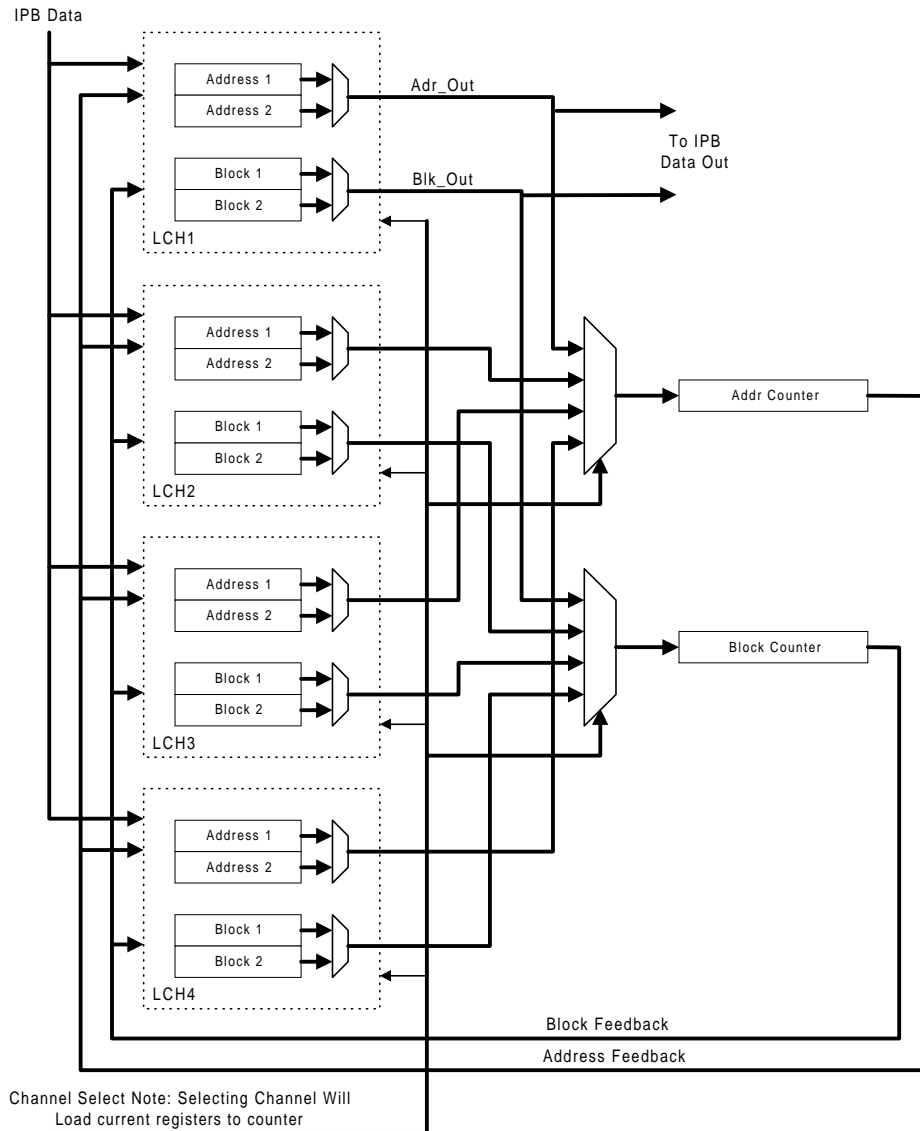
Figure 4-34. USB Logical Channels Block Diagram

4.7.4.1   USB Logical Channel Assignments

- Ch1: Scan   Mem to PC, DMA Read only
- Ch2: Print   PC to Mem, DMA Write only
- Ch3: FAX Rx   Mem to PC, DMA Read only
- Ch4: FAX Tx   PC to Mem, DMA Write only

4.7.5   DMA Controller Registers USB Logical Channel Assignments

DMA Channels 1—12 addressing is controlled by the following registers.

> **Note**: If the DMA address counter points to an invalid location, invalid data is read or written

Address: chcsl[10:0]	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Low Counter *(DMAiCntlo)*	DMA0-12 Addr. Bit 15	DMA0-12 Addr. Bit 14	DMA0-12 Addr. Bit 13	DMA0-12 Addr. Bit 12	DMA0-12 Addr. Bit 11	DMA0-12 Addr. Bit 10	DMA0-12 Addr. Bit 9	DMA0-12 Addr. Bit 8	Rst. Value 00h Read Value 00h
Address: chcsl[10:0]	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
DMA Low Counter *(DMAiCntlo)*	DMA0-12 Addr. Bit 7	DMA0-12 Addr. Bit 6	DMA0-12 Addr. Bit 5	DMA0-12 Addr. Bit 4	DMA0-12 Addr. Bit 3	DMA0-12 Addr. Bit 2	DMA0-12 Addr. Bit 1	DMA0-12 Addr. Bit 0	Rst. Value 00h Read Value 00h

DMA Channel 0—12 Lower Address Counter Value

DMAUSB0CntLo	01FF81C8-C9
DMAUSB1CntLo	01FF81CE-CF
DMAUSB2CntLo	01FF81D4-D5
DMAUSB3CntLo	01FF81DA-DB
DMA1CntLo	01FF8184-85
DMA2CntLo	01FF8188-89
DMA3CntLo	01FF8190-91
DMA4CntLo	01FF8194-95
DMA5CntLo	01FF8198-99
DMA6CntLo	01FF819C-9D
DMA7CntLo	01FF81A0-A1
DMA8CntLo	01FF81A4-A5
DMA9CntLo	01FF81A8-A9
DMA10CntLo	01FF81AC-AD
DMA11CntLo	01FF81E0-E1
DMA12CntLo	01FF81E6-E7

Address: chcsh[10:0]	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Hi Counter *(DMAiCnthi)*	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	DMA0-12 Addr. Bit 25	DMA0-12 Addr. Bit 24	Rst. Value 00h Read Value 00h
Address:chcsh[10:0]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA Hi Counter *(DMAiCnthi)*	DMA0-12 Addr. Bit 23	DMA0-12 Addr. Bit 22	DMA0-12 Addr. Bit 21	DMA0-12 Addr. Bit 20	DMA0-12 Addr. Bit 19	DMA0-12 Addr. Bit 18	DMA0-12 Addr. Bit 17	DMA0-12 Addr. Bit 16	Rst. Value 00h Read Value 00h

DMA Channel 0—12 Upper Address Counter Value

DMAUSB0CntHi	01FF81CA-CB
DMAUSB1CntHi	01FF81D0-D1
DMAUSB2CntHi	01FF81D6-D7
DMAUSB3CntHi	01FF81DC-DD
DMA1CntHi	01FF8186-87
DMA2CntHi	01FF818A-8B
DMA3CntHi	01FF8192-93
DMA4CntHi	01FF8196-97
DMA5CntHi	01FF819A-9B
DMA6CntHi	01FF819E-9F
DMA7CntHi	01FF81A2-A3
DMA8CntHi	01FF81A6-A7
DMA9CntHi	01FF81AA-AB
DMA10CntHi	01FF81AE-AF
DMA11CntHi	01FF81E2-E3
DMA12CntHi	01FF81E8-E9

Address:ch0cscntl	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA 0 Configuration *(DMA0config)* **$xx81B1**	Channel Enable LC3	Channel Enable LC2	Channel Enable LC1	Channel Enable LC0	Read/Write Mode LC3	Read/Write Mode LC2	Read/Write Mode LC1	Read/Write Mode LC0	Rst. Value 00h Read Value 00h

Address:ch0cscntl	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA 0 Configuration *(DMA0config)* **$xx81B0**	Not Used	Not Used	IRQ LC3	IRQ LC3	IRQ LC3	IRQ LC3	Not Used	DMA0 Enable	Rst. Value 00h Read Value 00h

Bit 15-12: Logical Channel Enable   Setting the bit to 1 will allow operation of the logical channel, setting the bit to 0 will disable the channel and clear it's registers.

Bit 11-8: Read/Write Mode Select.   Selects the data direction. Read = 1, Write = 0

Bit 5-2:   Interrupts from the individual Logical Channels. Clear by writing to the block size register.

Bit 0:   Setting this bit to 1 will enable DMA Channel 0. Setting this bit to 0 will fore all of the logical channels into their reset state.

Address: chcsl[10:0]	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMAUSB Low Counter *(DMAiCntlo)*	DMA0-10 Addr. Bit 15	DMA0-10 Addr. Bit 14	DMA0-10 Addr. Bit 13	DMA0-10 Addr. Bit 12	DMA0-10 Addr. Bit 11	DMA0-10 Addr. Bit 10	DMA0-10 Addr. Bit 9	DMA0-10 Addr. Bit 8	Rst. Value 00h Read Value 00h

Address: chcsl[10:0]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMAUSB Low Counter *(DMAiCntlo)*	DMA0-10 Addr. Bit 7	DMA0-10 Addr. Bit 6	DMA0-10 Addr. Bit 5	DMA0-10 Addr. Bit 4	DMA0-10 Addr. Bit 3	DMA0-10 Addr. Bit 2	DMA0-10 Addr. Bit 1	DMA0-10 Addr. Bit 0	Rst. Value 00h Read Value 00h

- DMAUSB Channel 0-3 Lower Address Counter Value. Address will only count by 2.
- Reading this register will return the active Address value when the last USBACK occurred.

> *Note: This register is double buffered. Writing to this location twice will load the active register as well as the buffer register. Upon a block limit interrupt, the buffer value will be activated therefor a new buffered value should be written to this register.*

DMAUSB0CntLo	01FF81C8-C9
DMAUSB1CntLo	01FF81CE-CF
DMAUSB2CntLo	01FF81D4-D5
DMAUSB3CntLo	01FF81DA-DB

Address: chcsh[10:0]	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMAUSB Hi Counter *(DMAiCnthi)*	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Rst. Value 00h Read Value 00h
Address:chcsh[10:0]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMAUSB Hi Counter *(DMAiCnthi)*	DMA0-10 Addr. Bit 23	DMA0-10 Addr. Bit 22	DMA0-10 Addr. Bit 21	DMA0-10 Addr. Bit 20	DMA0-10 Addr. Bit 19	DMA0-10 Addr. Bit 18	DMA0-10 Addr. Bit 17	DMA0-10 Addr. Bit 16	Rst. Value 00h Read Value 00h

- DMAUSB Channel 0-3 Upper Address Counter Value. Address will only count by 2.
- Reading this register will return the active Address value when the last USBACK occurred.

> *Note: This register is double buffered. Writing to this location twice will load the active register as well as the buffer register. Upon a block limit interrupt, the buffer value will be activated therefor a new buffered value should be written to this register.*

DMAUSB0CntHi	01FF81CA-CB
DMAUSB1CntHi	01FF81D0-D1
DMAUSB2CntHi	01FF81D6-D7
DMAUSB3CntHi	01FF81DC-DD

Address:ch4csbs	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMAUSB Transfer Block Size Reg. *(DMAUSBBlockSize)*	USB Channel Enable = 1	Stop At Block	Upper six bits of the Block Size Counter						Rst. Value 00h Read Value 00h

Address:ch4csbs	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMAUSBTransfer Block Size Reg. *(DMAUSBBlockSize)*	Low Byte Value for USB Memory Block Size Counter								Rst. Value 00h Read Value 00h

Block Size data written to this register will be placed into the inactive Block Size register. This register will become active once the prior block has completed (USB IRQ). Reading this register will return the Block Size value when the last USBACK occurred. Writing to this will initialize the logical channel. Therefore, when setting up a logical channel, this register should be written to last.

> *Note:* *This register is double buffered. Writing to this location twice will load the active register as well as the buffer register. Upon a block limit interrupt, the buffer value will be activated therefor a new buffered value should be written to this register.*

- DMA USB Channel Block Size Limit Counter

DMAUSB0 BlkSiz	01FF81CC-CD
DMAUSB1 BlkSiz	01FF81D2-D3
DMAUSB2 BlkSiz	01FF81D8-D9
DMAUSB3 BlkSiz	01FF81DE-DF

Bits [13:0] DMAUSB Memory Block Size

The block size of DMA channel can range from 1 – 16383 DMA transfers.

The Block Size counter will decrement regardless of the DMA Address counter's activity. The block size register content is 0000h when the block size limit is reached.

When the block size reaches its limit, an IRQ will be generated upon the next Channel ACK. Writing To this register will clear the IRQ.

> *Bit 14 Note:* *When this bit is set, the DMA channel will disable it's self until a new block size is entered.*
>
> *If this bit is 0, when a block limit is reached the next block size will be downloaded into the counter along with the next DMA address and transfers will continue within the new block. Upon the next channel ACK, an IRQ will be generated.*

Bit 15:   This bit must be set at all times to enable the channel and is not part of the double buffering process.

Address:ch1cscntl	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA 1 Configuration *(DMA0config)* **$xx8183**	DMAACK1 Delay-off 0=normal 1=delay ½ SIUCLK cycle	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Rst. Value 00h Read Value 00h

Address:ch1cscntl	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA 1 Configuration *(DMA0config)* **$xx8182**	Read = 1 Write = 0	Enable 0 = disable 1 = enable	Not Used	Not Used	Not Used	Not Used	Not Used	Count By 0=byte 1=halfword	Rst. Value 00h Read Value 00h

Bit 0:   Sets byte or halfword data access to match external devices.

Bit 6:   This bit controls the channel enable

Bit 7:   This bit controls the external data direction

Bit 15:   Setting this bit to a 1 will extend the external DMAACK ½ SIUCLK cycle so that it may be used as a chip select.

Address:ch2cscnt	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA 2 Configuration *(DMA2config)* **$xx81B3**	DMAACK2 Delay-off 0 = normal 1 = delay half SIUCLK cycle (Write Only)	Not Used	Not Used	Not Used	Not Used	Not Used	BRB to Memory, Data Transfer (Write Only)	Byte Mode = 0 ½ Word Mode = 1	Rst. Value 00h Read Value 00h

Address:ch2cscnt	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA 2 Configuration *(DMA2config)* **$xx81B2**	Read = 1 Write = 0	Requested Device 0=Bit Rot. 1= Ext Mem	0 = Inc 1 = Dec	Count By 1024	Count By 512	Count By 256	Count By 2	Count By 1	Rst. Value 00H Read Value 00h

DMA channel 2 Address Jump, Increment, and Decrement Control. The Count By control bits allows the user to select how the address is progressed after each DMA access. Only one Count By Bit can be set at a time. If there are no Count By control bits selected the address will remain fixed.

Bit 15:   If DMAACK2 is used as a select signal (like a chip select), the extended DMAACK2 may be needed. This 'DMAACK2 delay-off' bit needs to be set to '1' for extending DMAACK2 half SIUCLK cycle.

Bit 9:   This bit controls the source of the DMA Request. If this bit is set to 0 and bit 6 is set to 0, a external support chip must be used to provide an external DMA Request for transferring printer data. If an external device is not used, the Bit Rotation Block can directly transfer print data to memory. To operate in this mode bit 9 should be set to 1, bit 6 set to 0, and bit 7 set to 0 for DMA writes to memory. If bit 6 is set to 1, bit 9 must be set to 0.

Bit 8:	If the Bit Rotation Block is selected (bit 6 = 0), and the external data bus for the Print ASIC is byte wide, this bit needs to be '0'. If the Bit Rotation Block is selected (bit 6 = 0), and the external data bus for the Print ASIC is halfword wide, this bit needs to be '1'. If external memory is selected (bit 6 = 1), the external bus data width for the Print ASIC and external memory need to be matched (this bit is not used).
Bit 7:	Control bit definition is with reference to the DMA requested device and controls the read/write signals during the DMA cycle. If it is set to write, the write strobe will be active during the DMA cycle.
Bit 6:	If the Bit Rotation Block is selected (bit 6 = 0), the DMA Request will be held off when the Bit Rotation Block output register is not ready. The ASIC output bus drivers will be disabled during external DMA request write (bit 6 = 1 and bit 7 = 0).
Bit 5:	Controls the address count direction performed after each DMA cycle.
Bit 4-0:	These bits control how the address registers will count after each DMA cycle. Note: If byte mode is set, the Count By 1 should be set.

> *Note: Requests to this channel are delayed by two clocks for synchronization.*

Address:ch2csbs	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA 2 Transfer Block Size Reg. *(DMA2BlockSize)* $xx81B5	Channel 2 Enable = 1	Not Used	Not Used	Not Used	Not Used	Not Used	Upper two bits of the Block Size counter		Rst. Value 00h Read Value 00h
Address:ch2csbs	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA2 Transfer Block Size Low Byte Reg. *(DMA2BlockSizeLo)* $xx81B4	Low Byte Value for the External DMA Block Size Counter								Rst. Value 00h Read Value 00h

- DMA Channel 2 Block Size Limit Counter
- block size of DMA channel 2 (halfwords)
  - Unlimited block size (block size = ∞):

    bit[7:0] of the DMA2BlockSize register = 00h
    bit[9:8] of the DMA2BlockSize register = 00b.

  - Limited block size (block size = 1 - 1023)
- The Block Size counter will decrement regardless of the DMA Address counter's activity. The block size register content is 0000h when the block size limit is reached.
- Please see the operation description of DMA2BlkSize and DMA2BufBlkSize registers below (in the DMA2BufBlkSize register section).
- Writing to this register will also clear the DMA channel 2 interrupt.

| Bit 15: | Channel 2 Enable is cleared when the block size limit is reached, and must be set, (enabled) after a new block size is entered. The block size can only be written into the DMA2Blksize register when this bit is 0. |

Address:ch2csbbs	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Ch2 Buffered Transfer Block Size Reg. *(DMA2BfBlockSize)* **$xx81B7**	Ch2 Enable = 1	Not Used	Not Used	Not Used	Not Used	Not Used	Upper two bits of the Block Size counter		Rst. Value 00h Read Value 00h
Address:ch2csbbs	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA Ch2 Buffered Transfer Block Size Low Byte Reg. *(DMA2BfBlockSizeLo)* **$xx81B6**	Low Byte Value for the External DMA Block Size Counter								Rst. Value 00h Read Value 00h

- DMA Channel 2 Buffered Block Size Limit Counter
- block size of DMA channel 2 (1/2 Words)
    - Unlimited block size (block size = ∞):

      bit[7:0] of the DMA2BlockSize register = 00h
      bit[9:8] of the DMA2BlockSize register = 00b.
    - Limited block size (block size = 1 - 1023)

- The first transfer block size and the starting address must be set up in the DMA2BlkSize and DMA2 address registers. The second transfer block size and the starting address must be set up in the DMA2BufBlkSize and DMA2 buffered address registers. Then, Firmware only needs to update the DMA2BufBlkSize and DMA2 buffered address registers when the DMA channel 2 interrupt occurs. This allows preloading of the next block size and starting address before the DMA operation for the current block has completed.
- The Buffered Block Size will be down loaded into the Block Size Counter when the current Block Size is reached and a new value has been written into this buffered block size register. Writing to this register will also clear the DMA channel 2 interrupt.
- The block size can be written into the DMA2BufBlkSize register only when the value of bit 15 in the DMA2BlkSize register is set to 0.

Address: chcsbl[2]	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Ch2 Buffered Low Counter *(DMA2Cntblo)* **$xx818D**	DMAB2 Addr. Bit 15	DMAB2 Addr. Bit 14	DMAB2 Addr. Bit 13	DMAB2 Addr. Bit 12	DMAB2 Addr. Bit 11	DMAB2 Addr. Bit 10	DMAB2 Addr. Bit 9	DMAB2 Addr. Bit 8	Rst. Value 00h Read Value 00h
Address: chcsbl[2]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA Ch2 Buffered Low Counter *(DMA2Cntblo)* **$xx818C**	DMAB2 Addr. Bit 7	DMAB2 Addr. Bit 6	DMAB2 Addr. Bit 5	DMAB2 Addr. Bit 4	DMAB2 Addr. Bit 3	DMAB2 Addr. Bit 2	DMAB2 Addr. Bit 1	DMAB2 Addr. Bit 0	Rst. Value 00h Read Value 00h

- DMA Channel 2 Lower Buffered Address Counter Value
- The Buffered Address Counter Value will be down loaded into the Address Counter when the current Block Size is reached and a new value has been written into this register.

Address: chcsbh[2]	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Ch2 Buffered Hi Counter *(DMA2Cntbhi)* **$xx818F**	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Not Used	Rst. Value 00h Read Value 00h
Address:chcsbh[2]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA Ch2 Buffered Hi Counter *(DMA2Cntbhi)* **$xx818E**	DMAB2 Addr. Bit 23	DMAB2 Addr. Bit 22	DMAB2 Addr. Bit 21	DMAB2 Addr. Bit 20	DMAB2 Addr. Bit 19	DMAB2 Addr. Bit 18	DMAB2 Addr. Bit 17	DMAB2 Addr. Bit 16	Rst. Value 00h Read Value 00h

- DMA Channel 2 Upper Buffered Address Counter Value
- The Buffered Address Counter Value will be down loaded into the Address Counter when the current Block Size is reached and a new value has been written into the Buffered Block Size register.

Address:ch5csbs	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA5 Transfer Block Size Reg. *(DMA4BlockSize)* **$xx81BB**	Channel 5 Enable = 1	Not Used	Upper six bits of the Block Size counter						Rst. Value 00h Read Value 00h
Address:ch5csbs	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA5Transfer Block Size Reg. *(DMA4BlockSize)* **$xx81BA**	Low Byte Value for the PIO to Memory Block Size Counter								Rst. Value 00h Read Value 00h

- DMA Channel 5 Block Size Limit Counter
- block size of DMA channel 5 (bytes)
  - Unlimited block size (block size = ∞):

    bit[13:0] of the DMA5BlockSize register = 0000h
    Limited block size (block size = 1 - 16380)
- The Block Size counter will increment regardless of the DMA Address counter's activity

Bit 15:   Channel 5 Enable is cleared when the block size limit is reached, and must be set, (enabled) after a new block size is entered. This bit is also inverted and sent to the PIO as MaxDmaCnt. The new block size can only be written when this bit is 0.

Address:ch9csbs	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA9 Transfer Block Size Reg. *(DMA9BlockSize)* **$xx81BF**	Channel 9 Enable = 1	Not Used	Not Used	Not Used	Upper four bits of the Block Size counter				Rst. Value 00h Read Value 00h
Address:ch9csbs	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
DMA9Transfer Block Size Reg. *(DMA9BlockSize)* **$xx81BE**	Low Byte Value for the Bi-level resolution conversion logic Block Size Counter								Rst. Value 00h Read Value 00h

- DMA Channel 9 Block Size Limit Counter
- block size of DMA channel 9 (bytes)
  - Unlimited block size (block size = ∞):

    bit[11:0] of the DMA9BlockSize register = 000h
    Limited block size (block size = 1 - 4095)
- The Block Size counter will decrement regardless of the DMA Address counter's activity. The block size register content is 0000h when the block size limit is reached.
- Writing to this register will also clear the bi-level resolution conversion interrupt.

Bit 15:   Channel 9 Enable is cleared when the block size limit is reached, and must be set, (enabled) after a new block size is entered. The new block size can only be written when this bit is 0.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA6/10 Throttle *(DMA6/10Throttle)* **$01FF81BD**	Throttle Ch. Select 0= Ch. 6 1= Ch. 10	Throttle Value bit 14	Throttle Value bit 13	Throttle Value bit 12	Throttle Value bit 11	Throttle Value bit 10	Throttle Value bit 9	Throttle Value bit 8	Rst. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
DMA6/10 Throttle *(DMA6/10Throttle)* **$01FF81BC**	Throttle Value bit 7	Throttle Value bit 6	Throttle Value bit 5	Throttle Value bit 4	Throttle Value bit 3	Throttle Value bit 2	Throttle Value bit 1	Throttle Value bit 0	Rst. Value 00h Read Value 00h

- Throttle Value (TV) = 1-32767 (TV= 0; disables Throttle Function)

  Throttle Value (TV) allows 1 DMA access every "TV*SIUCLK" time period.

- DMA Channel 6 OR 10 Throttle Control: This register is loadable while the DMA is active. Write to this register will reset the throttle timer.

Address:ch10csbs	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA10 Transfer Block Size Reg. *(DMA10BlockSize)* **$xx81C1**	Channel 10 Enable = 1	Not Used	Not Used	Not Used	Upper four bits of the Block Size counter				Rst. Value 00h Read Value 00h
Address:ch10csbs	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
DMA10Transfer Block Size Reg. *(DMA10BlockSize)* **$xx81C0**	Low Byte Value for the PIO to Memory Block Size Counter								Rst. Value 00h Read Value 00h

- DMA Channel 10 Block Size Limit Counter
- block size of DMA channel 10 (bytes)
  - Unlimited block size (block size = ∞):

    bit[11:0] of the DMA10BlockSize register = 000h

  - Limited block size (block size = 1 - 4095)

- The Block Size counter will decrement regardless of the DMA Address counter's activity. The block size register content is 0001h when the block size limit is reached.
- Writing to this register will also clear the DMA channel 10 interrupt.

Bit 15: Channel 10 Enable is cleared when the block size limit is reached, and must be set, (enabled) after a new block size is entered. The new block size can only be written when this bit is 0.

Address: csincconf	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Increment Configuration *(DMAIncConfig)* **$xx81C3**	Not Used	USB3 Inc = 0 Dec = 1	USB2 Inc = 0 Dec = 1	USB1 Inc = 0 Dec = 1	USB0 Inc = 0 Dec = 1	DMA 12 Inc = 0 Dec = 1	DMA 11 Inc = 0 Dec = 1	DMA 10 Inc = 0 Dec = 1	Rst. Value 00h Read Value 00h
Address: csincconf	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
DMA Increment Configuration *(DMAIncConfig)* **$xx81C2**	DMA 9 Inc = 0 Dec = 1	DMA 8 Inc = 0 Dec = 1	DMA 7 Inc = 0 Dec = 1	DMA 6 Inc = 0 Dec = 1	DMA 5 Inc = 0 Dec = 1	DMA 4 Inc = 0 Dec = 1	DMA 1 Inc = 0 Dec = 1	Not Used	Rst. Value 00h Read Value 00h

- DMA Address Increment Configuration. The Increment Configuration control bit allows the user to Select where the DMA channel Increments or Decrements after each access.

Address: cscontenb	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Count Enable *(DMACntConfig)* **$xx81C5**	Not Used	Not Used	Not Used	Not Used	Not Used	DMA 12 Count Enable	DMA 11 Count Enable	DMA 10 Count Enable	Rst. Value 07h Read Value 07h
Address:cscontenb	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA Count Enable *(DMACntConfig)* **$xx81C4**	DMA 9 Count Enable	DMA 8 Count Enable	DMA 7 Count Enable	DMA 6 Count Enable	DMA 5 Count Enable	DMA 4 Count Enable	DMA 1 Count Enable	Not Used	Rst. Value FEH Read Value FEH

- DMA Address Count Enable Control. The Count Enable control bit allows the user to enable address Count after each DMA access. If the Count Enable control bit is **not set** the address will remain fixed.

Address: cscontenb	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
DMA Endian Control *(DMAEndian)* **$xx81C7**	Not Used	Not Used	Not Used	DMA 12 Endian Control	DMA 11 Endian Control	DMA 10 Endian Control	DMA 9 Endian Control	DMA 8 Endian Control	Rst. Value 00h Read Value 00h
Address:cscontenb	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
DMA Endian Control *(DMAEndian)* **$xx81C6**	DMA 7 Endian Control	DMA 6 Endian Control	DMA 5 Endian Control	DMA 4 Endian Control	DMA 3 Endian Control	DMA 2 Endian Control	DMA 1 Endian Control	Not Used	Rst. Value 00H Read Value 00H

- DMA Address Endian Control. If the Endian Control Bit is set to a 1, the data structure will be changed between the Little Endian mode and the Big Endian mode.

The current data structure		The changed data structure
address 00: Byte 0	←→	address 11: Byte 0
address 01: Byte 1	←→	address 10: Byte 1
address 02: Byte 2	←→	address 01: Byte 2
address 03: Byte 3	←→	address 00: Byte 3

For DMA 1 (the external DMA channel) and DMA 2 (programmed as external DMA channel), this endian control can't be used for accessing external memory (the associated bit needs to be set to 0). If the external I/O device tries to get/put data from/to internal memory or registers, this endian control can be used with no problem. For example, if the external print ASIC tries to get data from the internal bit rotation block through DMA 2, this endian control can be used with no problem. If the external print ASIC tries to get data from the external memory through DMA 2, this endian control can't be used. In this case, the right endian structure should be prepared when the data is put into the external memory through the other process and DMA channels.

This page is intentionally blank

5. RESET Logic/Battery Backup/Watch Dog Timer

5.1 Reset Logic/Battery Backup

The MFC2000 has two power resets, Battery Power Reset and Prime Power Reset. The Battery Power Reset is the primary reset used to initialize battery-powered logic when battery power is first applied. Prime Power Reset initializes all non-battery powered logic whenever system power is applied. A third reset is generated by the watchdog timer or by the RESETn pin. The major logic blocks and associated MFC2000 signals are illustrated in Figure 5-1.

> **Note:**   *Reset logic operation requires use of the RTC Crystal (connected to XIN and XOUT).*
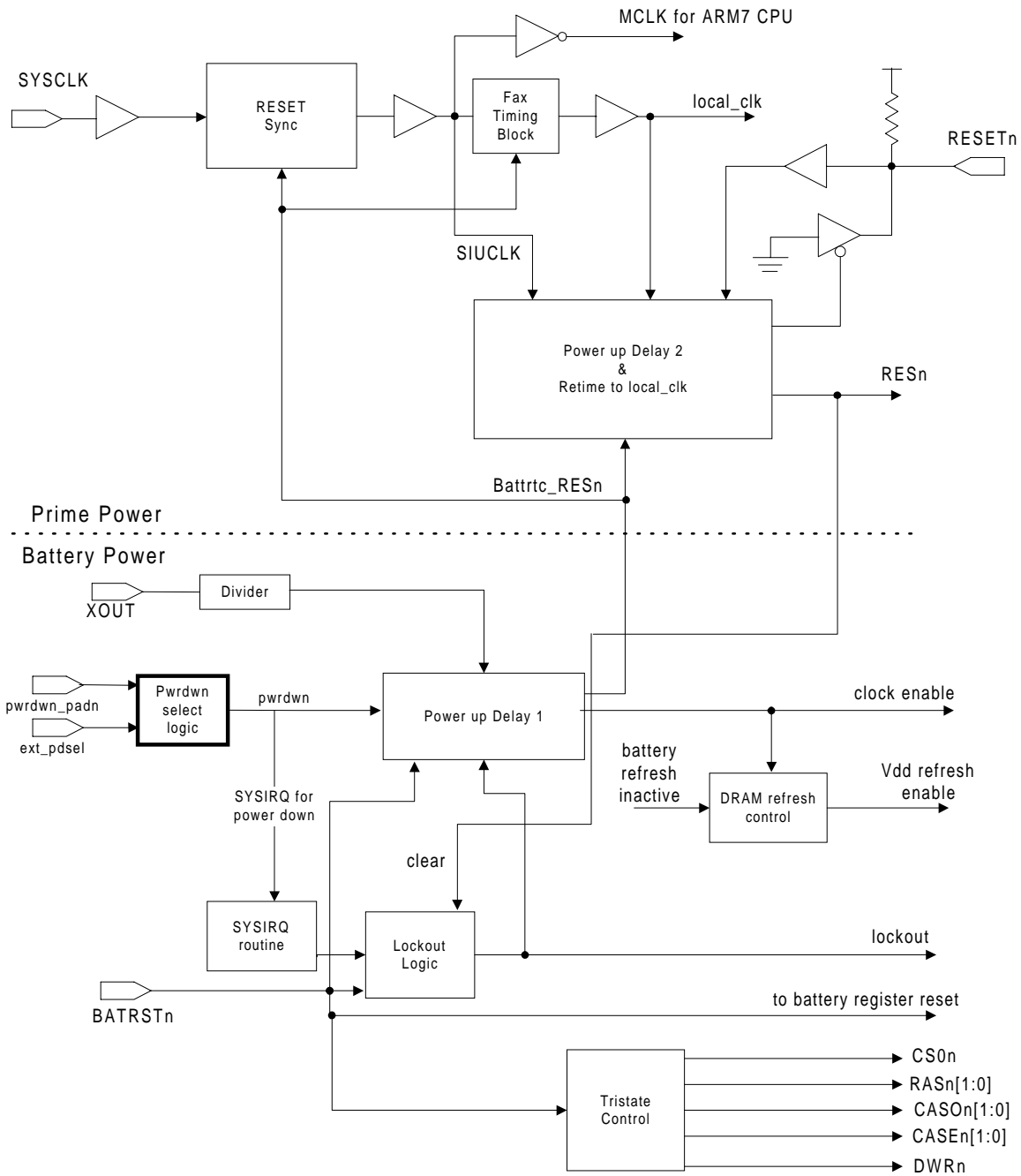
Figure 5-1. Power Reset Block Diagram
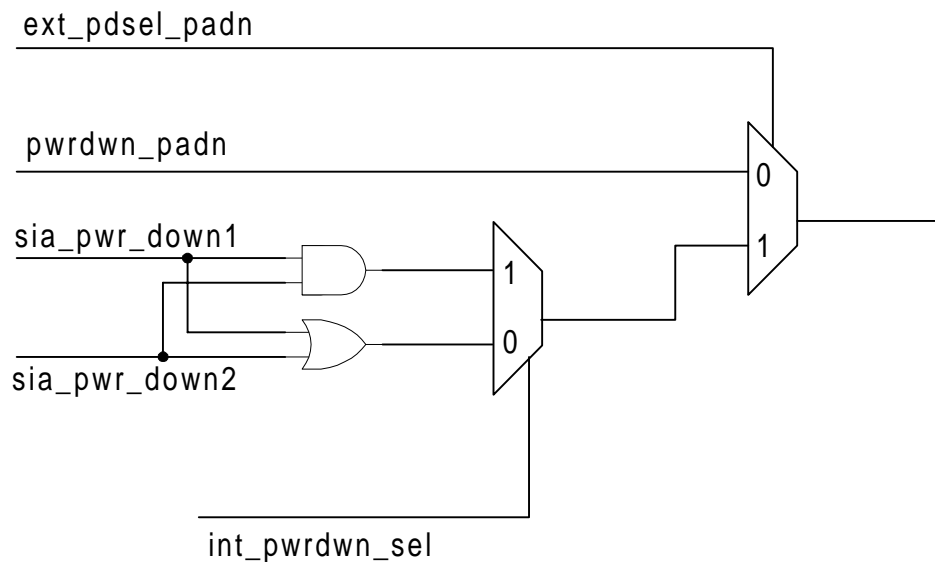
Figure 5-2. Power-down Select Logic

5.1.1   Battery Power Reset

Battery Power Reset is generated when BATRSTn (as shown at lower left in Figure 5-1) is driven low, and initializes the MFC2000 when battery power is first applied, as well as forcing a Prime Power Reset if V_{DD} is present. As long as battery power is maintained, there is no need to reactivate this reset. If the system design does not utilize the MFC2000's battery power features, this reset should be tied to PWRDWNn and be activated upon system power initialization.

Battery backup system requirements are undefined when battery power is first applied. Consequently, BATRSTn disables the battery backup control for DRAM and disables SRAM by resetting the BackupConfig register, and inhibits CPU access to this register via the (internal) lockout signal. The disabling of battery backup controls tri-states the associated output control signals (i.e., CS[0]n for SRAM, and CASOn[1:0]n, CASEn[1:0], RASn[1:0], and DWRn for DRAM, as shown at lower right in Figure 5-1.

> *Note:*   *The proper off-state logic level for battery backed-up devices external to the MFC2000 must be maintained by external pull-up/pull-down resistors on the appropriate MFC2000 pins.*

BATRSTn has a Schmitt trigger input buffer to allow the use of an external RC circuit to generate a reset.

5.1.2   Prime Power Reset

Prime Power Reset occurs when PWRDWNn (shown at upper left in Figure 5-1) is driven low while prime power is applied. There are two Prime Power Reset operating modes, one for V_{DD} power on and one for V_{DD} power off.

V$_{DD}$ Power On

V$_{DD}$ power on occurs when V$_{DD}$ power reaches it's normal operating level and PWRDWNn is driven high, resulting in "Power-Up Delay 1" (PUD1) initiation (shown at upper left in Figure 5-1). PUD1 has a duration of one or two Reset clocks (internal signal), depending on when PWRDWNn is driven high with respect to the Reset clock rising edge. Once PUD1 times out, the internal signal Battrtc_RESn is released. The internal Reset clock signal (RESn) duration is equal to approximately 125 ms for XOUT = 32.768 kHz.

> Reset clock = XOUT/4096

"Power-Up Delay 2" (PUD2) is a time delay whose duration is approximately 8 periods of SIUCLK (6 SIUCLK periods + time for retiming to AUXCLK). This delay allows CPU pre-operation initialization. Once PUD2 times out, RESETn gets resynchronized to AUXCLK and then, changes from an active low state to a high impedance state. After RESETn goes to the high impedance state, RESn goes high approximately after 8 periods of SIUCLK. After the internal RESn signal goes high, all internal logic gets reset and the battery back-up lockout is removed on the next XOUT rising edge.

V$_{DD}$ Power Off

V$_{DD}$ power off occurs when PWRDWNn is driven low while V$_{DD}$ is at a normal operating level. As an MFC2000 battery backed up feature, PWRDWNn has been designed to work with an early warning power loss detector. This allows the firmware to backup sensitive variables, and the MFC2000 to protect nonvolatile data from erroneous logic operations while prime power transitions through low voltage levels (i.e., < 3.135V).

When PWRDWNn is driven low, an SYSIRQ is issued to the CPU, and the CPU SYSIRQ routine then performs necessary system house cleaning tasks before power loss. The final SYSIRQ routine task is to write to the LockEnn register, which enables the battery lockout logic. The battery lockout logic forces an immediate lockout if battery DRAM operation is not enabled. If DRAM operation is enabled, the battery lockout logic waits for the absence of a V$_{DD}$ refresh cycle before performing the lockout. While in lockout, battery refresh mode commences, Battrtc_RESn becomes active, and the MFC2000 is inactive until the next V$_{DD}$ power on cycle.

Time-out Logic for the power-down lockout

There is a 1-2 second time-out on lockenn, if power down goes low enough to get latched into the battery logic, but a lockenn doesn't get generated by software. This will guarantee that a reset will occur if power down ever gets set and software doesn't set lockenn so that a reset gets generated.

There are two status bits to the backup configuration register. Bit 12 indicates that a battery reset (from the pin batrstn) has been detected. Bit 11 indicates that a lock enable has occurred and was caused by the time-out circuitry or a software write. To reset either of these bits a 1 must be written to the appropriate bit.

5.1.3 External Reset (RESETn) and Watchdog Reset

A driver on the AMPFC's RESETn pin provides a reset signal to external devices whenever the internal Battrtc_RESn or watchdog reset is active (low). This pin also includes an input driver that allows external control circuitry to reset the CPU and all V_{DD} powered registers (including a power-down SYSIRQ condition). The external RESETn and internal RESn are also activated when the watchdog timer times out (refer to the watchdog timer section that follows).

An externally generated reset does not affect the battery powered registers nor DRAM refresh operation.

[CAUTION] Don't drive the RESETn pin 'high' from external. Only drive the RESETn pin 'low' to generate the reset from external and don't drive the RESETn pin when external reset is not needed.

5.1.4 Power Reset Timing Diagram

The battery and V_{DD} power on reset timing is illustrated in Figure 5-1. The first event shown is the battery voltage and lockout rising together as battery voltage is applied, followed by BATRSTn going high, and the start up of XOUT. On the second falling edge of Reset clock after V_{DD} has been applied and PWRDWNn has been driven high, Battrtc_RESn goes high and approximately eight SIUCLKs are counted as Power-Up Delay 2. At the completion of eight SIUCLKs, battery lockout is removed allowing access to the BackupConfig register.
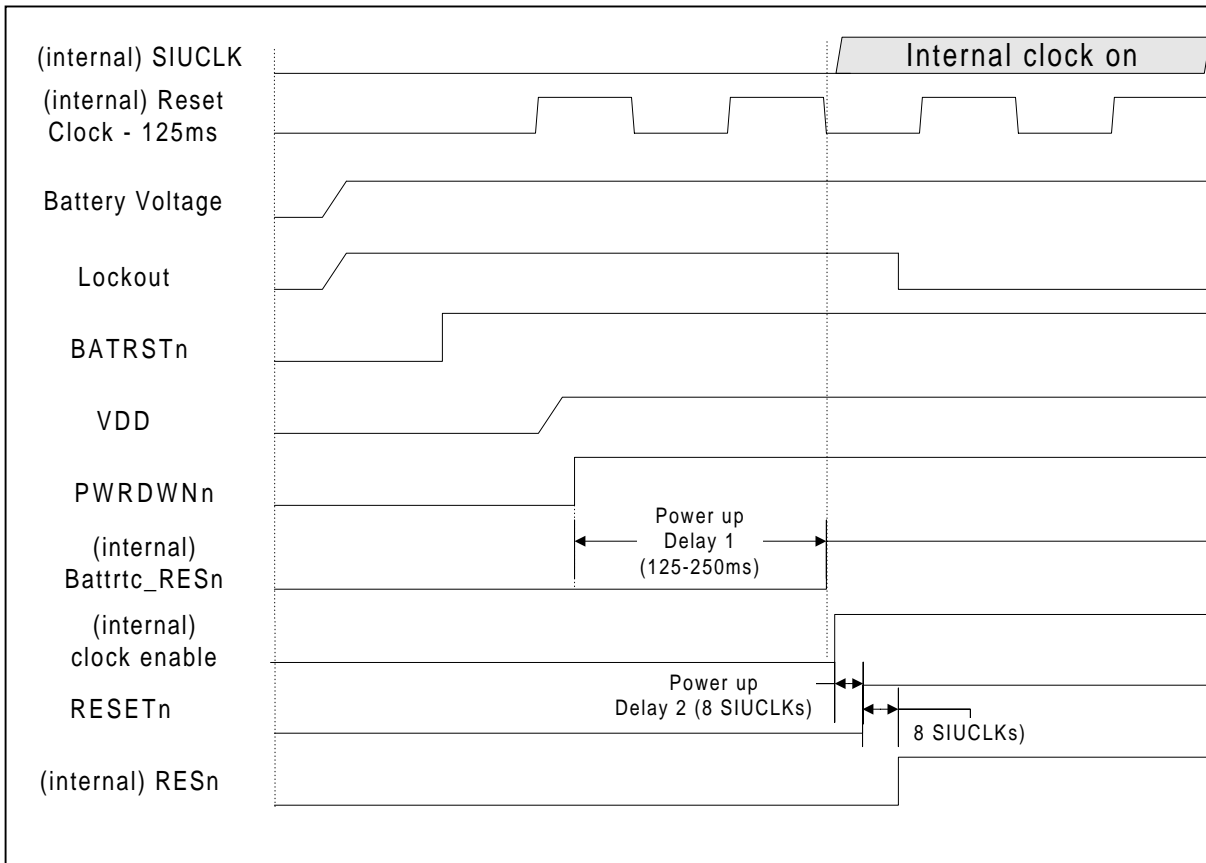
Figure 5-3. Power Reset Timing Diagram

5.1.5   Power on/off Sequence

5.1.5.1   Power Type

+3.3V prime power signal

- Connect to VDD pins on the MFC2000 chip to supply prime power to the MFC2000 die in the MFC2000 chip.

+5V prime power signal

- Connect to ADVA and ADVD pins on the MFC2000 chip to supply prime power to the ADC die in the MFC2000 chip.

- Connect to VGG pins on the MFC2000 chip to supply +5v prime power to all +5v tolerant pads in the MFC2000 chip.

+3V battery power signal

- Connect to the VBAT pin on the MFC2000 chip to supply battery power to the battery backup RTC in the MFC2000 chip and external SRAM.

- Connect to the VDRAM pin on the MFC2000 chip to supply battery power to the battery backup DRAM refresh logic in the MFC2000 chip and external DRAM.

5.1.5.2   Prime Power Sequence

It is recommended that +3.3v prime power signal is supplied first and then, supply +5v prime power signal for the prime power-on sequence. In other words, The voltage level of +5v prime power signal must be lower than the voltage level of +3.3v prime power signal before +3.3v prime power signal reaches the stable level.

V_{+5} is turned off first and then, turn off $V_{+3.3}$ for the prime power-off sequence. In other words, The voltage level of +5v prime power signal must be always lower than the voltage level of +3.3v prime power signal during the power-off sequence.

The recommended connection between +5v prime power signal and VGG is shown:
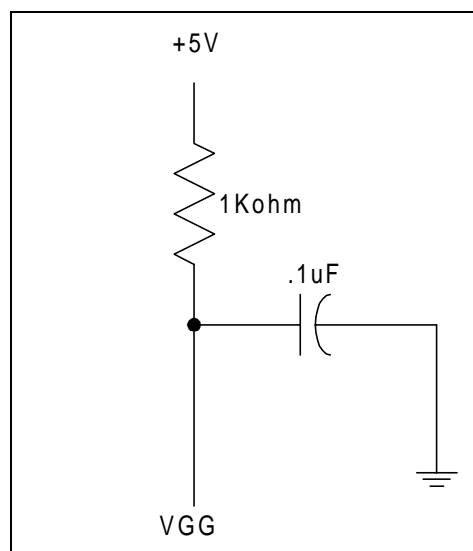
Figure 5-4. +5v Prime Power Signal and VGG

5.1.6   Register Description

Battery Backup and Prime Power Reset Registers

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Backup Configuration Register *(BackupConfig)* **$01FF8099**	(Not Used)	(Not Used)	Internal Power Down Select	Battery Reset Flag	Lockout Time-out Flag	SRAM Enable 0 = disable 1 = enable	Bank 1 Data interface size: 0 = 8-bit 1 = 16-bit	Bank 0 Data interface size: 0 = 8-bit 1 = 16-bit	Rst. Value xxx00000b Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Backup Configuration Register *(BackupConfig)* **$01FF8098**	DRAM Backup Time 0 = no backup 1 = 1-2 days 2 = 2-3 days 3 = infinite days		Refresh Rate 0 = normal 1 = slow	Oscillator Speed 0 = 32.768 KHz 1 = 65.536 KHz	Bank 1 Enable 0 = disable 1 = enable	Bank 0 Enable 0 = disable 1 = enable	Bank 1 Interleave Enable 0 = non interleaved 1 = 2 way interleaved	Bank 0 Interleave Enable 0 = non interleaved 1 = 2 way interleaved	Rst. Value 00h Read Value 00h

Register Description: This register is set to all zeros when first powered up and is battery backed up with the RTC Battery during power down. When a time out condition occurs, the RASn and CASn signals are tristated. When prime power has returned from a power down sequence, the user will have to perform a checksum on the DRAM data to know if a time out has occurred since there is no indication that the DRAM battery has lost power. The user will have to wait 1ms before accessing the DRAM after prime power has returned.

Bits 15-14:Not used

Bits 13:   Internal Power Down Select

0 = PWRDWNn is generated by or-ing power_down1 with power_down2

1= PWRDWNn is generated by and-ing power_down1 with power_down2

Note: Power_down1 and power_down2 are output signals from the power-down detection circuit 1 and 2.

Bit12:   Battery Reset Flag   This bit indicates that a BATRSTn occurred. To clear this bit, a 1 must be written to this bit.

Bit11:   Lockout Time-out Flag   This bit indicates that a power down occurred, but no lockout was set with in the 1-2 second period. The lockout timer initiated the LOCKENn to create the lockout condition. Once the lockout condition occurs, if the power down signal is high, the chip will come out of reset after a PUD1 delay. To clear this bit, a 1 must be written to this bit.

Bit10:   SRAM Chip Select Enable   This bit enables the SRAM chip select CSN0.

Bit 9:   Bank 1 Interface Size   This register defines whether the databus to the bank 1 DRAMs is 8 bits wide or 16 bits wide. An 8-bit wide DRAM interface uses RASn[1] and CASOn[0]. A 16-bit wide non-interleaved DRAM interface uses RASn[1] and CASOn[1:0]. A 16-bit wide interleaved DRAM interface uses RASn[1], CASEn[1:0], and CASOn[1:0].

Bit 8:   Bank 0 Interface Size   This register defines whether the databus to the bank 0 DRAMs is 8 bits wide or 16 bits wide. An 8-bit wide DRAM interface uses RASn[0] and CASOn[0]. A 16-bit wide non-interleaved DRAM interface uses RASn[0] and CASOn[1:0]. A 16-bit wide interleaved DRAM interface uses RASn[0], CASEn[1:0], and CASOn[1:0].

Bits 7-6:   DRAM Battery Backup Time   These bits control the amount of time that the DRAM controller will spend refreshing the DRAMs when in the battery backup mode. After reset when the CPU is being released to run, the CPU will not be able to write data to bits 7 and 6 of the BackupConfiguration register immediately since the immediate write will not take effect. The CPU must wait at least one oscillator clock cycle before writing data into bits 7 and 6.

Bit 7	Bit 6	Battery Backup Duration:
0	0	No battery backup
0	1	1-2 days
1	0	2-3 days
1	1	Infinite

Bit 5-4:   DRAM Refresh Rate   These bits are used to set up the DRAM refresh rate. See the following table:

Refresh Speed: (bit 5)	Oscillator Speed: (bit 4)	Refresh Speed:	Description:
0	0	normal	RTC crystal frequency = 32.768 kHz, Refresh clock = the crystal frequency= 32.768 kHz, The refresh cycle time = 15.625 ms/1024 cycles.
0	1	fast	RTC crystal frequency = 65.536 kHz, Refresh clock = the crystal frequency = 65.536 kHz, Refresh cycle time = 7.8125 ms/1024 cycles.
1	0	slow	RTC crystal frequency = 32.768 kHz, Refresh clock = the crystal frequency/8 = 4.096 kHz Refresh cycle time = 125 ms/1024 cycles.
1	1	slow	RTC crystal frequency = 65.536 kHz, Refresh clock = the crystal frequency/16 = 4.096 kHz Refresh cycle time = 125 ms/1024 cycles.

Bits 3:   Bank 1 Enable   This bit controls whether or not the Bank 1 DRAMs will be enabled. The Enable signal will allow CAS before RAS refresh to occur based on the non-interleave or interleave setting. If the bank setting indicates a non-interleaved mode, RASn[1] and CASOn[0] (8-bit mode) or CASOn[1:0] (16-bit mode) will refresh the DRAM. If the bank setting indicates an interleaved mode, RASn[1], CASOn[1:0] and CASEn[1:0] will refresh the DRAM. DWRn will be high during refresh. CASEn[1:0] will be tristated if Bank 0 is in non-interleaved mode. If this bit is set to a zero, RASn[1] will be tristated. Default disabled.

Bit 2:   Bank 0 Enable   This bit controls whether or not the Bank 0 DRAMs will be enabled. The Enable signal will allow CAS before RAS refresh to occur based on the non-interleave or interleave setting. If the bank setting indicates a non-interleaved mode, RASn[0] and CASOn[0] (8-bit mode) or CASOn[1:0] (16-bit mode) will refresh the DRAM. If the bank setting indicates an interleaved mode, RASn[0], CASOn[1:0] and CASEn[1:0] will refresh the DRAM. DWRn will be high during refresh. CASEn[1:0] will be tristated if Bank 1 is in non-interleaved mode. If this bit is set to a zero, RASn[0] will be tristated. Default disabled.

Bit 1:   Bank 1 Interleave Enable   This register defines whether the bank 1 DRAMs are to be accessed using 2-way interleave or non-interleaved access. 2-way interleaved access is only valid with a 16-bit interface (the 16 bit vs. 8 bit interface size bit for bank 1 is not used by the DRAM controller in the interleave mode, but is used by the SIU to output the data correctly).

Bit 0:   Bank 0 Interleave Enable   This register defines whether the bank 0 DRAMs are to be accessed using 2-way interleave or non-interleaved access. 2-way interleaved access is only valid with a 16-bit interface (the 16 bit vs. 8 bit interface size bit for bank 1 is not used by the DRAM controller in the interleave mode, but is used by the SIU to output the data correctly).

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Lock Enable *(LockEnn)* **01FF80A1(DW)**	A Dummy write to the Lock Enable register will enable the battery/prime power lockout sequence for a power down condition.								Rst. Value xxh Read Value xxh
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Lock Enable *(LockEnn)* **01FF80A0 (DW)**	A Dummy write to the Lock Enable register will enable the battery/prime power lockout sequence for a power down condition.								Rst. Value xxh Read Value xxh

• A write to this register enables the battery power lockout sequence for a power down condition. Lockout can only be cleared by resn.

• This is not a battery backed-up register.

5.1.7   POWER DOWN DETECTION

The power down detection circuit was only external for the MFC1000 chip. The PWRDWNn input signal to MFC1000 is the output of the external power down detection circuit. The MFC2000 has two internal power down detectors. The PWRDWNn is configurable either from internal power down detectors or from external power down detector. The EXT_PWRDWN_SELn is used to select internal or external power down detector. The external power down detector is selected when the EXT_PWRDWN_SELn pin is low. If the internal power detectors are selected, the combination of two power down detector can be further selected through the internal register. The purpose of having internal power down detector is to automatically generate reset after voltage drop on the power supply without having external power down detector. The power supply voltage is compared with internally generated threshold voltages. In order to prevent erratic resets, the comparator is designed with hysteresis feature. During the power up, the output of the comparator will be low when the power supply voltage is at least V_{on}. Then as the voltage continues to increase and becomes larger than V_{on}, the comparator output will be high. If at any time, the supply voltage drops below the threshold voltage level of V_{off}, then the comparator output will be low.
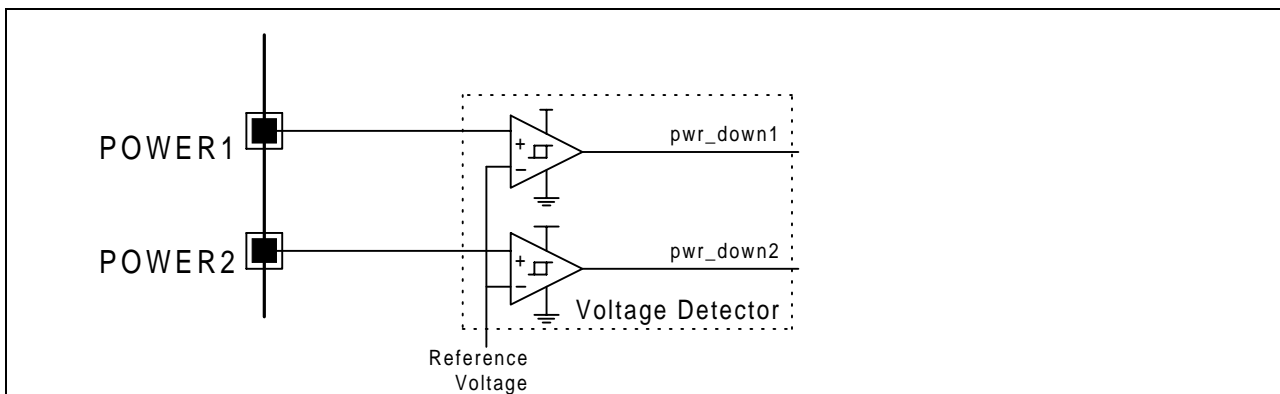
Figure 5-5. Internal Power Detection
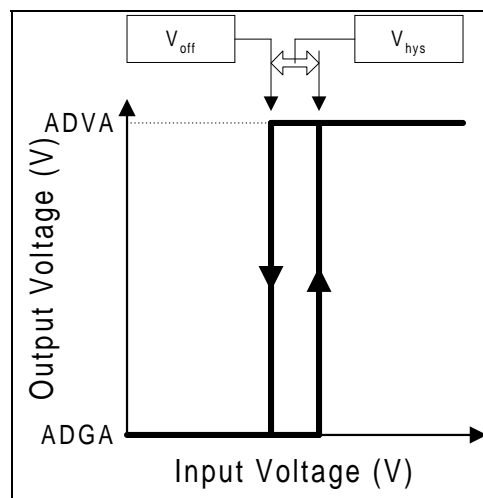
[Sia2k-comp-tf.vcd]

Figure 5-6. Figure Caption Required

	Voltage Threshold	Margin
V_{off}	TADCREF	±30 mV
V_{hys}	20 mV	±2 mV

> **Note**: *The output voltage to represent logic 0 must be guaranteed to be at ADGA (typically ADGA = GND = 0V).*

The input resistance to the voltage detector is specified as follows:

		Min
R_{in}	Input Resistance	10 MΩ

A voltage divider circuit will be used externally in order to obtain the desired voltage to monitor the power.
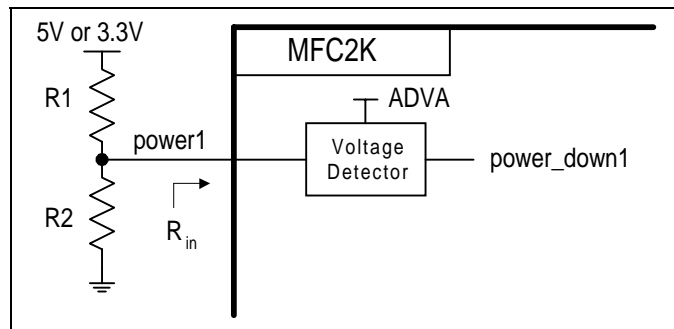
Figure 5-7. Voltage Divider Circuit

5.2   Watchdog Timer

The Watchdog Timer is intended to guard against firmware lockup on the part of either Executive-controlled background tasks or Interrupt-driven tasks, and can only be enabled by a sequence of events under control of the Watchdog Control Logic. Once the Watchdog Timer has been enabled, it can not be disabled (inactivated) unless a system reset occurs. The watchdog timer block diagram is provided in Figure 5-8, and the Watchdog Time-Out Timing Diagram is provided in Figure 5-9.

The Watchdog Timer is enabled by the following sequence of events, which MUST occur in the order described.

1. Write the value of the desired Time-out interval to the Watchdog Interval register. The first write to this register disables subsequent writes, and enables the next event in the enable sequence.

2. Write $0F followed by $F0 to the WatchdogEnRetrigger register. The data written to this register MUST be in the order specified or it is ignored.

3. After the operations are completed, the value in the Watchdog Interval register is loaded into the Time-out Down Counter.

4.   As soon as the Watchdog Timer is enabled, the Time-out Down Counter begins counting.

The Time-out Down Counter is decremented based on a 1 ms pulse which is derived from (125 μs prescaler period /8); the count occurs on the rising edge of SIUCLK.

After the Time-out Down Counter reaches 'zero', the (internal) watchdog reset signal becomes active at the next 1 ms pulse. The watchdog reset immediately activates the internal resn and external RESETn signals. These resets are deactivated when the Power Up Delay 2 timer times out (i.e., 8 SIUCLK's).

In order to prevent the Watchdog Timer from timing-out and generating a reset pulse, it must be retriggered by writing $0F followed by $F0 to the WatchdogEnRetrigger register. This action causes the value in the WatchdogInterval register to be re-loaded into the Time-out Down Counter.

> **Note:**   *The 125us prescaler must be initialized before activating the watchdog timer.*
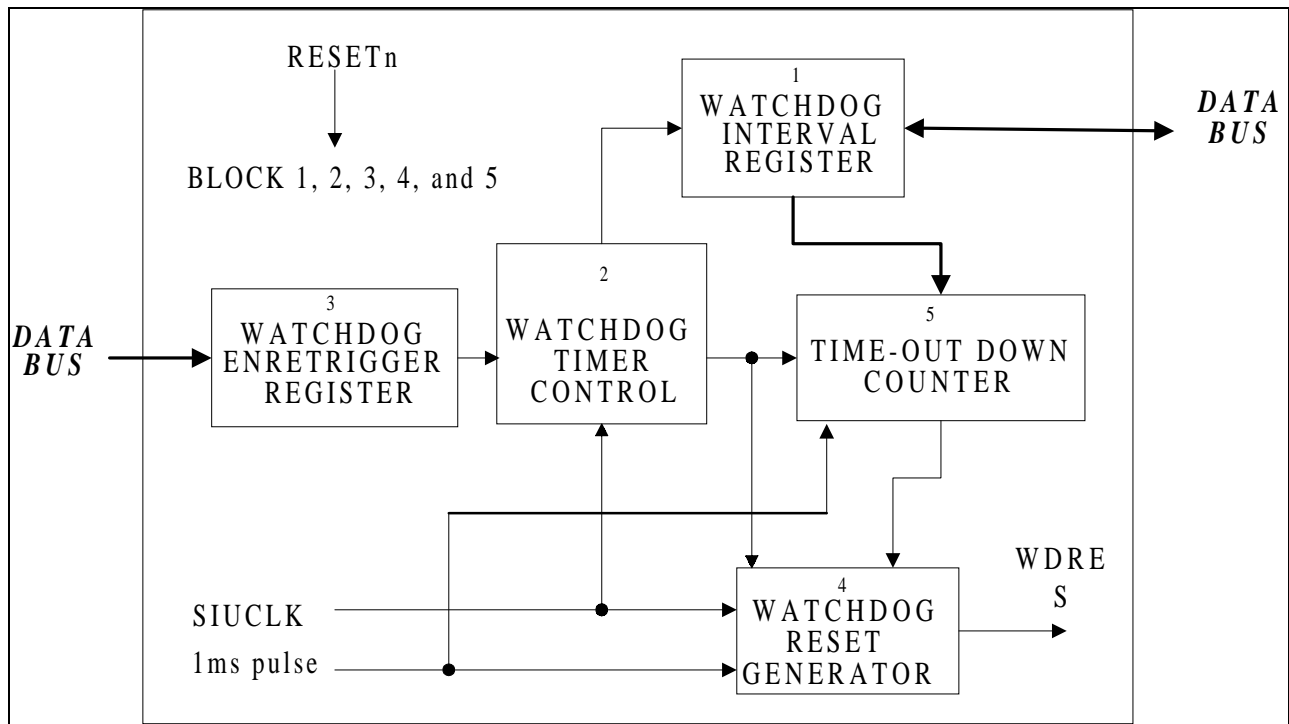
Figure 5-8. Watchdog Timer Block Diagram
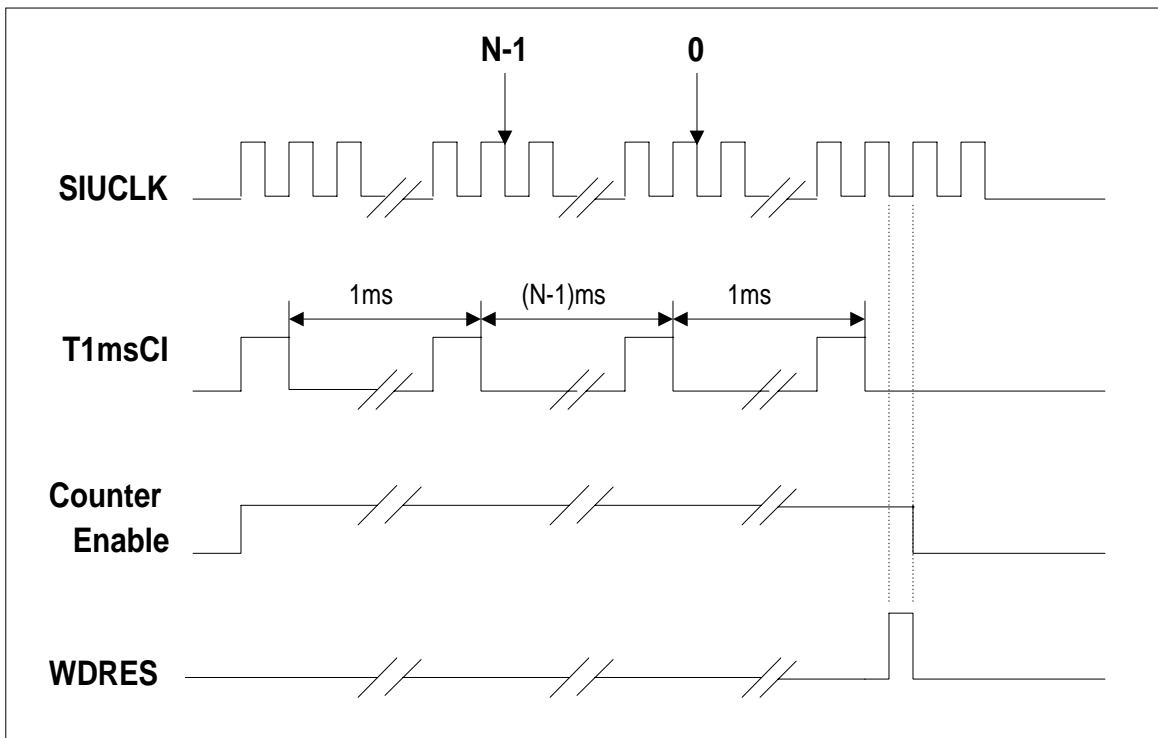
Figure 5-9. Watchdog Time-Out Timing Diagram

5.2.1   Watchdog Timer Registers

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Watchdog Enable Retrig. *(WatchdogEnRetrigger)* **01FF8041**	(Not Used)								Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Watchdog Enable Retrig. *(WatchdogEnRetrigger)* **01FF8040 (W)**	Writing 0Fh followed by F0h to this register after setting the Watchdog Interval register will enable the Watchdog timer. (The 125µsPrescaler must be initialized before activating the Watchdog timer.) Always reads back '00h'.								Rst. Value 00h Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Watchdog Interval *(WatchdogInterval)* **01FF8043**	(Not Used)								Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Watchdog Interval *(WatchdogInterval)* **01FF8042**	Watchdog Interval (n = 0-255) (n*1ms)								Rst. Value 00h Read Value 00h

- Once enabled, the Watchdog Timer can only be disabled by reset.

- On reset, the Watchdog Timer is disabled until it is enabled by writing to the WatchdogInterval register followed by writing $0F and $F0 to the WatchdogEnRetrigger register.

- The basic Watchdog time-out unit (1 ms) is based on the internal 125 μsec timer. Therefore, the 125μsPrescaler and MSINTPeriod registers (used to activate the timer) must be set in order to operate the Watchdog Timer.

- The accuracy of the time-out period is equal to the accuracy of the 125 μsec timer with an additional error of up to 1 msec (i.e., the amount of error is dependent on the relationship of the trigger point to the internal 1 msec timer).

- If N is programmed into the WatchdogInterval register, the Watchdog Timer time-out period is "N * 1 ms" (N = 0 - 255).

- Reading the address of the WatchdogInterval register returns the value of N that was programmed into the WatchdogInterval register.

- The following conditions do not enable the watchdog timer:

Only the first write to the WatchdogInterval register is valid. The subsequent writes to this register are ignored.

Any other data written to the WatchdogEnRetrigger register between the times $0F and $F0 are written is ignored.

Once enabled, the Watchdog Timer is retriggered on the writing of alternate $0F and $F0. (Writing of any other data, and the writing of repetitive $0F or $F0 is ignored.)

This page is intentionally blank.

6. Fax Timing Control Interface

6.1 PLL

6.1.1 Description

The MFC2000 is an ASIC that provides multifunctional control for color fax, color scan and color printing. The ASIC includes an ARM microprocessor running at 30 MHz, 37.5 MHz or 40 MHz and a Countach Imaging DSP running at 100 MHz or 85.7 MHz. Additionally, the ASIC also supports the USB interface running at 48 MHz and the AUXCLK at 10 MHz.

The PLL for the ASIC is designed to provide all the above frequencies in six operating modes based on the combination of ARM frequency select and Countach Imaging DSP frequency select. The six operating modes are defined in the following table.

6.1.2 Frequency Requirement

The frequencies for the six operation modes are defined as follows:

Table 6-1. Operation Mode Frequencies

CLKConfig[1:0]	CLKConfig[2]	ARM (SIUCLK)	Countach	USB	AUXCLK
00	1	30 MHz	85.7 MHz	48 MHz	10 MHz
01	1	37.5 MHz	85.7 MHz	48 MHz	10 MHz
10	1	40 MHz	85.7 MHz	48 MHz	10 MHz
00	0	30 MHz	100 MHz	48 MHz	10 MHz
01	0	37.5 MHz	100 MHz	48 MHz	10 MHz
10	0	40 MHz	100 MHz	48 MHz	10 MHz
11	Don't Care				

The MFC2000 can only operate in one mode at a time. The mode selection is made during the reset through clock configuration pins (CLKConfig[2:0]). The input reference frequency is 28.224 MHz, which is generated by a crystal oscillator pad. 28.224MHz is also used for the internal P80 modem DSP.

6.2   Fax Timing Logic

The fax timing logic block generates all of the time bases needed by the MFC2000. The fax timing logic block diagram is provided in Figure 6-1.

Notes:
(1) I_LCLPHI is only used internally to indicate the SIUCLK cycle before the last SIUCLK cycle in one ICLK time period.
(2) IHS_LCLPHI is only used internally to indicate the SIUCLK cycle before the last SIUCLK cycle in one IHSCLK time period.
(3) ICLK is the base clock for internal peripherals and is only used internally to synchronize all the internal logic. The frequency of ICLK must be equal to or less than 10MHz. ICLK duty cycle is 33% when ICLK = SIUCLK/3.
(4) IHSCLK is the high speed base clock for the PIO and Bit Rotation blocks and is only used internally to synchronize all the internal logic. The frequency of IHSCLK must be equal to or less than 20MHz.
(5) AUXCLK has the same frequency as ICLK and is output to the external world through a pin. AUXCLK is used to synchronize external peripherals. AUXCLK duty cycle is always 50% no matter what is the relationship between ICLK and SIUCLK.
(6) MMCLK is the base clock for the motor control logic.
(7) TIM_8MS is used by the PIO Block.
(8) TIM_1MS is the base clock for the the watchdog timer.
(9) TIM_MSINT is used to synchronize the scanning and motor stepping.
(10) TIM_1MS/2MSIRQ is used to generate an interrupt every 1ms or 2ms (selectable). The actual period of the 1MSIRQ = 1.00708 ms and 2MSIRQ = 2.01416ms

Figure 6-1. Fax Timing Control Logic Block Diagram

6.3 MFC2000 Timing Chain

The MFC2000 timing chain is illustrated in Figure 6-2.
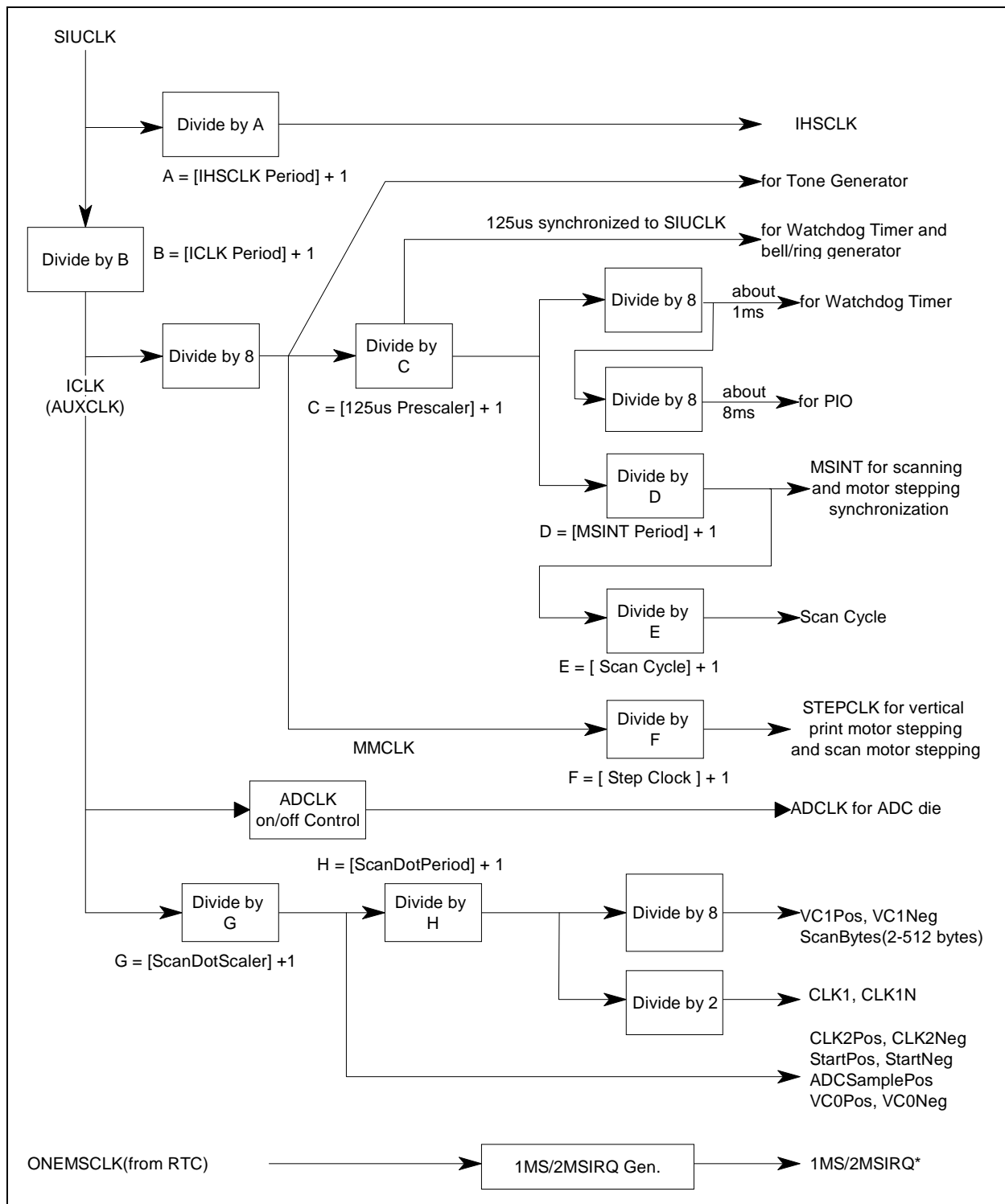
Figure 6-2. MFC2000 Timing Chain

6.4　Scan Control Timing

In this example, the scan motor has the constant speed. The scan motor stepping is synchronized to the MSINT. The scan IRQ is used to disable motor stepping when we want to finish scanning. If the delay time for the scan motor is long enough, the last step will occur after the scan IRQ. Then, we need to use the scan step IRQ (sstepirq) for disabling motor stepping. The detail description for the motor stepping is described in Section 12.
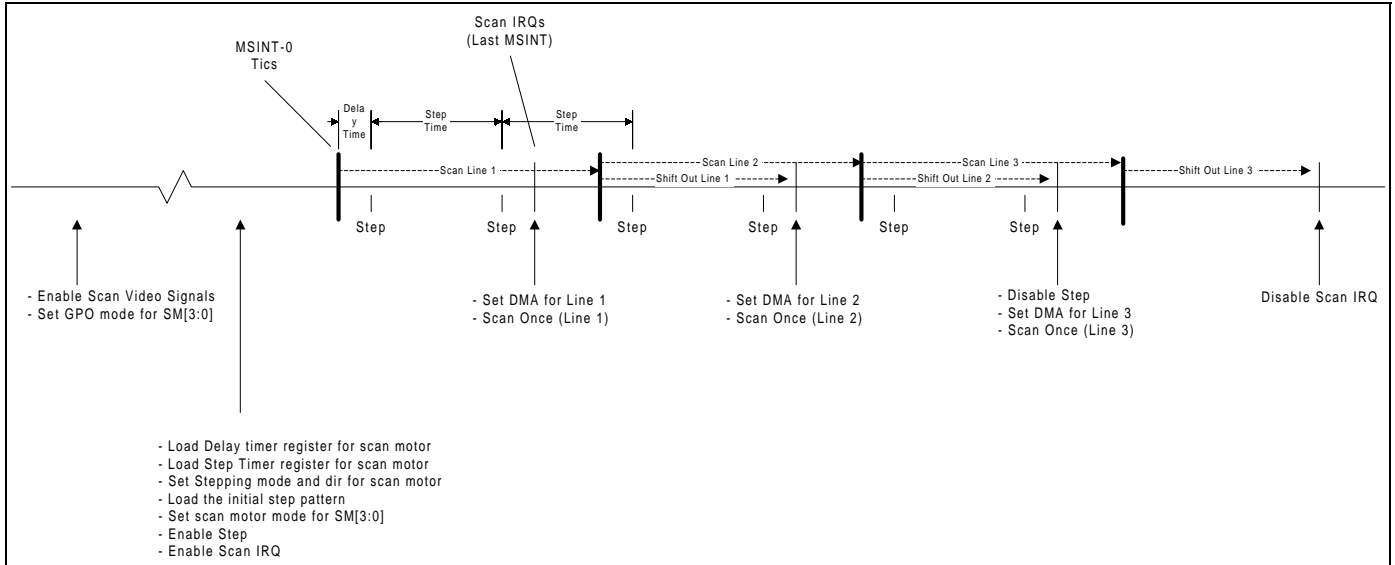
Figure 6-3. Scan Control Timing

6.5   Fax Timing Registers

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
125 uS Prescaler *(125usPrescaler)* **01FF8881**	(Not Used)								Rst Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
125 uS Prescaler *(125usPrescaler)* **01FF8880**	n (8 bits) as applied in the equation: (n+1)*(ICLK period *8)								Rst Value 00h Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ICLK(AUXCLK) Period *(ICLKPeriod)* **01FF8883**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ICLK(AUXCLK) Period *(ICLKPeriod)* **01FF8882**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	IHSCLK Period	ICLK (AUXCLK) Period ICLK= SIUCLK/(ICLKPeriod + 1)		Rst Value xxxxx111b Read Value 07h

IHSCLK = MCLK/(IHSCLKPeriod + 1)

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Mechanical Subsystem (MS) Period *(MSINTPeriod)* **01FF8885**	(Not Used)	(Not Used)	(Not Used)						Rst Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Mechanical Subsystem (MS) Period *(MSINTPeriod)* **01FF8884**	2MSIRQ Select 0=1MSIRQ 1=2MSIRQ	(Not Used)	n (6 bits) as applied in the equation: (n+1)*125 uS Prescaler period)						Rst Value 0x000000b Read Value 0x000000b

On Reset, the 125usPrescaler, 1-ms Timer, and MSINT counters are disabled. Writing to the MSINTPeriod register enables these timers and counters. In addition, the 125usPrescaler must be set to the appropriate value in order for the WatchDog and timing chains to operate correctly.

If a new value is written into the MSINTPeriod register while the MSINT counter is running, the new value is not loaded into the counter until it counts down to 0.

1MS/2MSIRQ is a programmable interrupt source. If the '2MSIRQ Select' bit (bit7) of the MSINTPeriod register is set to 0, the interrupt is generated every 1ms (1.00708ms). If the '2MSIRQ Select' bit (bit7) of the MSINTPeriod register is set to 1, the interrupt is generated every 2ms (2.01416ms).

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Interrupt Clear *(IntClear)* **01FF8887**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Interrupt Clear *(IntClear)* **01FF8886 (W)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	1=clear 1MS/2MSIRQ	Rst Value xxxxxxx0b Read Value 00h

7. VIDEO/SCANNER CONTROLLER

The video/scanner controller consists of three main blocks:

• Scanner controller
• Industry standard two-wire serial programming interface
• Video decoder interface

The following diagram shows the block diagram of VSC.
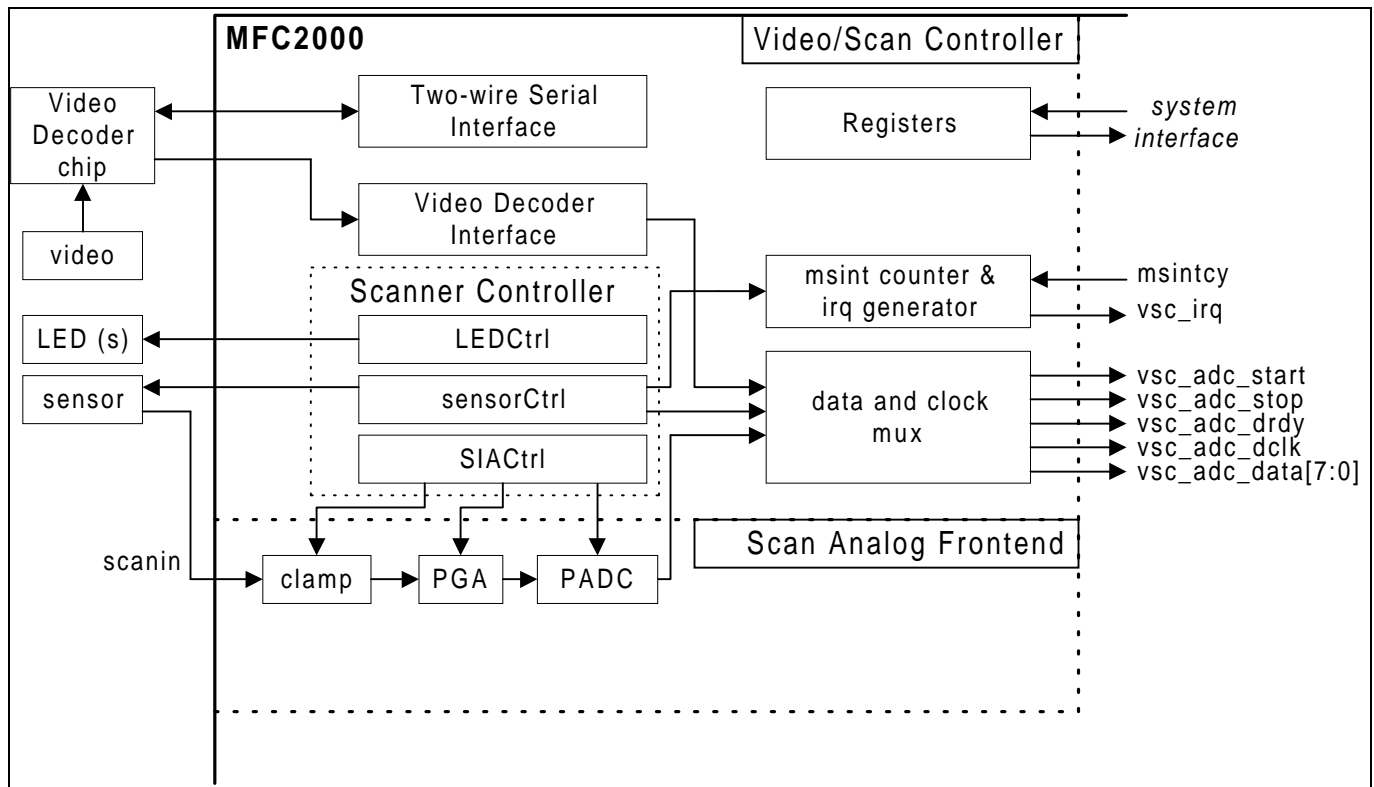
Figure 7-1. Video/Scanner Controller Block Diagram

7.1   Scanner Controller

The function of scanner controller can be divided into three areas:

1.   Scanner sensor controller

2.   Scanner light controller

3.   Scan integrated analog controller

Pins related to scanner and external ADC

⇨   Scanner related

SC_START[0]
SC_LEDCTRL[0]
SC_LEDCTRL[1]/SC_START[1]
SC_LEDCTRL[2]/SC_START[2]
SC_CLK1/SC_CLK2A
GPIO[0]/EXT_CLAMP
GPIO[8]/SC_CLK1/SC_CLK2B
GPIO[15]/SC_CLK1/SC_CLK2C
GPIO[22]/EADC_SAMPLE/SC_CLK2D
EV_VD[5]/ScanIAConfig[5] as GPO
EV_VD[4]/ScanIAConfig[4] as GPO
EV_VD[3]/SAMPLE/CLK2D
EV_VD[2]/CLK1/CLK2C
EV_VD[1]/CLK1/CLK2B
EV_VD[0]/CLAMP

⇨   Ext ADC related
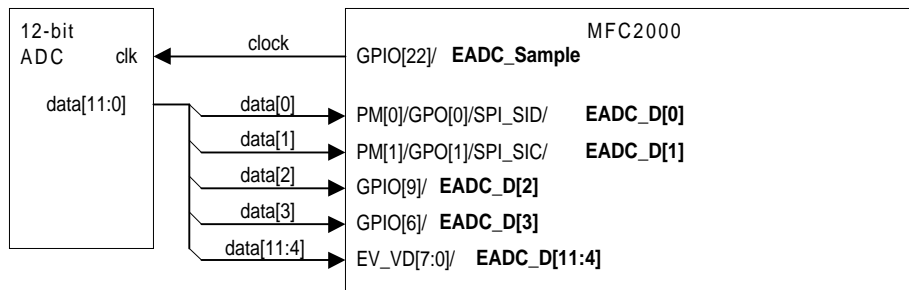
⇨   Related GPIO Configuration:

Configuration	Bit on GPIOConfig2 ($01FF8832)
EXT_CLAMP on GPIO[0]	set bit 4
SC_CLK1/2B on GPIO[8]	set bit 2
SC_CLK1/2C on GPIO[15]	set bit 3
EADC_SAMPLE/SC_CLK2D on GPIO[22]	set bit 6

7.1.1   Function Description

Features for scanner controller:

- supports wide variety of CIS and linear CCD scanners (color or monochrome)
- supports up to 8192 (= 2^{13}) pixels/line
- programmable pulse width modulation for light control (up to 3 LEDs)
- programmable clamping signal (mode, delay, width, length)
- programmable sampling location (mode, location)
- supports external 12-bit ADC with programmable latency and programmable data transfer position

MFC2000	Pin function	Alias
sc_led0	LEDCtrl[0]	scctrl[0]
sc_st0	start[0]	scctrl[1]
sc_led0st1	LEDCtrl[1]/start[1]	scctrl[2]
sc_led2st2	LEDCtrl[2]/start[2]	scctrl[3]
sc_clk12a	clk1/clk2a	scctrl[4]
sc_clk12b	clk1/clk2b	scctrl[5]
sc_clk12c	clk1/clk2c	scctrl[6]

A brief description of signal functionality:

start	Sensor control signal which is only active at the beginning of a scan cycle in order to initiate the transfer of sensor data
clk1	Sensor control signal which toggles at pixel boundary; thus the frequency of this signal is half of pixel frequency
clk2	Sensor control signal whose rising and falling edges are programmable within a pixel period; thus the frequency of this signal is the same as the pixel frequency
LEDCtrl	Light control signal which can either be the envelope or the modulation signal. The rising and falling edge of the envelope and the duty cycle of the modulation are programmable.

7.1.2   Register Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Control *(ScanCtrl)* **$01FF8541**	(Not Used)	(Not Used)	LEDCtrl[2] enable	LEDCtrl[1] enable	LEDCtrl[0] enable	start[2] enable	start[1] enable	start[0] enable	Rst. Value xx000000b Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Scan Control *(ScanCtrl)* **$01FF8540**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Scan Once	Rst. Value xxxxxxx0b Read Value 00h

Bit 13-11	LEDCtrl[2:0] enable	This bit will be synchronized with "cycle_start", and it is not auto cleared.
Bit 10-8	start[2:0] enable	This bit will be synchronized with "cycle_start", and it is not auto cleared.
Bit 0	ScanOnce	Writing 1 to this bit will initiate an image line scanning operation. This bit will be cleared automatically once the operation has been completed.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Control Status *(ScanCtrlStat)* **$01FF8543**	(Not Used)	(Not Used)	LEDCtrl[2] enable (Read only)	LEDCtrl[1] enable (Read only)	LEDCtrl[0] enable (Read only)	start[2] enable (Read only)	start[1] enable (Read only)	start[0] enable (Read only)	Rst. Value xx000000b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Scan Control Status *(ScanCtrlStat)* **$01FF8542**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Scan Once (Read only)	Rst. Value xxxxxxx0b Read Value 00h

This register is read only.

The value written in *ScanCtrl* register will be transferred to this register at the beginning of scan cycle.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
VSC IRQ Status *(VSCIRQStatus)* **$01FF8545**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
VSC IRQ Status *(VSCIRQStatus)* **$01FF8544**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	SPI IRQ	MSINT IRQ	Rst. Value xxxxxx00b Read Value 00h

Bit 1   SPI IRQ   This is the interrupt status which comes from SPI block. The enable for this interrupt can be found in SPI_Config register.

Writing 1 to the status bit will clear the interrupt.

Bit 0   MSINT IRQ   The scanenable must be set in order to have this interrupt active. The timing of this interrupt is programmable within a scan cycle.

Writing 1 to the status bit will clear the interrupt.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
VSC Control *(VSCCtrl)* **$01FF8547**	Shift Enable[7:0]								Rst. Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
VSC Control *(VSCCtrl)* **$01FF8546**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	VSC on EV bus	ihsclk period	VSC Mode	Rst. Value xxxxxxx0b Read Value 01h

Bit 15-8   Shift Enable[7:0]   These bits are used to shift scanner controller signals by ½ ihsclk. By default, all scanner controller signals are synchronized to the rising edge of ihsclk. By setting this bit, the corresponding scanner controller signal will be shifted by ½ ihsclk, i.e., it will be synchronized to the falling edge of ihsclk.

ShiftEnb[7]: to shift 'sample' signal
ShiftEnb[6:1]: to shift 'scctrl[6:1]' signals
ShiftEnb[0]: to shift 'clamp' signal

Bit 2   VSC on EV bus   When this bit is set, the "EV_VD[5:0]" bus will be turned into output with the following signals on the bus:

EV_VD[5:0]	Output signals
5	ScanIAConfig[5] as GPO
4	ScanIAConfig[4] as GPO
3	sample/clk2d
2	clk1/clk2c
1	clk1/clk2b
0	clamp

Bit 1   ihsclk period   0: ihsclk period = sysclk period
1: ihsclk period = sysclk period/2)

This bit determines the period of ihsclk (the primary clock of VSC).

Bit 0   VSC Mode   0: scanner
1: video

This mode will determine the source of data.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Cycle *(ScanCycle)* $01FF8891	(Not Used)	(Not Used)	(Not Used)	MSINT IRQ location					Rst. Value xxx11111h Read Value 1Fh
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Scan Cycle *(ScanCycle)* $01FF8890	(Not Used)	(Not Used)	(Not Used)	Scan Cycle (n) MSINTs per Scan Cycle = (n+1)					Rst. Value xxx00000b Read Value 00h

Bits 12-8   MSINT IRQ location   The value written here is double-buffered. At the beginning of a scan cycle, the value will be transferred into another buffer which is compared to a scan cycle counter. If the value in the buffer matches the value in the counter, then an interrupt will be generated (if it is enabled) at the beginning of the count.

Bits 4-0   Scan Cycle   (n, n = 0-31; MSINTs per scan cycle = n+1)

The value written here is double-buffered. At the beginning of a scan cycle, the value will be transferred into another buffer.

Notes

1. *Bits [4:0] of this register specify the number of MSINTs (1-32) per Scan Cycle.*
   Total Scan Cycle Period = [MSINT period * (ScanCycle register value + 1)].

2. *The transition from maximum count to 0 determines the start of a scan cycle.*

3. *Reading bits [4:0] of this register returns the current scan cycle MSINT count.*

4. *This register must be set up prior to enabling the scan (i.e., setting the "scanenable" bit in ScanConfig register).*

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Configuration *(ScanConfig)* $01FF8893	MSINT IRQ Enable	scctrl[6] invert	scctrl[5] invert	scctrl[4] invert	scctrl[3] invert	scctrl[2] invert	scctrl[1] invert	scctrl[0] invert	Rst. Value 00h Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Scan Configuration *(ScanConfig)* $01FF8892	(Not Used)	External ADC Enable	scctrl[6] select 0: clk1 1: clk2c	scctrl[5] select 0: clk1 1: clk2b	scctrl[4] select 0: clk1 1: clk2a	scctrl[3] select 0: led2 1: start	scctrl[2] select 0: led1 1: start	Scan Enable	Rst. Value xx000000b Read Value 00h

Bit 15	MSINT IRQ Enable	When this bit and ScanEnable bit are set, an interrupt will be generated based on the MSINT IRQ location defined in ScanCycle register.
Bit 14-8	scctrl[6:0] invert	These bits are used to invert the polarity of the corresponding control signal.
Bits 6	External ADC Enable	0 = internal ADC as the source of adc data 1 = external ADC as the source of adc data
Bit 5-1	scctrl[6:0] select	These bits are used to configure which scanner controller signals should be connected to the pins.
Bit 0	Scan Enable	This bit must be set in order to activate the scanner controller logic.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Dot Control *(ScanDotCtrl)* $01FF8895	(Not Used)								Rst. Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Scan Dot Control *(ScanDotCtrl)* $01FF8894	Scan Dot Scaler Value (0-15) n as used in the following equation: (n+1)*IHSCLK				Scan Dot Period (1-15) n as used in the following equation: (n+1)*ScanDotScaler				Rst Value 00h Read Value 00h

Bits 7-4	ScanDotScaler	(n: 0-15), (n+1) * IHSCLK period
Bits 3-0	ScanDotPeriod	(n: 1-15), (n+1) * ScanDotScaler

Note: *ScanDotPeriod = 0 is not allowed.*

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Length *(ScanLength)* **$01FF8897**	(Not Used)	(Not Used)	(Not Used)	ScanLength[12:8]					Rst. Value xxx00000b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Scan Length *(ScanLength)* **$01FF8896**	ScanLength[7:0]								Rst. Value 00h Read Value 00h

Bits 12-0 ScanLength[12:0]   This register specifies the number of valid pixels in one scan line. The amount of data stored in a line is ScanLength * 2 bytes. Each pixel is represented by 12-bit data, and it is stored in memory as 2 bytes of data.

> *Note: The internal pixel counter is 14-bit wide. The extra bit is used to match the maximum value of clamp delay and LED edges. It is guaranteed that this counter will not roll-over once it reaches its maximum count.*

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Start Delay *(ScanStartDelay)* **$01FF8899**	(Not Used)	(Not Used)	(Not Used)	Delay [12:8]					Rst Value xxx00000b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Scan Start Delay *(ScanStartDelay)* **$01FF8898**	Delay [7:0]								Rst Value 00h Read Value 00h

Bits 12-0 Delay[12:0]   1.) The ScanStartDelay register control the number of scan dots (pixels) that are to be skipped before valid scanner data is available. For example, if ScanStartDelay = 1, the first valid scanner pixel is dot number 2, the second dot after the beginning of the START pulse.

2.) ScanStart Delay Time = (Delay[12:0] +1) * (ScanDotPeriod).

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Start Edges *(StartEdges)* **$01FF889B**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Start Edges *(StartEdges)* **$01FF889A**	StartNeg Value (0-15) Start trailing edge position = (StartNeg+1)*(ScanDotScalerPeriod)				StartPos Value (0-15) Start leading edge position = (StartPos+1)*(ScanDotScalerPeriod)				Rst Value 00h Read Value 00h

Bits 7-4   StartNeg (0-15)   StartNeg establishes the location of the START signal trailing edge.

Bits 3-0   StartPos (0-15)   StartNeg establishes the location of the START signal leading edge.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Start Config *(StartConfig)* **$01FF889D**	Guardband Enable for scctrl[6]	Guardband Enable for scctrl[5]	Guardband Enable for scctrl[4]	Guardband[4:0] (in terms of pixel period)					Rst. Value 00h Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Start Config *(StartConfig)* **$01FF889C**	Offset[3:0] (in terms of pixel period)				Length[3:0] (in terms of pixel period)				Rst. Value xxxxxxx0b Read Value 00h

Bit 15-13   Guardband Enable for scctrl[6:4]

When a guardband is enabled, then the corresponding control signal will not change (will be "frozen") for the duration specified by "guardband" in bit 12-8.

Bit 12-8   Guardband[4:0]   Some scanner requires a quiet period while start pulse is active. Within this quiet period, only the start pulse will be toggling while other control signals are prevented from toggling. The duration of this period is determined by guardband * pixelperiod.

Bit 7-4   Offset[3:0]   The start pulse can be moved with respect to the start of scan cycle by this offset value which is expressed in terms of pixel period.

Bit 3-0   Length[3:0]   The duration of the start pulse is programmable according to this "Length" value.

Start pulse duration is [length + 1] * pixel_period

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clk2a Edges *(Clk2aEdges)* **$01FF889F**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Clk2a Edges *(Clk2aEdges)* **$01FF889E**	Clk2aNeg Value (0-15)				Clk2aPos Value (0-15)				Rst Value 00h Read Value 00h

Bits 7-4   Clk2aNeg   Clk2aNeg establishes the trailing edge of the Clk2a signal relative to the start of each dot period, and is specified in ScanDotScaler Clock periods.

Bits 3-0   Clk2aPos   Clk2aPos establishes the leading edge of the Clk2a signal relative to the start of each dot period, and is specified in ScanDotScaler Clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clk2b Edges *(Clk2bEdges)* **$01FF88A1**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Clk2b Edges *(Clk2bEdges)* **$01FF88A0**	Clk2bNeg Value (0-15)				Clk2bPos Value (0-15)				Rst Value 00h Read Value 00h

Bits 7-4   Clk2bNeg   Clk2bNeg establishes the trailing edge of the Clk2b signal relative to the start of each dot period, and is specified in ScanDotScaler Clock periods.

Bits 3-0   Clk2bPos   Clk2bPos establishes the leading edge of the Clk2b signal relative to the start of each dot period, and is specified in ScanDotScaler Clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clk2c Edges *(Clk2cEdges)* **$01FF88A3**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Clk2c Edges *(Clk2cEdges)* **$01FF88A2**	Clk2cNeg Value (0-15)				Clk2cPos Value (0-15)				Rst Value 00h Read Value 00h

Bits 7-4   Clk2cNeg   Clk2cNeg establishes the trailing edge of the Clk2c signal relative to the start of each dot period, and is specified in ScanDotScaler Clock periods.

Bits 3-0   Clk2cPos   Clk2cPos establishes the leading edge of the Clk2c signal relative to the start of each dot period, and is specified in ScanDotScaler Clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
ADCSample Configuration *(ADCSampleCfg)* **$01FF88A5**	Mode	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value 00xxxxxxb Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
ADC Sample Configuration *(ADCSampleCfg)* **$01FF88A4**	SampleNeg Value (0-15)				SamplePos Value (0-15)				Rst Value 00h Read Value 00h

Bits 15	Mode	This mode will only affect the number of sample pulses and not the number of sampled data to be stored into memory.
		0 = sampling continuously as long as ScanEnable = 1 1 = sampling valid pixels only
Bits 7-4	SampleNeg	SampleNeg establishes the trailing edge of the Sample signal relative to the start of each dot period, and is specified in ScanDotScaler clock periods.
Bits 3-0	SamplePos	SamplePos establishes the leading edge of the Sample signal relative to the start of each dot period, and is specified in ScanDotScaler clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clamp Control *(ClampCtrl)* **$01FF88A7**	Clamp Mode	Internal Clamp Enable	External Clamp Enable	External Clamp Guardband Enable	External Clamp Invert	(Not Used)	(Not Used)	(Not Used)	Rst Value 0xxxxxxxb Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Clamp Control *(ClampCtrl)* **$01FF88A6**	(Not Used)	ClampLengh[6:0]							Rst Value x0000000b Read Value 00h

Bit 15	ClampMode (1 = line–based, 0 = pixel–based)
	Setting the ClampMode bit causes line-based clamping to be used. Clearing this bit causes pixel-based clamping to be used.
	For line–based clamping, the clamp signal is enabled for the number of dots specified in ClampLength register, and can occur either before or after the valid pixel data, as selected by the ClampDelay register.
	For pixel–based clamping, the clamp signal runs continuously throughout the entire scanline.
Bits 6-0	ClampLength[6:0] This value specifies the number of pixels to be used for line–based clamping.

Note: Both internal and external clamp share the same control signals for 'shift' and 'delay'.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clamp Delay *(ClampDelay)* **$01FF88A9**	Clamp Position	(Not Used)	Clamp Delay[13:8]						Rst Value 0x000000b Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Clamp Delay *(ClampDelay)* **$01FF88A8**	Clamp Delay [7:0]								Rst Value 00h Read Value 00h

Bit 15   Clamp Position (only valid for line–based clamping)

Setting the ClampPosition bit causes the ClampDelay value to be applied from the start of the scanline (i.e., during the ScanStartDelay time). Clearing this bit causes the ClampDelay value to be applied from the *first valid pixel* of the scanline (i.e., after the ScanStartDelay has been accounted for).

Bits 13-0   Clamp Delay[13:0] (only valid for line–based clamping)

The ClampDelay register control the number of scan dots (pixels) that are to be skipped before enabling the clamp signal for the number of pixels specified in ClampLength register.

> **Note**: If ClampPosition bit = 0 and a new scanline is started before the ClampDelay value has been satisfied, the clamp signal will not activate. The same is true if ClampPosition bit = 1 and the ScanStartDelay value is less than the ClampDelay value.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clamp Edges *(ClampEdges)* **$01FF88AB**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Clamp Edges *(ClampEdges)* **$01FF88AA**	ClampNeg Value (0-15)				ClampPos Value (0-15)				Rst Value 00h Read Value 00h

Bits 7-4   ClampNeg   ClampNeg establishes the trailing edge of the Clamp signal relative to the start of each dot period, and is specified in ScanDotScaler clock periods.

Bits 3-0   ClampPos   ClampPos establishes the leading edge of the Clamp signal relative to the start of each dot period, and is specified in ScanDotScaler clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Clk2d Control *(Clk2dCtrl)* $01FF88C5	Select	(Not Used)	(Not Used)	Shift	Delay[1]	Delay[0]	Invert	Guardband Enable	Rst. Value xxx00000b Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Clk2d Control *(Clk2dCtrl)* $01FF88C4		Negedge[3:0]				PosEdge[3:0]			Rst. Value x0h Read Value 00h

Bit 15　　Select　　　　　　　　0 = selecting sample signal
　　　　　　　　　　　　　　　1 = clk2d

Bit 12　　Shift　　　　　　　　　0 = clk2d synchronous to the rising edge of ihsclk
　　　　　　　　　　　　　　　1 = to the falling edge of ihsclk

Bit 11-10　Delay　　　　　　　　programmable delay amount
　　　　　　　　　　　　　　　0 = no delay
　　　　　　　　　　　　　　　1 = 1 unit delay
　　　　　　　　　　　　　　　2 = 2 unit delay
　　　　　　　　　　　　　　　3 = 3 unit delay

Bit 9　　Invert　　　　　　　　0 = normal
　　　　　　　　　　　　　　　1 = invert)

Bit 8　　Guardband Enable　　0 = no guardband
　　　　　　　　　　　　　　　1 = apply guardband

Bit 7-4　NegEdge position

Bit 3-0　PosEdge position

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LEDCtrl[0] Edges *(LED0Edges)* 01FF88AD				NegEdge[7:0] Trailing edge position = (NegEdge[7:0] * 64) pixels before/after scan start delay					Rst. Value 00h Read Value 00h
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LEDCtrl[0] Edges *(LED0Edges)* 01FF88AC				PosEdge[7:0] Leading edge position = (PosEdge[7:0] * 64) pixels before/after scan start delay					Rst. Value 00h Read Value 00h

Bits 15-8　NegEdge[7:0]　　　　NegEdge establishes the trailing edge of the envelope for LEDCtrl signal and is specified in 64 pixel clock periods.

Bits 7-0　PosEdge[7:0]　　　　PosEdge establishes the leading edge of the envelope for LEDCtrl signal and is specified in 64 pixel clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LEDCtrl[1] Edges *(LED1Edges)* **01FF88AF**	NegEdge[7:0] Trailing edge position = (NegEdge[7:0] * 64) pixels before/after scan start delay								Rst. Value 00h Read Value 00h
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LEDCtrl[1] Edges *(LED1Edges)* **01FF88AE**	PosEdge[7:0] Leading edge position = (PosEdge[7:0] * 64) pixels before/after scan start delay								Rst. Value 00h Read Value 00h

Bits 15-8   NegEdge[7:0]   NegEdge establishes the trailing edge of the envelope for LEDCtrl signal and is specified in 64 pixel clock periods.

Bits 7-0   PosEdge[7:0]   PosEdge establishes the leading edge of the envelope for LEDCtrl signal and is specified in 64 pixel clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LEDCtrl[2] Edges *(LED2Edges)* **01FF88B1**	NegEdge[7:0] Trailing edge position = (NegEdge[7:0] * 64) pixels before/after scan start delay								Rst. Value 00h Read Value 00h
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LEDCtrl[2] Edges *(LED2Edges)* **01FF88B0**	PosEdge[7:0] Leading edge position = (PosEdge[7:0] * 64) pixels before/after scan start delay								Rst. Value 00h Read Value 00h

Bits 15-8   NegEdge[7:0]   NegEdge establishes the trailing edge of the envelope for LEDCtrl signal and is specified in 64 pixel clock periods.

Bits 7-0   PosEdge[7:0]   PosEdge establishes the leading edge of the envelope for LEDCtrl signal and is specified in 64 pixel clock periods.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LED0 PWM Config *(LED0PWM)* **$01FF88B3**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	clock divider[2:0]			Rst. Value xxxxx000b Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LED0 PWM Config *(LED0PWM)* **$01FF88B2**	duty cycle[7:0]								Rst. Value 00h Read Value 00h

Bits 10-8   clock divider[2:0]   The clock used for PWM is scaled according to "clock divider" setting.

Bits 7-0   duty cycle[7:0]   The period of PWM is divided into 256. The high time of PWM is determined by "duty cycle".

Min frequency (clk_div = 7)
ihsclk = 50 ns
pwm clk = T(ihsclk) * (clk_div + 1) = 50 ns * (7 + 1) = 400 ns
pwm period = 400 ns * 256 = 102,400 ns = 102.4 us (9.8 KHz)

Max frequency (clk_div = 0)
ihsclk = 50 ns
pwm clk = ihsclk = 50 ns
pwm period = 50 ns * 256 = 12.8 us (78.1 KHz)

> **Note**: In order to have a non-modulated signal, the duty cycle must be set to 0xFF.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LED1 PWM Config *(LED1PWM)* **$01FF88B5**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	clock divider[2:0]			Rst. Value xxxxx000b Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LED1 PWM Config *(LED1PWM)* **$01FF88B4**	duty cycle[7:0]								Rst. Value 00h Read Value 00h

Bits 10-8   clock divider[2:0]   The clock used for PWM is scaled according to "clock divider" setting.

Bits 7-0   duty cycle[7:0]   The period of PWM is divided into 256. The high time of PWM is determined by "duty cycle".

> **Note**: In order to have a non-modulated signal, the duty cycle must be set to 0xFF.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LED2 PWM Config *(LED2PWM)* **$01FF88B7**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	clock divider[2:0]			Rst. Value xxxxx000b Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LED2 PWM Config *(LED2PWM)* **$01FF88B6**	duty cycle[7:0]								Rst. Value 00h Read Value 00h

Bits 10-8   clock divider[2:0]   The clock used for PWM is scaled according to "clock divider" setting.

Bits 7-0   duty cycle[7:0]   The period of PWM is divided into 256. The high time of PWM is determined by "duty cycle".

> **Note**: *In order to have a non-modulated signal, the duty cycle must be set to 0xFF.*

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
LEDCtrl Config *(LEDConfig)* **$01FF88B9**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xx000111b Read Value 07h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
LEDCtrl Config *(LEDConfig)* **$01FF88B8**	(Not Used)	(Not Used)	LEDCtrl[2] sync_select	LEDCtrl[1] sync_select	LEDCtrl[0] sync_select	LEDCtrl[2] position	LEDCtrl[1] position	LEDCtrl[0] position	Rst. Value xx000000b Read Value 00h

Bits 5-3   LEDCtrl[2:0] sync_select   1 = sync is selected, ignoring LED PWM config & LED Edges settings
                                              0 = normal LEDCtrl

                                              When sync_select bit is set and the LED is enabled (by setting the corresponding bit in ScanCtrl register), then the LEDCtrl will be active from the beginning of scancycle until the end of scancycle. In this mode of operation, the corresponding LED PWM & LED Edges settings are ignored. Once the LED is disabled, then at the end of scancycle, the LEDCtrl will not be active anymore. In other words, by setting sync_select bit, the LEDCtrl is outputing the LED enabled which is synchronized with the cyclestart.

                                              When sync_select bit is clear and the LED is enabled, then the LEDCtrl operates in normal mode, which depends on the LED PWM & LED Edges settings.

Bits 2-0   LEDCtrl[2:0] position   1 = before ScanStartDelay
                                              0 = after ScanStartDelay

                                              This position setting only affects the leading edge of LEDCtrl envelope. If the *place* bit is reset, then the LEDCtrl will be activated after valid pixel. Otherwise, if the *place* bit is set, then LEDCtrl can be activated right after scancycle.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan IA Config *(ScanIAConfig)* **$01FF88BB**	ADC enable	(Not Used)	OffsetSel[5]	OffsetSel[4]	OffsetSel[3]	OffsetSel[2]	OffsetSel[1]	OffsetSel[0]	Rst. Value xxx00000b Read Value 00h

Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Scan IA Config *(ScanIAConfig)* **$01FF88BA**	(Not Used)	(Not Used)	GPO on EV_VD[5]	GPO on EV_VD[4]	GainSel[3]	GainSel[2]	GainSel[1]	GainSel[0]	Rst. Value x0h Read Value 00h

In order to apply the content of this register to the analog die, the content must be sent serially by writing to the 'ADCSerialCmd' register. Please refer to the 'ADCSerialCmd' register description for more information.

Bit 15   ADC enable   This bit is used to enable the internal adc. The ADC circuit by default is disabled.

Bit 12-8   OffsetSel[4:0]   An analog subtracter is used in conjunction with OffsetSel in order to remove the DC offset from scanner signal.

Bit 5   GPO on EV_VD[5]   When 'VSC on EV bus' bit, i.e., bit 2 of VSCCtrl register, is set, then EV_VD[7:0] bus will function as output.. EV_VD[5] output is connected to this bit to function as general purpose output (GPO).

Bit 4   GPO on EV_VD[4]   When 'VSC on EV bus' bit, i.e., bit 2 of VSCCtrl register, is set, then EV_VD[7:0] bus will function as output.. EV_VD[4] output is connected to this bit to function as general purpose output (GPO).

Bit 3-0   GainSel[3:0]   GainSel is connected to a programmable gain amplifier to adjust the gain of the scanner signal.

Note: This register is double-buffered. The value written into this register will not be applied to SIA immediately. Rather, the value will be applied after the start of scancycle. Reading this register reflects the value applied to SIA.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
ADCSerialCmd *(ADCSerialCmd)* **$01FF854F**	Send Status (read only)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
ADCSerialCmd *(ADCSerialCmd)* **$01FF854E**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(reserved) '0'	Command Type	Transmit Mode[1]	Transmit Mode[0]	Rst. Value xxxx0000b Read Value 00h

Bit 15   Send Status (read only)   This bit indicates one of the following statuses:

                                                     waiting for an event to occur, as selected by 'transmit mode[1:0]', prior to sending the serial command
                                                     sending serial command is in progress.

                                                     The wait for the event can be aborted by clearing 'scanenable' bit on 'ScanConfig' register, and the command will not be sent.

Bit 15   reserved   It is important to always write '0' to this bit.

Bit 2   Command Type

Command Type	Function
0	To send 'offset' and 'gain' settings as stored in 'ScanIAConfig' register
1	To send 'ADC enable' bit as stored in 'ScanIAConfig' register

Conexant

Bit 1-0   Transmit Mode[1:0]

Prior to writing to this register, check the 'Send Status' bit, and make sure that this bit is clear. If this bit is set, then writing to this register will be ignored. If this bit is clear, then writing to this register will cause the following action based on the 'Transmit Mode':

Transmit Mode	Time to send the serial data
0	right away
1	at the beginning of a scan cycle; no 'scanonce' operation is involved
2	at the end of valid pixel (i.e. the last valid pixel in a line); 'scanonce' operation is involved
3	at the next msint

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Scan Control Delay (ScanCtrlDelay) $01FF88BD	sample delay[1:0]		Scctrl[6] Delay[1:0]		scctrl[5] delay[1:0]		scctrl[4] delay[1:0]		Rst. Value xx000000b Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Scan Control Delay (ScanCtrlDelay) $01FF88BC	scctrl[3] delay[1:0]		Scctrl[2] Delay[1:0]		scctrl[1] delay[1:0]		clamp delay[1:0]		Rst. Value 00h Read Value 00h

Bit 15-14	sample delay	This setting controls the delay for 'sample' signal.
Bit 13-12	scctrl[6] delay	This setting controls the delay for 'scctrl[6]' signal.
Bit 11-10	scctrl[5] delay	This setting controls the delay for 'scctrl[5]' signal.
Bit 9-8	scctrl[4] delay	This setting controls the delay for 'scctrl[4]' signal.
Bit 7-6	scctrl[3] delay	This setting controls the delay for 'scctrl[3]' signal.
Bit 5-4	scctrl[2] delay	This setting controls the delay for 'scctrl[2]' signal.
Bit 3-2	scctrl[1] delay	This setting controls the delay for 'scctrl[1]' signal.
Bit 1-0	clamp delay	This setting controls the delay for 'clamp' signal.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
ADC Config (ADCConfig) $01FF88BF	AdclkNeg Value (0-15)				AdclkPos Value (0-15)				Rst. Value 00h Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
ADC Config (ADCConfig) $01FF88BE	Digital Data Capture Position [3:0] (0-15)				Data Latency[3:0] (0-15)				Rst. Value 00h Read Value 00h

Bit 15-12   AdclkNeg   (only valid when internal ADC is selected)

AdclkNeg establishes the trailing edge of the adclk signal relative to the start of each dot period, and is specified in terms of ScanDotScaler clock periods.

Bit 11-8   AdclkPos   (only valid when internal ADC is selected)

AdclkPos establishes the leading edge of the adclk signal relative to the start of each dot period, and is specified in terms of ScanDotScaler clock periods.

Bit 7-4   Digital Data Capture Position   (valid for external ADC only)

This setting indicates where the digital data from ADC should be captured within a pixel period, and is specified in terms of ScanDotScaler clock periods.

Bit 3-0   Data Latency   (valid with either internal or external ADC)

This setting indicates the latency or pipeline delay as required by ADC, and is expressed in the number of pixel clocks.

Important note on how to program adclk & adsample when an internal ADC is selected:

1. *The location and the width of adsample must be established first according to the scanner requirement (the analog signal is being sampled while adsample = 1, and at the falling edge of adsample, the signal is stored into an internal capacitor. While adsample = 0, the signal value is being held in the capacitor.)*

2. *Once adsample has been programmed properly, the adclk must be configured according to the following rule:*

    a) adclk must be low while adsample is high

7.1.3   Timing
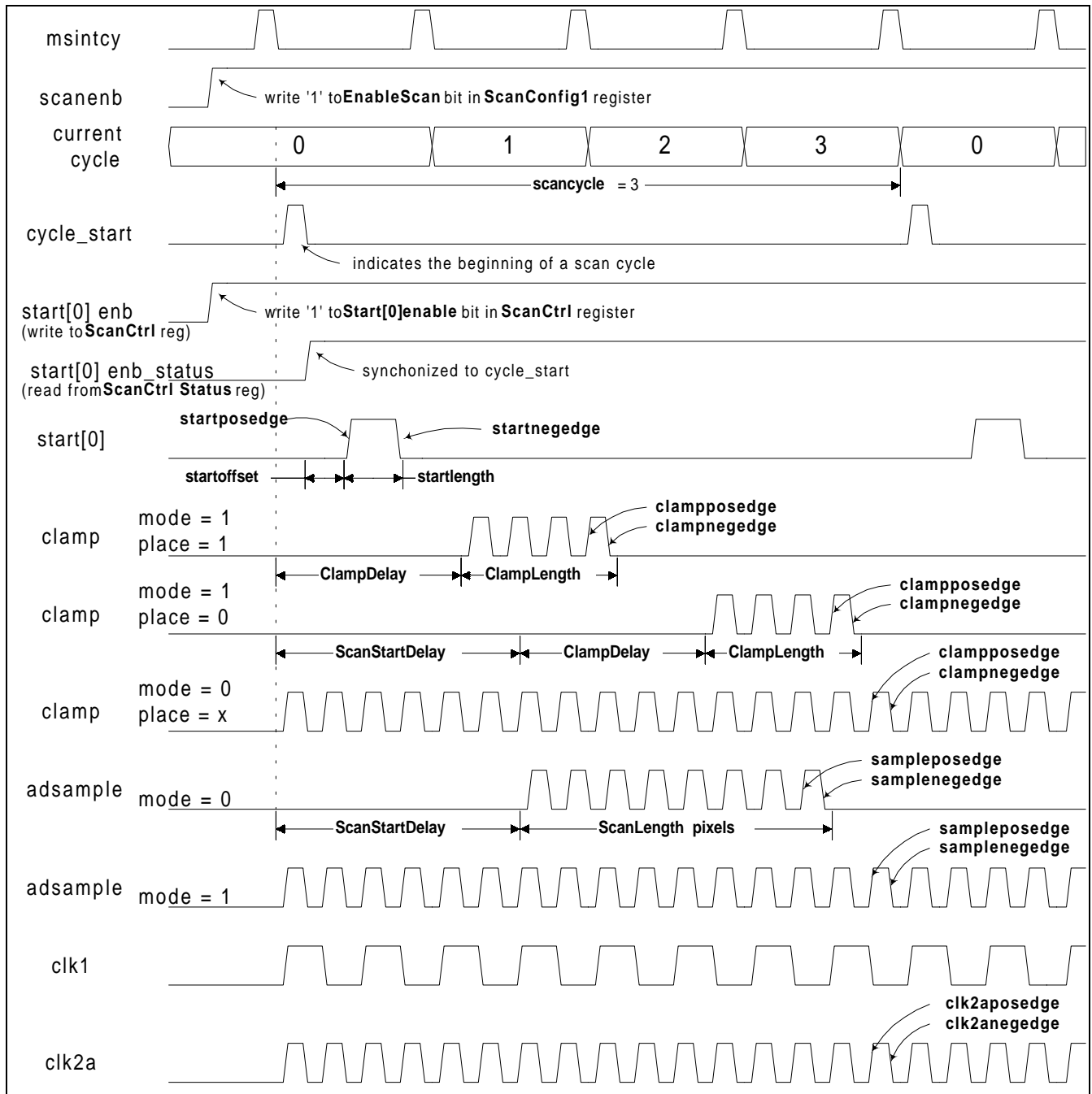
Figure 7-2. Untitled Timing Diagram
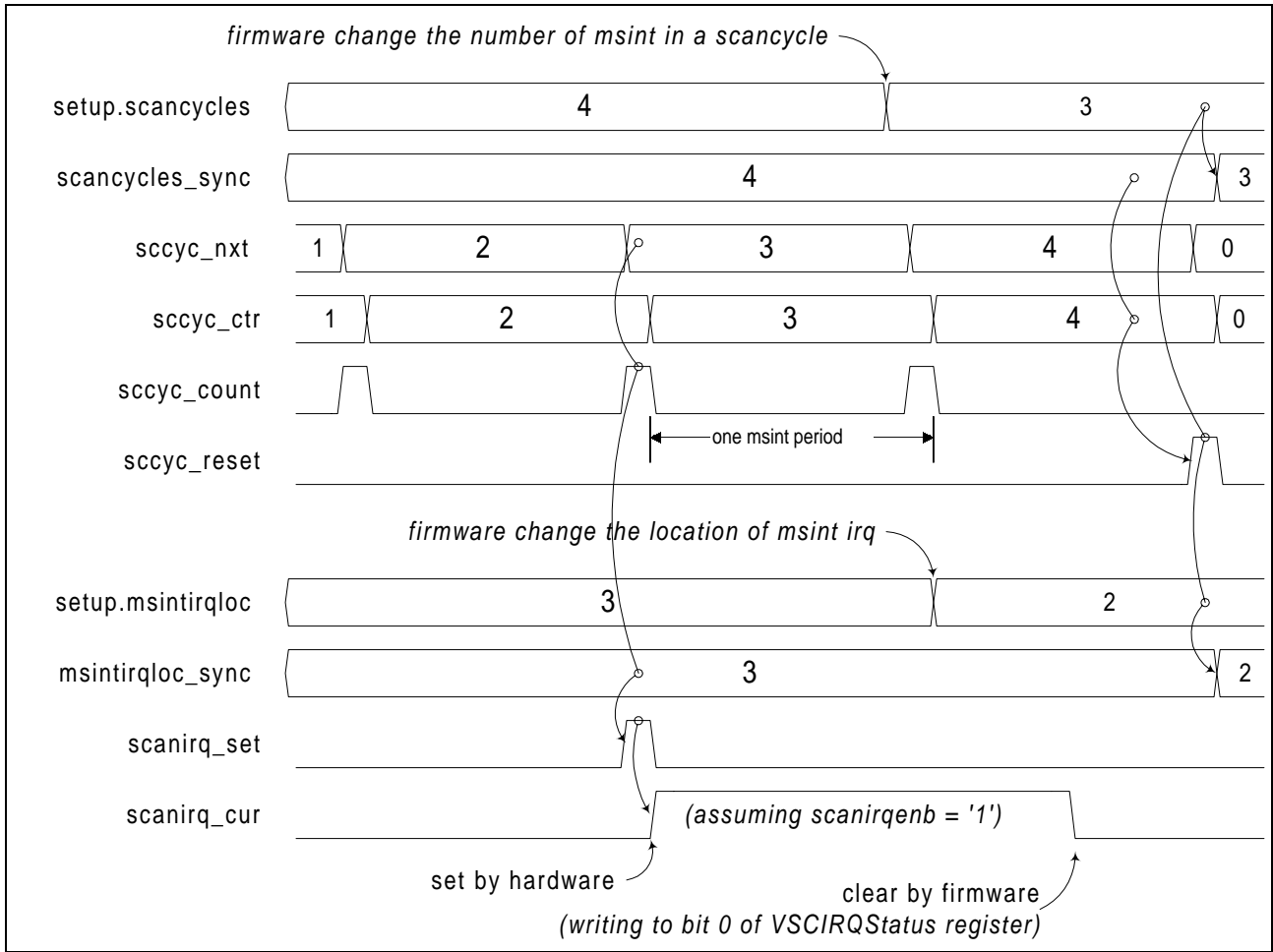
Figure 7-3. Untitled Timing Diagram
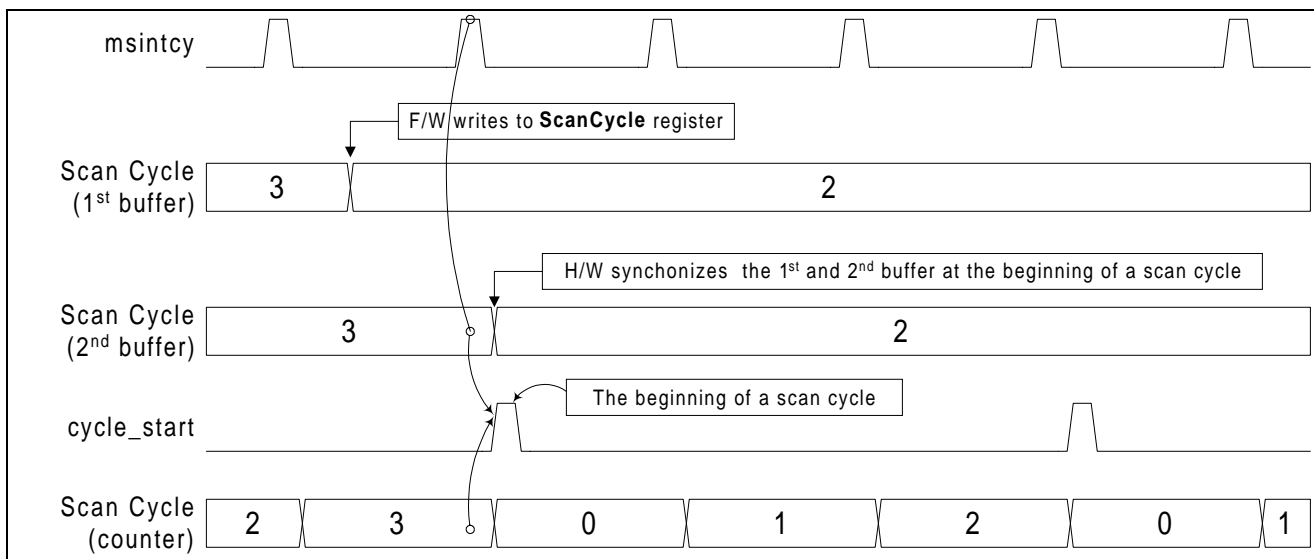
Figure 7-4. Untitled Timing Diagram

Conexant
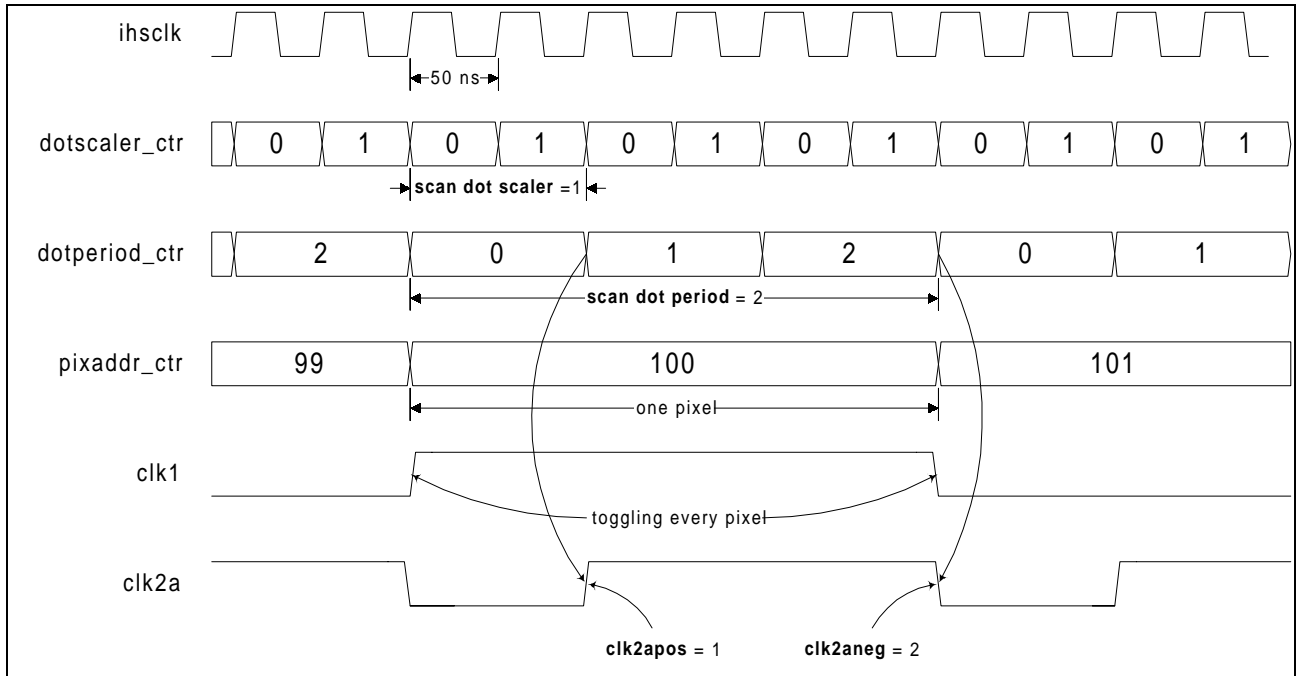
Figure 7-5. Untitled Timing Diagram

*Note: The rising and falling edges timing of **clk2a** also applies to **clk2b**, **clk2c**, **start**, **clamp**, and **adsample**.*
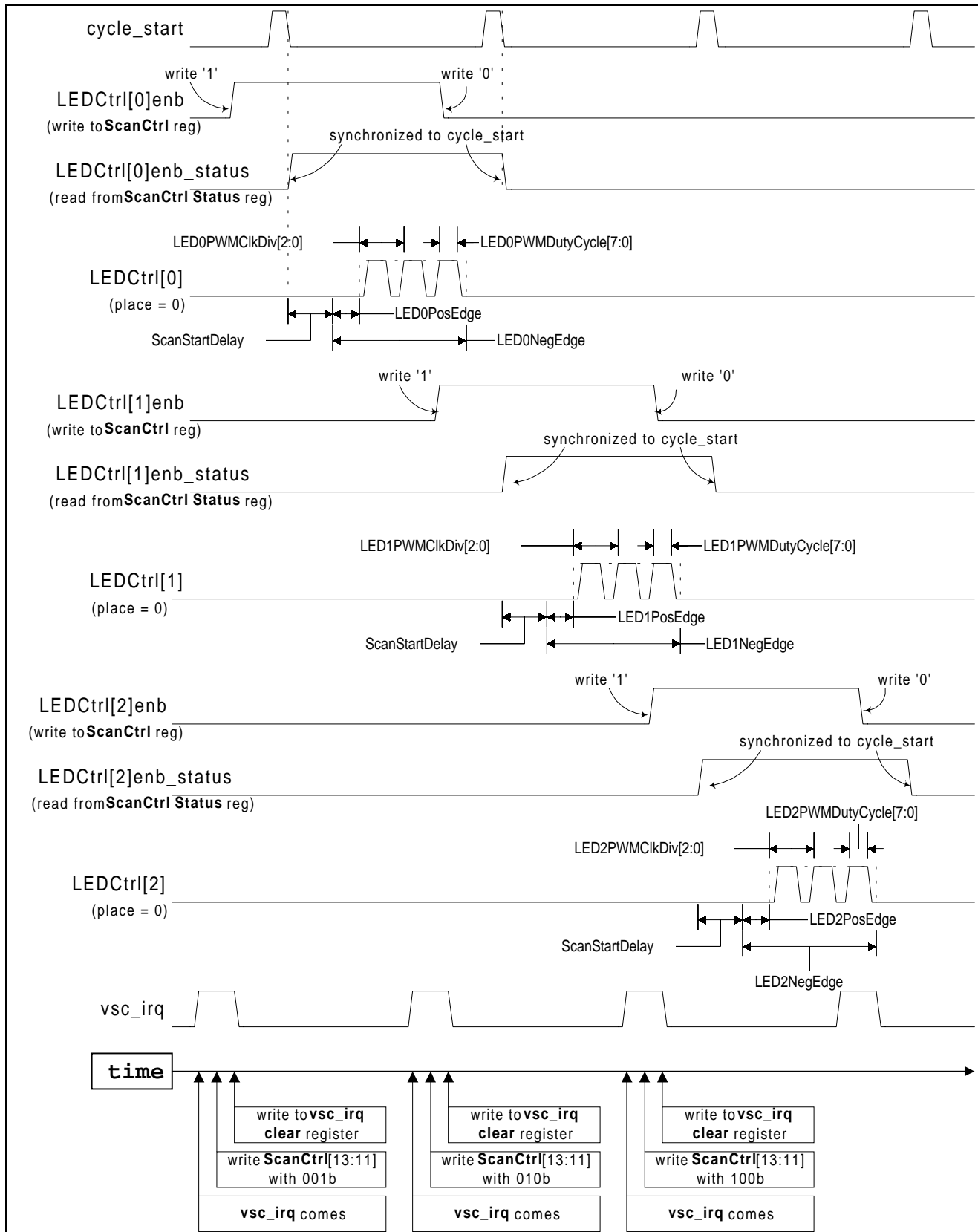
[led-time.vsd]

Figure 7-6. Untitled Timing Diagram
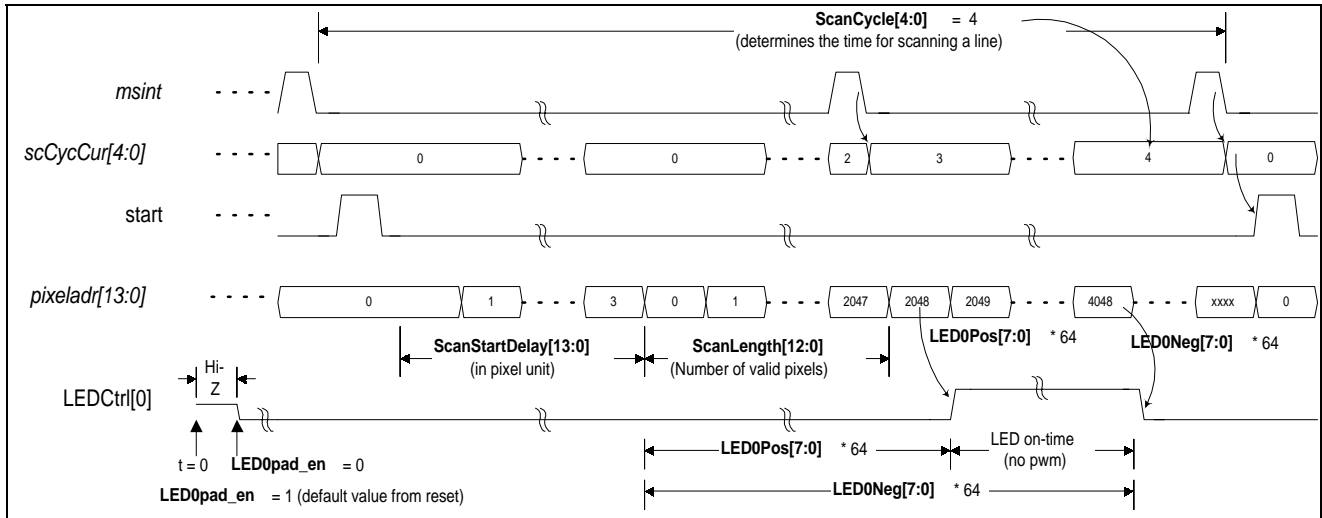
Figure 7-7. Untitled Timing Diagram

Notes:

1. *MSInt_period = (msintPeriod + 1) * 125 us*
   *msintPeriod[5:0] is set in MSINTPeriod register (assuming that value in **125usPrescaler** register has been adjusted to obtain 125 us period)*

2. *ScanCycle_period = (ScanCycle + 1) * MSInt_period*

3. *ScanCycle in conjunction with MSIntPeriod must be selected so that:*
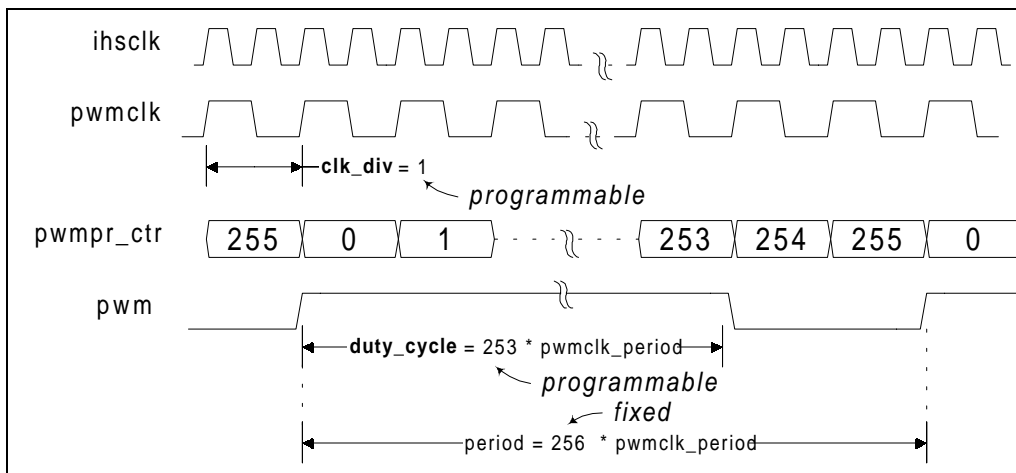
Figure 7-8. Untitled Timing Diagram
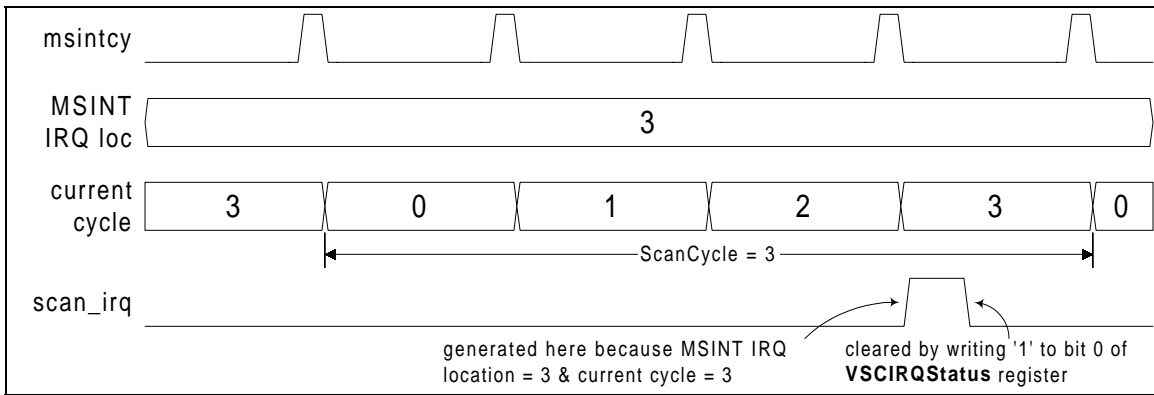
Figure 7-9. Untitled Timing Diagram

The scanner controller will issue adc_start, adc_drdy, and adc_stop as follows:
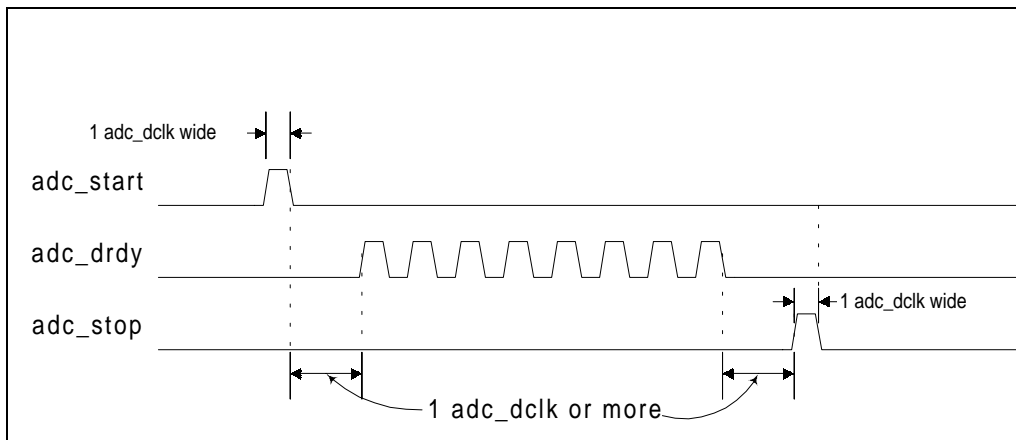
Figure 7-10. Untitled Timing Diagram

External ADC timing summary

National: ADC 12662
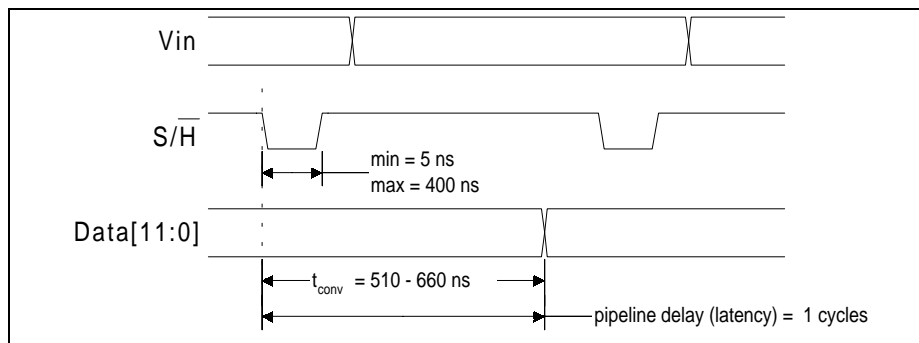
[adc-national-adc12662.vsd]

Figure 7-11. Untitled Timing Diagram

Analog: AD 9220
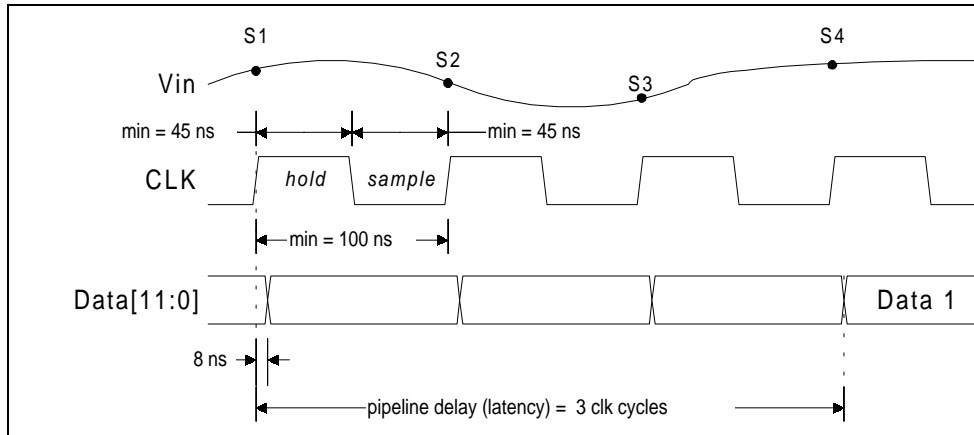
Figure 7-12. Untitled Timing Diagram

Linear: LTC 1412
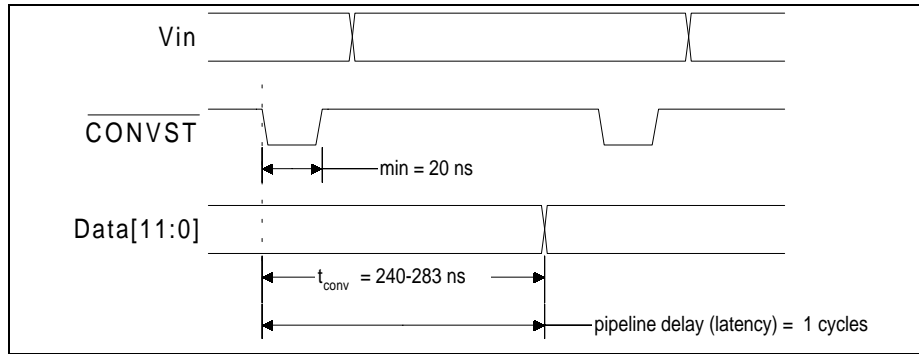
Figure 7-13. Untitled Timing Diagram

7.1.4   Firmware Operation

In order to obtain the optimal digital data, the dynamic range of analog signal must be maximized. In order to maximize the dynamic range of scanner signal, the duration of the light exposure on the sensor must be just right. If it is too short, then the signal will be too low; and if it is too long, then the signal will be saturated. The amount of light exposure can be controlled by having programmable pulse width modulation on the LED control. Typically, the signal from the sensor has a dc offset. This offset must be removed, and the resulting signal can be amplified to restore the dynamic range.

Examples of how to setup the registers for various scanners are shown below. These setups are configured for the fastest scanning operation. (only setup, not operation registers are shown)

> *Notes*:
>
> *1.) The value for scan cycle depends on how* ***msint*** *signal is configured.*
>
> *2.) The LED setting may need adjustment according to the proper light intensity for different color.*

Manufacturer	ROHM
Part #	IA3008 – ZE22
Type	CIS
Resolution	300 dpi
Pixels/line	2560
Scan time (color)	9 ms/line
Scan time (B/W)	1.8~4.5 ms/line
Data Rate	1.5 MHz (max)
Main clock 'H' duty	25 – 50 %

Interface:

Scanner	Description	MFC2000
R$_{GND}$	Red LED control	LEDCtrl[0]
G$_{GND}$	Green LED control	LEDCtrl[1]
B$_{GND}$	Blue LED control	LEDCtrl[2]
SI	Start pulse	start[0]
CLK	Main clock	clk2a
Ao	Scanner signal	vin
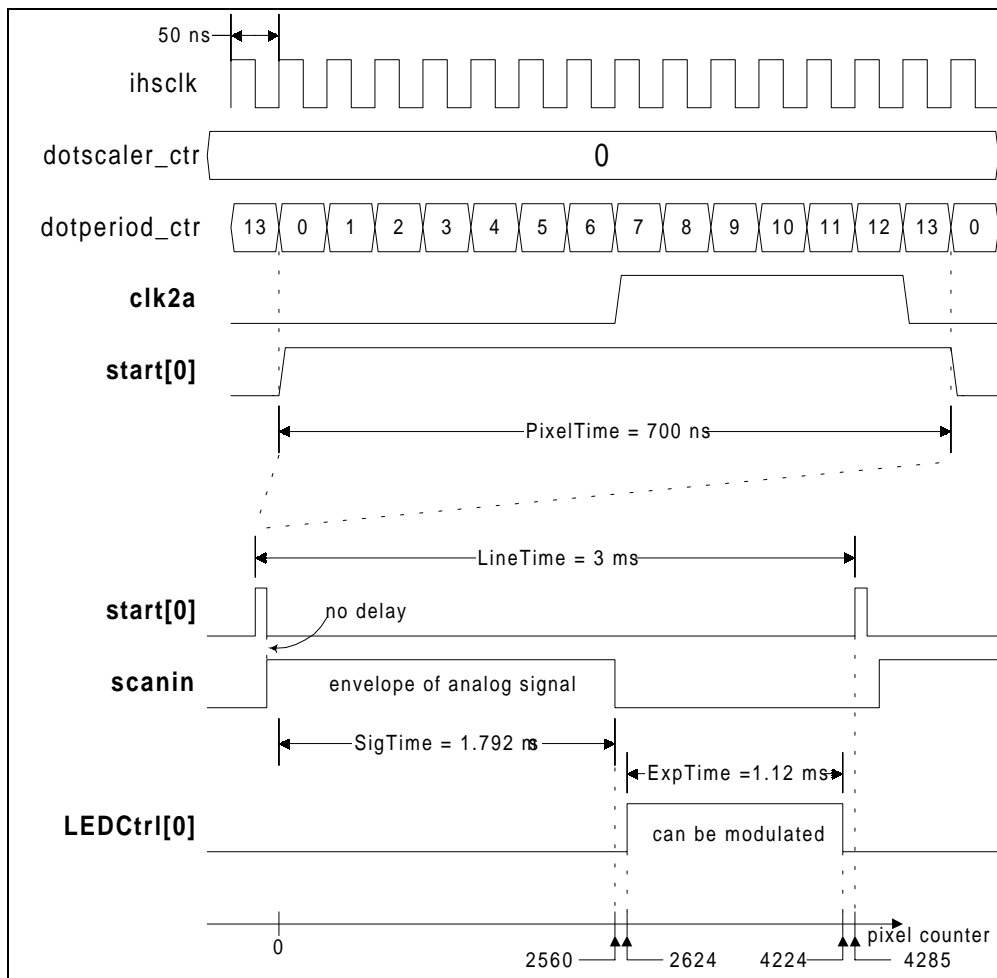
Timing:

[Rohm–IA3008–ZE22.vsd]

Figure 7-14. Untitled Timing Diagram

Table 7-1. Register setup for Rohm–IA3008–ZE22

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0
	scctrl[6:2] select	xx100b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	13
StartConfig	GuardbandEnb[6:4]	0
	Guardband[4:0]	n/a
	Offset[3:0]	0
	Length[3:0]	0
StartEdges	StartPos[3:0]	0
	StartNeg[3:0]	13
ScanStartDelay	Delay[12:0]	0
Clk2aEdges	Clk2aPos[3:0]	7
	Clk2aNeg[3:0]	12
Clk2bEdges	Clk2bPos[3:0]	n/a
	Clk2bNeg[3:0]	n/a
Clk2cEdges	Clk2cPos[3:0]	n/a
	Clk2cNeg[3:0]	n/a
ADCSampleCfg	SamplePos[3:0]	9
	SampleNeg[3:0]	11
ADCConfig	AdclkPos[3:0]	0
	AdclkNeg[3:0]	6
	DataPlace[3:0]	10
	DataLatency[3:0]	
ClampCtrl	ClampMode	n/a
	ClampEnb	0
	ClampLength[6:0]	n/a
ClampDelay	ClampPosition	n/a
	ClampDelay[13:0]	n/a
ClampEdges	ClampPos[3:0]	n/a
	ClampNeg[3:0]	n/a
LEDConfig	LEDCtrl sync_sel[2:0]	0
	LEDCtrl place[2:0]	0
LED0Edges	LED0Pos[7:0]	41
	LED0Neg[7:0]	66
LED1Edges	LED1Pos[7:0]	41
	LED1Neg[7:0]	66
LED2Edges	LED2Pos[7:0]	41
	LED2Neg[7:0]	66
ScanLength	ScanLength[12:0]	2560
ScanCyles	ScanCycle[4:0]	*3 ms*

Manufacturer	DYNA
Part #	DL507–07UAH
Type	CIS
Resolution	300 dpi
Pixels/line	2552
Scan time (color)	7.5 ms/line
Scan time (B/W)	
Data Rate	2.0 MHz (typ)
Main clock 'H' duty	25 % (typ)

Interface:

Scanner	Description	MFC2000
LEDR	Red LED control	LEDCtrl[0]
LEDG	Green LED control	LEDCtrl[1]
LEDB	Blue LED control	LEDCtrl[2]
SI	Start pulse	start[0]
CLK	Main clock	clk2a
SIG	Scanner signal	vin
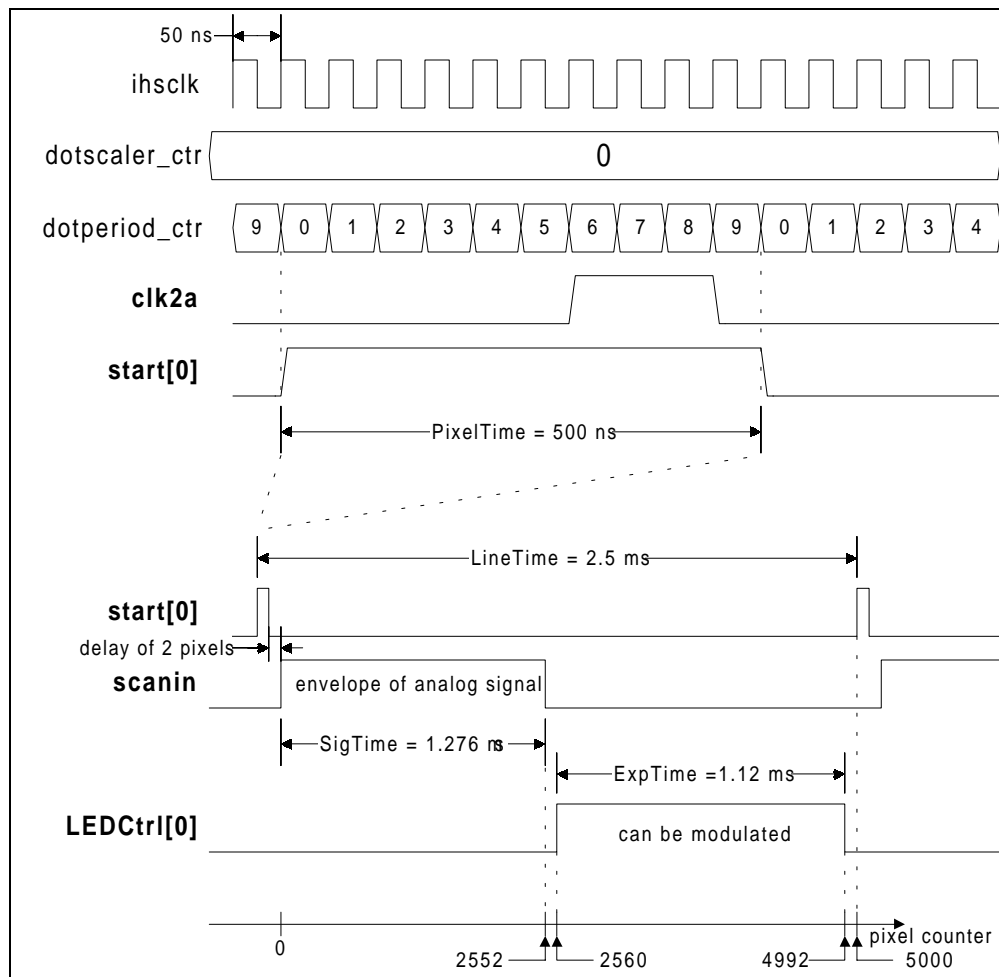
Timing:

Figure 7-15. Untitled Timing Diagram

Conexant

Table 7-2. Register setup for Dyna–DL507–07UAH

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0
	scctrl[6:2] select	xx100b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	9
StartConfig	GuardbandEnb[6:4]	0
	Guardband[4:0]	n/a
	Offset[3:0]	0
	Length[3:0]	0
StartEdges	StartPos[3:0]	0
	StartNeg[3:0]	9
ScanStartDelay	Delay[12:0]	0
Clk2aEdges	Clk2aPos[3:0]	6
	Clk2aNeg[3:0]	8
Clk2bEdges	Clk2bPos[3:0]	n/a
	Clk2bNeg[3:0]	n/a
Clk2cEdges	Clk2cPos[3:0]	n/a
	Clk2cNeg[3:0]	n/a
ADCSampleCfg	SamplePos[3:0]	
	SampleNeg[3:0]	
ADCConfig	AdclkPos[3:0]	
	AdclkNeg[3:0]	
	DataPlace[3:0]	
	DataLatency[3:0]	
ClampCtrl	ClampMode	n/a
	ClampEnb	0
	ClampLength[6:0]	n/a
ClampDelay	ClampPosition	n/a
	ClampDelay[13:0]	n/a
ClampEdges	ClampPos[3:0]	n/a
	ClampNeg[3:0]	n/a
LEDConfig	LEDCtrl sync_sel[2:0]	0
	LEDCtrl place[2:0]	0
LED0Edges	LED0Pos[7:0]	40
	LED0Neg[7:0]	78
LED1Edges	LED1Pos[7:0]	40
	LED1Neg[7:0]	78
LED2Edges	LED2Pos[7:0]	40
	LED2Neg[7:0]	78
ScanLength	ScanLength[12:0]	2552
ScanCyles	ScanCycle[4:0]	*2.5 ms*

Manufacturer	Mitsubishi
Part #	GT3R216
Type	CIS
Resolution	300 dpi
Pixels/line	2552
Scan time (color)	7.5 ms/line
Scan time (B/W)	1.5 ms/line
Data Rate	3 – 4 MHz
Main clock 'H' duty	20 – 30 %

Interface:

Scanner	Description	MFC2000
LEDr	Red LED control	LEDCtrl[0]
LEDg	Green LED control	LEDCtrl[1]
LEDb	Blue LED control	LEDCtrl[2]
SI	Start pulse	start[0]
CLK	Main clock	clk2a
SIG	Scanner signal	vin
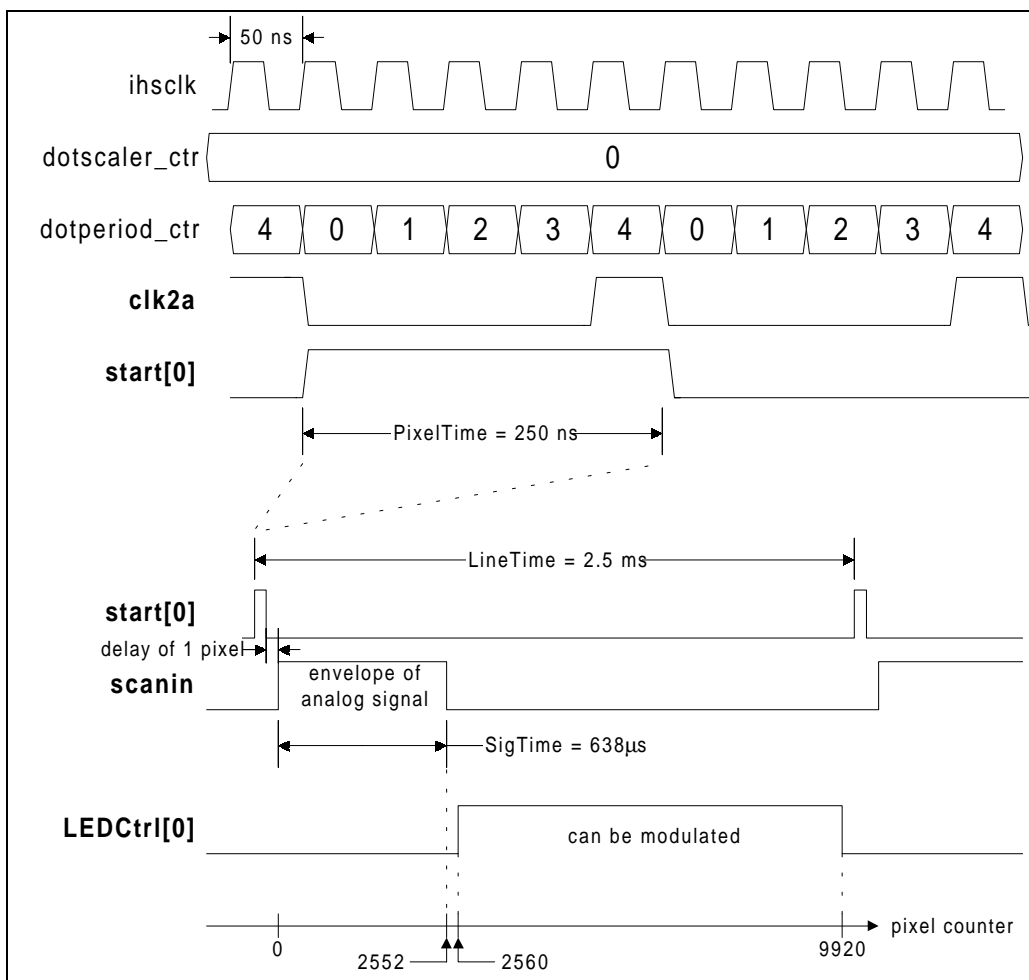
Timing:

Figure 7-16. Untitled Timing Diagram

Table 7-3. Register setup for Mitsubishi-GT3R216

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0
	scctrl[6:2] select	xx100b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	4
StartConfig	GuardbandEnb[6:4]	0
	Guardband[4:0]	n/a
	Offset[3:0]	0
	Length[3:0]	0
StartEdges	StartPos[3:0]	0
	StartNeg[3:0]	4
ScanStartDelay	Delay[12:0]	
Clk2aEdges	Clk2aPos[3:0]	4
	Clk2aNeg[3:0]	4
Clk2bEdges	Clk2bPos[3:0]	n/a
	Clk2bNeg[3:0]	n/a
Clk2cEdges	Clk2cPos[3:0]	n/a
	Clk2cNeg[3:0]	n/a
ADCSampleCfg	SamplePos[3:0]	
	SampleNeg[3:0]	
ADCConfig	AdclkPos[3:0]	
	AdclkNeg[3:0]	
	DataPlace[3:0]	
	DataLatency[3:0]	
ClampCtrl	ClampMode	n/a
	ClampEnb	0
	ClampLength[6:0]	n/a
ClampDelay	ClampPosition	n/a
	ClampDelay[13:0]	n/a
ClampEdges	ClampPos[3:0]	n/a
	ClampNeg[3:0]	n/a
LEDConfig	LEDCtrl sync_sel[2:0]	0
	LEDCtrl place[2:0]	0
LED0Edges	LED0Pos[7:0]	40
	LED0Neg[7:0]	155
LED1Edges	LED1Pos[7:0]	40
	LED1Neg[7:0]	155
LED2Edges	LED2Pos[7:0]	40
	LED2Neg[7:0]	155
ScanLength	ScanLength[12:0]	2552
ScanCyles	ScanCycle[4:0]	*2.5 ms*

Manufacturer	TOSHIBA
Part #	CIPS218MC300
Type	CIS pkg w/ CCD
Resolution	300 dpi
Pixels/line	2576
Scan time (color)	
Scan time (B/W)	
Data Rate	0.1 – 2.5 MHz
Main clock 'H' duty	50 %

Interface:

Scanner	Description	MFC2000
•LEDR	Red LED control	LEDCtrl[0]
•LEDG	Green LED control	LEDCtrl[1]
•LEDB	Blue LED control	LEDCtrl[2]
•TR	Start pulse	start[0]
•M	Main clock	clk2a
•RS	Clock	clk2b
•RNC	Clock	clk2c

Timing:
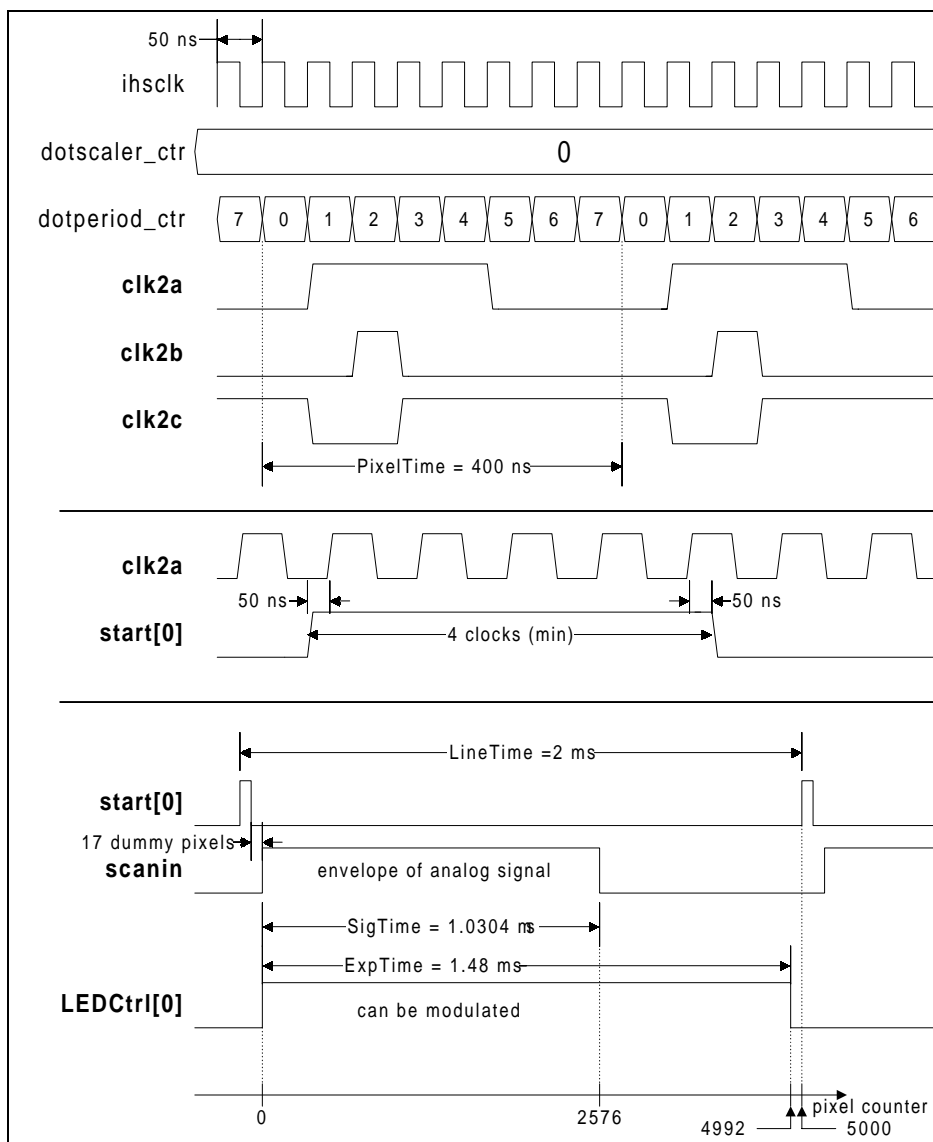
Figure 7-17. Untitled Timing Diagram

Table 7-4. Register Setup for Toshiba–CIPS218MC300

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0000000b
	scctrl[6:2] select	11100b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	7
StartConfig	GuardbandEnb[6:4]	0
	Guardband[4:0]	n/a
	Offset[3:0]	1
	Length[3:0]	4
StartEdges	StartPos[3:0]	0
	StartNeg[3:0]	1
ScanStartDelay	Delay[12:0]	16
Clk2aEdges	Clk2aPos[3:0]	1
	Clk2aNeg[3:0]	4
Clk2bEdges	Clk2bPos[3:0]	2
	Clk2bNeg[3:0]	2
Clk2cEdges	Clk2cPos[3:0]	3
	Clk2cNeg[3:0]	0
ADCSampleCfg	SamplePos[3:0]	
	SampleNeg[3:0]	
ADCConfig	AdclkPos[3:0]	
	AdclkNeg[3:0]	
	DataPlace[3:0]	
	DataLatency[3:0]	
ClampCtrl	ClampMode	1
	ClampEnb	1
	ClampLength[6:0]	16
ClampDelay	ClampPosition	1
	ClampDelay[13:0]	0
ClampEdges	ClampPos[3:0]	7
	ClampNeg[3:0]	7
LEDConfig	LEDCtrl sync_sel[2:0]	0
	LEDCtrl place[2:0]	0
LED0Edges	LED0Pos[7:0]	0
	LED0Neg[7:0]	78
LED1Edges	LED1Pos[7:0]	0
	LED1Neg[7:0]	78
LED2Edges	LED2Pos[7:0]	0
	LED2Neg[7:0]	78
ScanLength	ScanLength[12:0]	2576
ScanCyles	ScanCycle[4:0]	*2 ms*

Manufacturer	NEC
Part #	•PD3724
Type	CCD
Resolution	300 dpi
Pixels	2700 x 3
Scan time (color)	
Scan time (B/W)	
Data Rate	3 MHz (max)
Main clock 'H' duty	50 %

Interface:

Scanner	Description	MFC2000
	LED control	LEDCtrl[0]
•TG[3:1]	Start pulse	start[0]
SEL1	Color selector1	LEDCtrl[1]
SEL2	Color selector2	LEDCtrl[2]
•RB	Reset clock	clk2a
•1, •1L	Shift register clock 1	clk1
•2•••2L	Shift register clock 2	clk1

Timing:
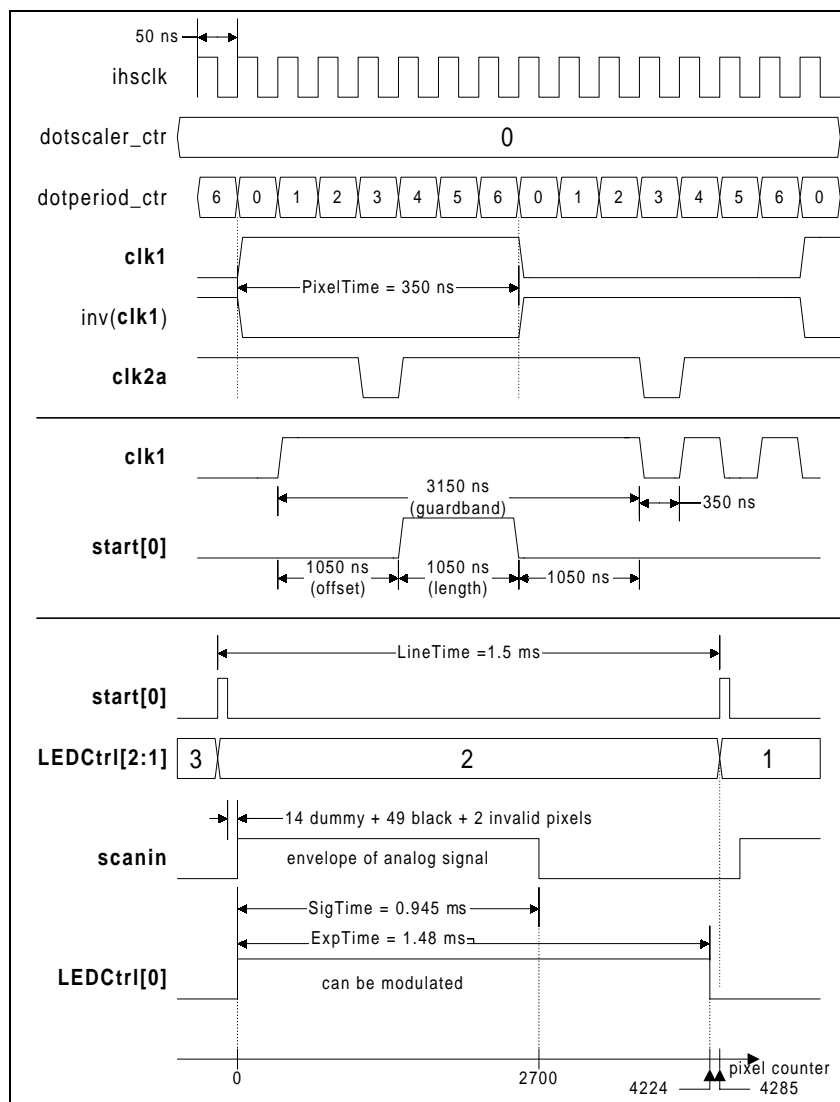
Figure 7-18. Untitled Timing Diagram

Table 7-5. Register Setup for NEC – μPD3724

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0100000b
	scctrl[6:2] select	00100b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	6
StartConfig	GuardbandEnb[6:4]	7
	Guardband[4:0]	8
	Offset[3:0]	2
	Length[3:0]	2
StartEdges	StartPos[3:0]	0
	StartNeg[3:0]	6
ScanStartDelay	Delay[12:0]	65 + 9 = 74
Clk2aEdges	Clk2aPos[3:0]	4
	Clk2aNeg[3:0]	2
Clk2bEdges	Clk2bPos[3:0]	n/a
	Clk2bNeg[3:0]	n/a
Clk2cEdges	Clk2cPos[3:0]	n/a
	Clk2cNeg[3:0]	n/a
ADCSampleCfg	SamplePos[3:0]	0
	SampleNeg[3:0]	1
ADCConfig	AdclkPos[3:0]	
	AdclkNeg[3:0]	
	DataPlace[3:0]	
	DataLatency[3:0]	
ClampCtrl	ClampMode	1
	ClampEnb	1
	ClampLength[6:0]	49
ClampDelay	ClampPosition	1
	ClampDelay[13:0]	13 + 9 = 22
ClampEdges	ClampPos[3:0]	0
	ClampNeg[3:0]	1
LEDConfig	LEDCtrl sync_sel[2:0]	6
	LEDCtrl place[2:0]	1
LED0Edges	LED0Pos[7:0]	0
	LED0Neg[7:0]	66
LED1Edges	LED1Pos[7:0]	n/a
	LED1Neg[7:0]	n/a
LED2Edges	LED2Pos[7:0]	n/a
	LED2Neg[7:0]	n/a
ScanLength	ScanLength[12:0]	2700
ScanCyles	ScanCyle[4:0]	*1.5 ms*

Manufacturer	NEC
Part #	•PD3794
Type	CCD
Resolution	300 dpi
Pixels	2700 x 3
Scan time (color)	
Scan time (B/W)	
Data Rate	4 MHz (max)
Main clock 'H' duty	

Interface:

Scanner	Description	MFC2000
	LED control	LEDCtrl[0]
•TG[3:1]	Start pulse	start[0]
SEL1	Color selector1	LEDCtrl[1]
SEL2	Color selector2	LEDCtrl[2]
•RB	Reset clock	clk2a
•1	Shift register clock 1	clk2b
•2	Shift register clock 2	clk2c

Timing:

50 ns

ihsclk

dotscaler_ctr 0

dotperiod_ctr 4 0 1 2 3 4 0 1 2 3 4 0 1 2 3 4

clk2a

clk2b

clk2c

PixelTime = 250 ns

clk1

5000 ns (guardband) — 250 ns

start[0]

1000 ns (offset) — 3000 ns (length) — 1000 ns

LineTime =1 ms

start[0]

LEDCtrl[2:1] 3 2 1

14 dummy + 48 black + 2 invalid pixels

scanin — envelope of analog signal

SigTime = 0.675 ms

ExpTime = ~1 ms

LEDCtrl[0] — can be modulated

pixel counter

0   2700   3904 — 3936
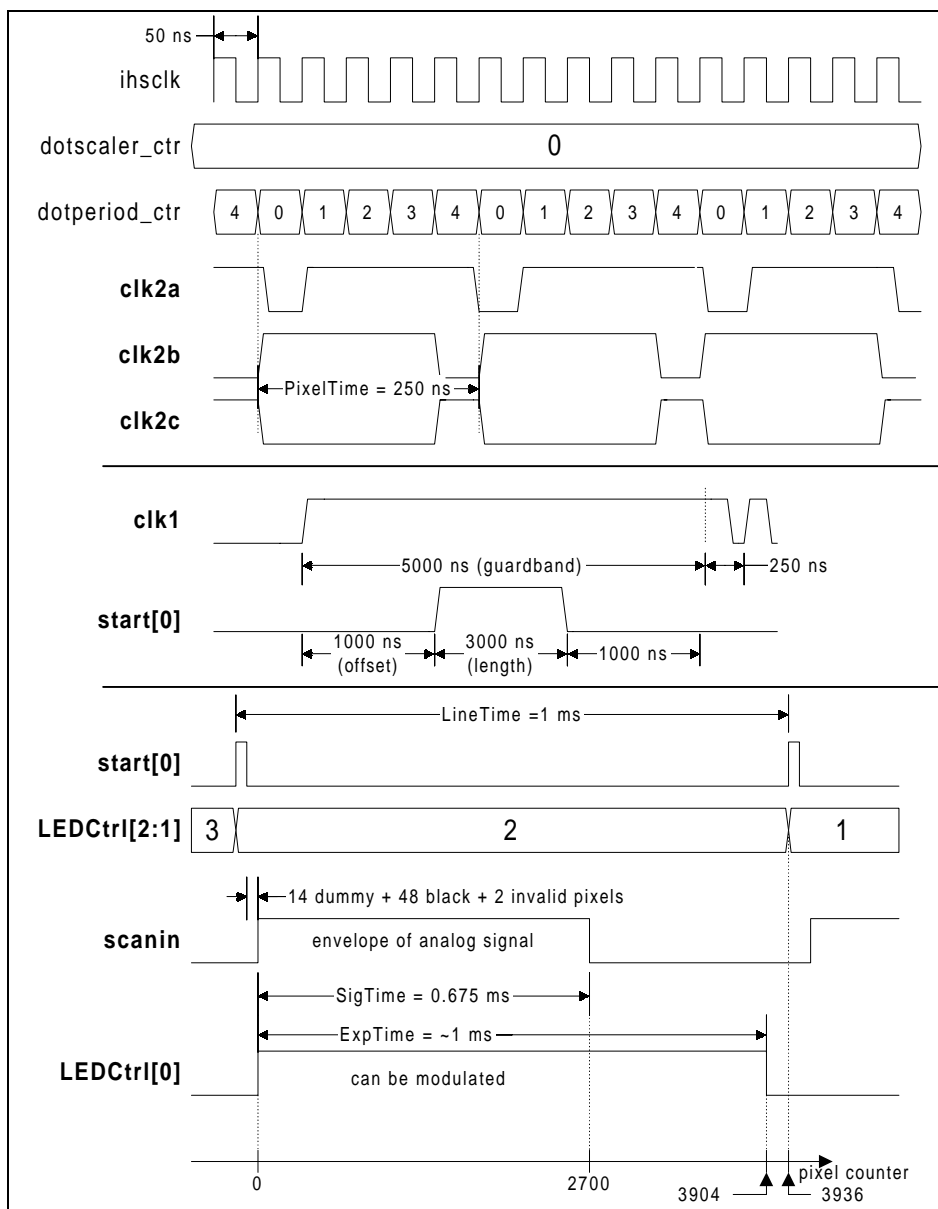
Figure 7-19. Untitled Timing Diagram

Table 7-6. Register setup for NEC – μPD3794

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0000000b
	scctrl[6:2] select	11000b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	4
StartConfig	GuardbandEnb[6:4]	6
	Guardband[4:0]	19
	Offset[3:0]	3
	Length[3:0]	11
StartEdges	StartPos[3:0]	0
	StartNeg[3:0]	4
ScanStartDelay	Delay[12:0]	83
Clk2aEdges	Clk2aPos[3:0]	4
	Clk2aNeg[3:0]	1
Clk2bEdges	Clk2bPos[3:0]	3
	Clk2bNeg[3:0]	0
Clk2cEdges	Clk2cPos[3:0]	4
	Clk2cNeg[3:0]	4
ADCSampleCfg	SamplePos[3:0]	4
	SampleNeg[3:0]	4
ADCConfig	AdclkPos[3:0]	
	AdclkNeg[3:0]	
	DataPlace[3:0]	
	DataLatency[3:0]	
ClampCtrl	ClampMode	1
	ClampEnb	1
	ClampLength[6:0]	47
ClampDelay	ClampPosition	1
	ClampDelay[13:0]	33
ClampEdges	ClampPos[3:0]	2
	ClampNeg[3:0]	2
LEDConfig	LEDCtrl sync_sel[2:0]	6
	LEDCtrl place[2:0]	1
LED0Edges	LED0Pos[7:0]	0
	LED0Neg[7:0]	60
LED1Edges	LED1Pos[7:0]	n/a
	LED1Neg[7:0]	n/a
LED2Edges	LED2Pos[7:0]	n/a
	LED2Neg[7:0]	n/a
ScanLength	ScanLength[12:0]	2700
ScanCyles	ScanCycle[4:0]	0

Manufacturer	SONY
Part #	ILX516K
Type	CCD
Resolution	400 dpi
Pixels	3648 x 3
Scan time (color)	
Scan time (B/W)	
Data Rate	5 MHz (max)
Main clock 'H' duty	50 %

Interface:

Scanner	Description	MFC2000
	LED control	LEDCtrl[0]
•ROG (3)	Start pulse	start[0]
SEL1	Color selector1	LEDCtrl[1]
SEL2	Color selector2	LEDCtrl[2]
•RS	Reset clock	clk2a
•1, •LH	Shift register clock 1	clk2b
•2	Shift register clock 2	clk2c

Timing:

50 ns
ihsclk
dotscaler_ctr 0
dotperiod_ctr 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2
clk2a
min = 45 ns   min = 45 ns   valid output
clk2b
clk2c
PixelTime = 200 ns (min)   typ = 10 ns   typ = 10 ns
clk2b
start[0]
800 ns (min)   200 ns
200 ns (min = 50 ns)   800 ns (min)
LineTime =1.5 ms
start[0]
LEDCtrl[2:1] 0   1   2
63 dummy pixels
scanin   envelope of analog signal
SigTime = 0.7296 ms
ExpTime = 1.4976 ms
LEDCtrl[0]   can be modulated
0   3648   7488   7500   pixel counter
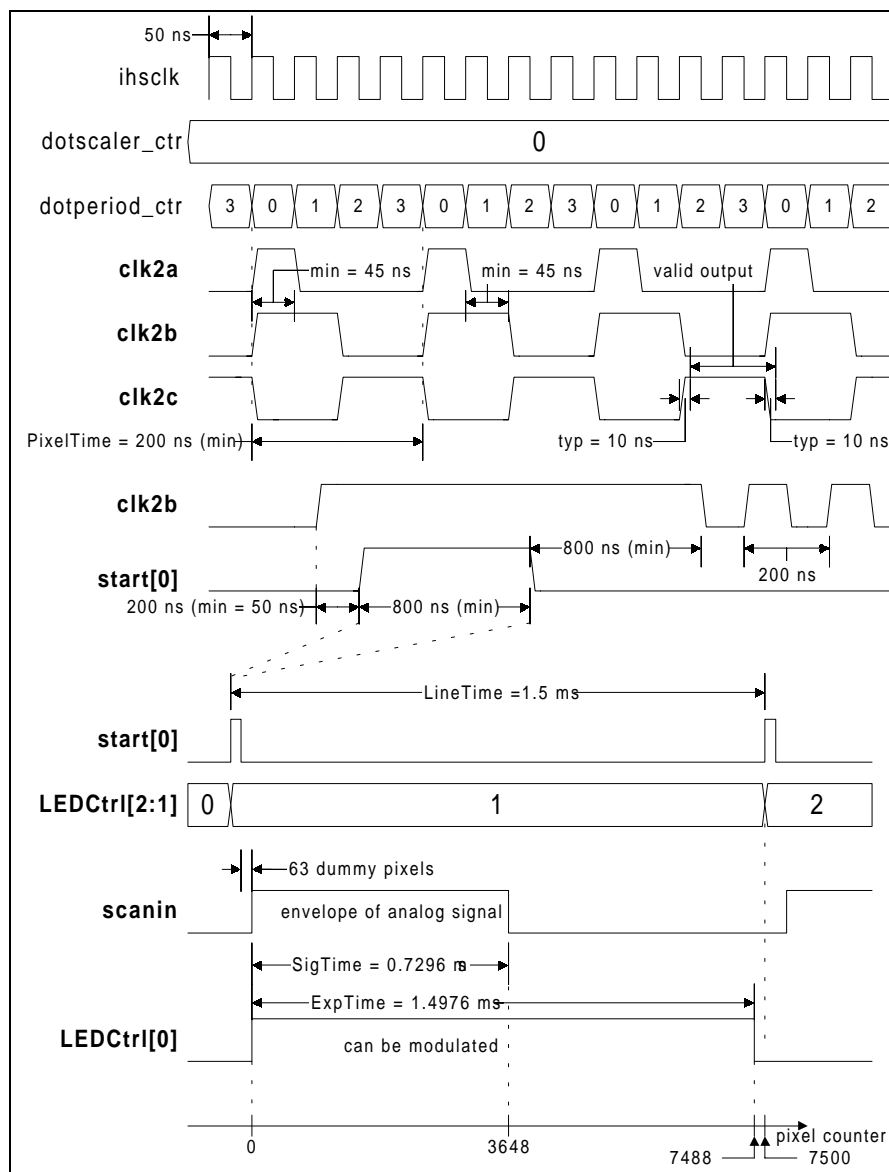
Figure 7-20. Untitled Timing Diagram

Table 7-7. Register Setup for SONY – ILX516K

Register Name	Bits Name	Value
ScanConfig	scctrl[6:0] invert	0110000b
	scctrl[6:2] select	11100b
ScanDotControl	Scan Dot Scaler[3:0]	0
	Scan Dot Period[3:0]	3
StartConfig	Guardband[6:4]	7
	Guardband[4:0]	9
	Offset[3:0]	0
	Length[3:0]	3
StartEdges	StartPos[3:0]	1
	StartNeg[3:0]	3
ScanStartDelay	Delay[12:0]	63 + 10 = 73
Clk2aEdges	Clk2aPos[3:0]	1
	Clk2aNeg[3:0]	3
Clk2bEdges	Clk2bPos[3:0]	2
	Clk2bNeg[3:0]	3
Clk2cEdges	Clk2cPos[3:0]	2
	Clk2cNeg[3:0]	3
ADCSampleCfg	SamplePos[3:0]	3
	SampleNeg[3:0]	3
ADCConfig	AdclkPos[3:0]	
	AdclkNeg[3:0]	
	DataPlace[3:0]	
	DataLatency[3:0]	
ClampCtrl	ClampMode	1
	ClampEnb	1
	ClampLength[6:0]	49
ClampDelay	ClampPosition	1
	ClampDelay[13:0]	10 + 14 = 24
ClampEdges	ClampPos[3:0]	1
	ClampNeg[3:0]	1
LEDConfig	LEDCtrl sync_sel[2:0]	6
	LEDCtrl place[2:0]	1
LED0Edges	LED0Pos[7:0]	0
	LED0Neg[7:0]	117
LED1Edges	LED1Pos[7:0]	n/a
	LED1Neg[7:0]	n/a
LED2Edges	LED2Pos[7:0]	n/a
	LED2Neg[7:0]	n/a
ScanLength	ScanLength[12:0]	3648
ScanCyles	ScanCyle[4:0]	*1.5 ms*

External circuit required for SONY−ILX516K interface:
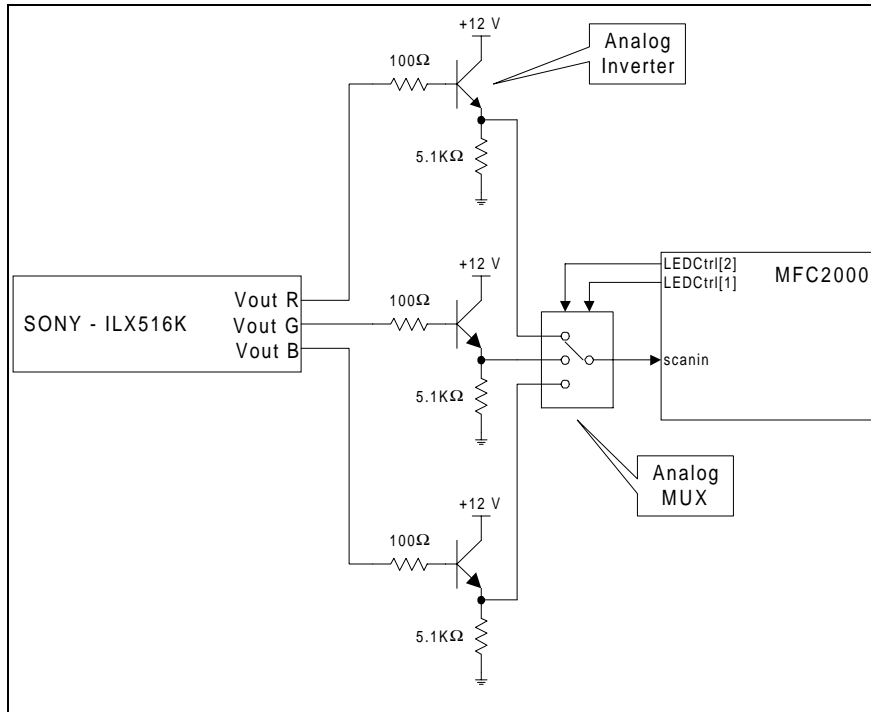
Figure 7-21. External circuit required for SONY–ILX516K interface

LED timing for SONY−ILX516K
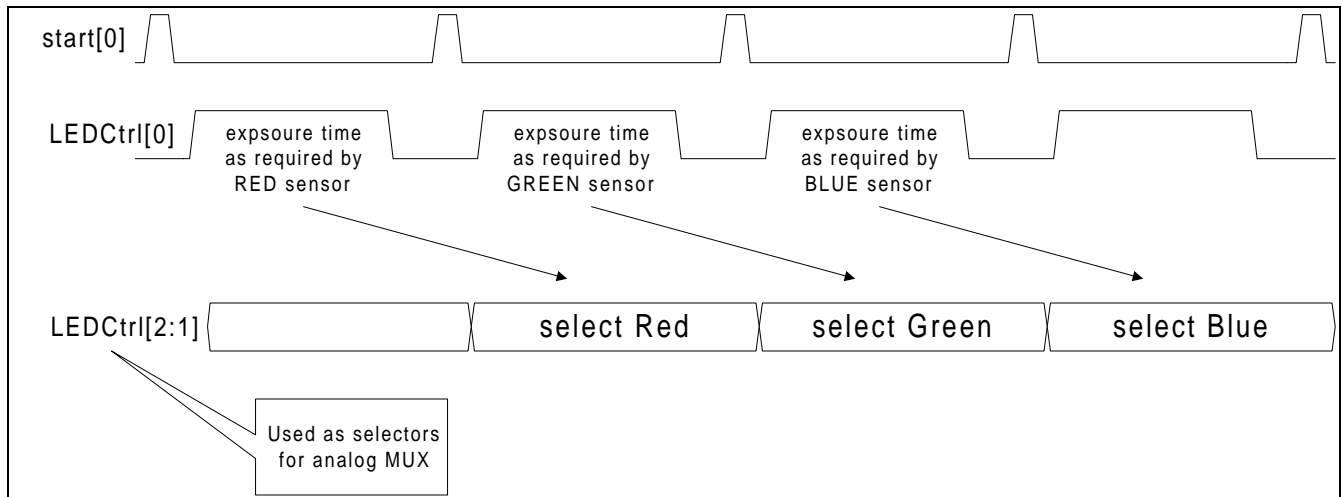
Figure 7-22. LED timing for SONY–ILX516K

7.2   Serial Programming Interface

The serial programming interface is a two-wire bidirectional serial bus, which provides a simple and efficient way for data exchange between devices.

Main Features:

- Only two bus lines are required; a serial interface data (SID) and a serial interface clock (SIC)
- Master–transmitter & master–receiver
- Schmitt trigger on SID input
- Synchronization capability (allowing slave to add wait state by pulling the clock low)
- Optional interrupt or polling operation
- Optional firmware operation for manual control of the SPI bus
- Soft reset

7.2.1   Function Description

Physical Connection
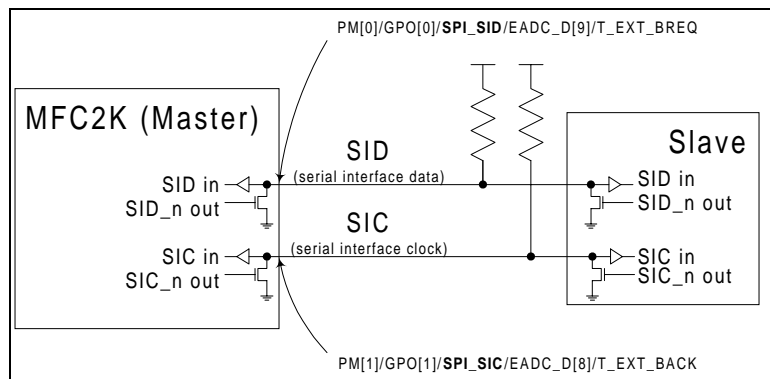
Figure 7-23. Serial Programming Interface, Physical Connection

Notes:

1. *Master is the device that initiates a transaction, generates clock signal, and terminates the transaction.*

2. *Data and clock can only be driven low, and cannot be driven high. The pull-up resistors are responsible to create a logic 1. This configuration forms wire – AND connection.*

3. *Slave can only drive the clock signal to add wait state.*

4. *There is no minimum requirement for clock frequency.*

Bus Protocol
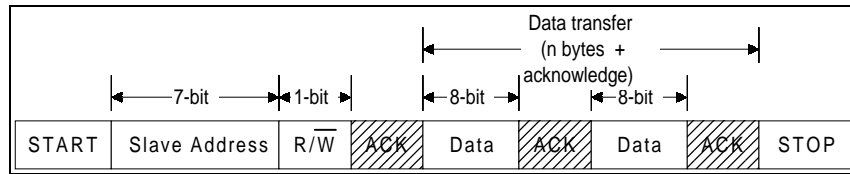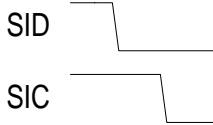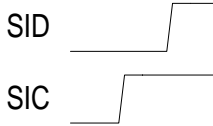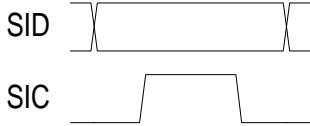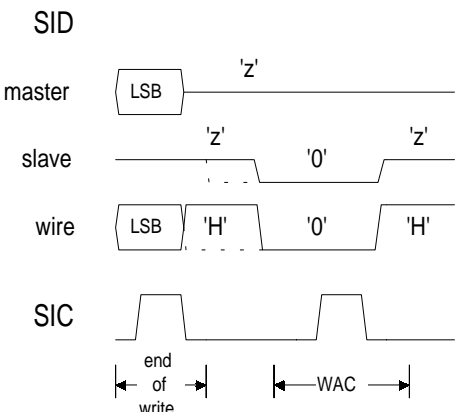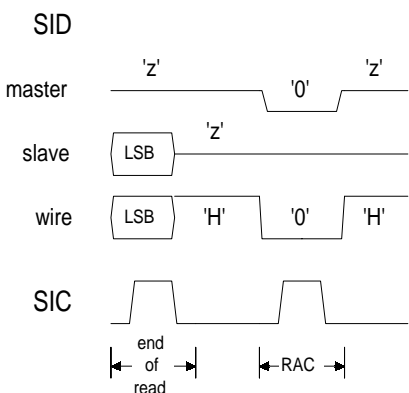
Figure 7-24. Bus Protocol
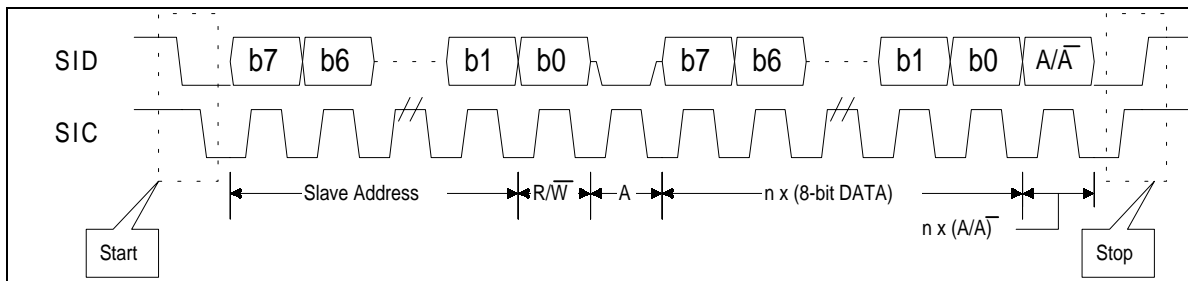
[spi_timing.vsd]

Figure 7-25. Serial Programming Interface, Timing Diagram

START condition		Data is pulled low while clock is still high, followed by pulling the clock low. Clock and data will be held low.
	SID SIC	
STOP condition	SID SIC	Clock is released to high, followed by releasing data to high.
Bit transfer	SID SIC	Address/data bit must be valid while SIC is high.
Getting ACK	SID master LSB 'z' slave 'z' '0' 'z' wire LSB 'H' '0' 'H' SIC end of write WAC	This is the condition after master transmitted data to slave, which will be referred as WAC (write acknowlege). Slave is giving acknowledge by driving the data line low. Master is getting the acknowledge by generating a clock pulse to capture the acknowledge.
Giving ACK	SID master 'z' '0' 'z' slave LSB 'z' wire LSB 'H' '0' 'H' SIC end of read RAC	This is the condition after master received data from slave, which will be referred as RAC (read acknowledge). Master is giving acknowledge to the slave by driving data line low and by generating a clock pulse. Slave is receiving the acknowledge from master.

Clock Stretching

Slave can add wait state as needed by stretching the low period of the clock. This will slow down the bus as shown below.
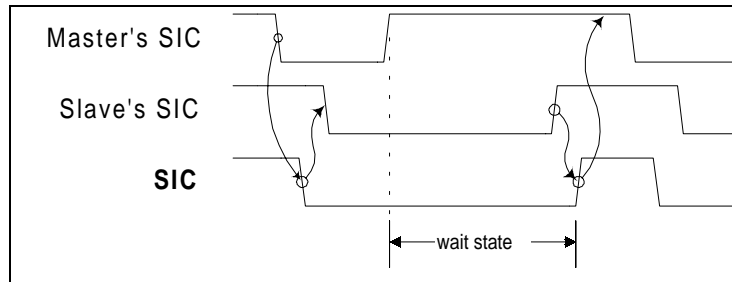
Figure 7-26. Stretching the Low Period of the Clock

7.2.2   Register Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
SPI Control *(SPICtrl)* **$01FF854B**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	WRI (read only)	Rst. Value xxxxxxx0b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
SPI Control *(SPICtrl)* **$01FF854A**	RST	REA	STA	STO	RAC	WAC	SID	SIC	Rst. Value 00h Read Value 00h

Bit 8   WRI – Writing   This is a read only bit which indicates the writing operation. Reading a 1 means the SPI is in the process of sending/writing data to slave.

Bit 7   RST – Reset   Writing a 1 to this bit will cause the SPI logic controller to be resetted. If SPI is in operation while RST bit is set, then the operation will be terminated. Reading a 1 from this bit indicates that RST operation is active. After the reset, both SID and SIC lines will be released.

Bit 6   REA – Read   Writing a 1 to this bit will generate 8 clocks to read data from slave, and SPI will capture serial data into the upper byte of 'SPI_data' register. Reading a 1 from this bit indicates that REA operation is active.

Bit 5   STA – Start   Writing a 1 to this bit will generate START condition. Reading a 1 from this bit indicates STA operation is active.

Bit 4   STO – Stop   Writing a 1 to this bit will generate STOP condition. Reading a 1 from this bit indicates STO operation is active.

Bit 3   RAC – Read Acknowledge   Writing a 1 to this bit will generate ACKNOWLEDGE condition after read operation in which the master will:

1. generate clock pulse
2. drive SID low

Reading a 1 from this bit indicates RAC operation is active.

Bit 2   WAC – Write Acknowledge   Writing a 1 to this bit will generate ACKNOWLEDGE condition after
                                            write operation in which the master will:

                                            1. generate clock pulse
                                            2. NOT drive SID but will capture acknowledge bit from slave and
                                            store it into bit 8 of 'SPI_Stat' register

                                            Reading a 1 from this bit indicates WAC operation is active.

Bit 1   SID – Serial Input Data   This bit is used to control the SID line directly, and it is only valid if the
                                            "Manual Operation" bit in SPI_Config is equal to 1. When SID = 1,
                                            then SPI is not driving the SID line. When SID = 0, then SPI will drive
                                            SID line low.

Bit 0   SIC – Serial Input Clock   This bit is used to control the SIC line directly, and it is only valid if the
                                            "Manual Operation" bit in SPI_Config is equal to 1. When SIC = 1,
                                            then SPI is not driving the SIC line. When SIC = 0, then SPI will drive
                                            SIC line low.

> *Note*:   *In normal case, only one of bit [7:2] should be set. However, if for some reasons, more than one bits are set, then the hardware will respond according to the following priority. Other bit(s) that are set will be ignored.*

Priority	Operation
1	RST
2	STO
3	STA
4	REA
5	RAC
6	WAC
7	WRI

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
SPI Status *(SPIStat)* $01FF854D	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Received ACK (read only)	Rst. Value xxxxxxx0b Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
SPI Status *(SPIStat)* $01FF854C	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	SID line (read only)	SIC line (read only)	Rst. Value xxxxxx00b Read Value 00h

Bit 8   Received ACK   This is a read only bit which is only valid after writing a 1 to "WAC" bit and after verifying that WAC operation is completed, i.e., WAC bit is 0. After master sends data to slave, master must check if slave has received the data properly. This checking is done by requesting acknowledge from slave. In this situation, master is sending a clock pulse, and capture the acknowledge on SID line while SIC is high. This is the acknowledge stored in bit 8.

Bit 1   SID – Serial Input Data line   This is a read only bit which comes directly from the SID pin.

Bit 0   SIC – Serial Input Clock line   This is a read only bit which comes directly from the SIC pin.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
SPI Configuration *(SPIConfig)* $01FF88C1	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
SPI Configuration *(SPIConfig)* $01FF88C0	Manual Operation	(Not Used)	(Not Used)	(Not Used)	SPI Interrupt Enable	Clock Mode	Clk Divider[1]	Clk Divider[0]	Rst. Value 0xxx0000b Read Value 00h

Bit 7   Manual Operation   By setting this bit, firmware can control the SID and SIC directly by writing to bit 1 and bit 0 of SPI_Ctrl register.

Bit 3   SPI Interrupt Enable   By setting this bit, the interrupt from SPI is enabled. At the end of REA, STA, STO, RAC, WAC, or WRI operation, the interrupt will be active in order to indicate the completion of operation. Once the firmware has serviced this interrupt, it can be cleared by writing a 1 to bit 1 of 'VSCIRQStatus' register.

Bit 2   Clock Mode

Bit 1-0   Clock Divider[1:0]

Clock Mode = 0			Clock Mode = 1	
Div[1:0]	SIC Freq		Div[1:0]	SIC Freq
0	1.2 KHz		0	104 KHz
1	9.8 KHz		1	156 KHz
2	39 KHz		2	208 KHz
3	78 KHz		3	312 KHz

Note: The freqencies specified above are based on iclk = 10MHz.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
SPI Data *(SPIData)* **$01FF88C3**	Read Data[7] (Read only)	Read Data[6] (Read only)	Read Data[5] (Read only)	Read Data[4] (Read only)	Read Data[3] (Read only)	Read Data[2] (Read only)	Read Data[1] (Read only)	Read Data[0] (Read only)	Rst. Value 00h Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
SPI Data *(SPIData)* **$01FF88C2**	Write Data[7]	Write Data[6]	Write Data[5]	Write Data[4]	Write Data[3]	Write Data[2]	Write Data[1]	Write Data[0]	Rst. Value 00h Read Value 00h

Bit 15-8   Read Data[15:8]   The data received is stored in this location.

Bit 7-0   Write Data[7:0]   The data transmitted comes from this location.

Writing to SPIData register will initiate the transmission of bit[7:0]. It should be noted that if the writing occurs while ACT = 1, then the current transmission will be corrupted.

7.2.3　Firmware Operation

<u>Initialization</u>: select the desired frequency for SIC by writing the appropriate value to "SPI_Config" register

<u>Transmission:</u>

```
                              ┌─────────────┐
                              │    START    │
                              └─────────────┘
                                     │ no
                     ┌───────────────────────────────┐
                     │  write 0x0020 to 'SPI_Ctrl' reg │ - - Generate START condition
                     └───────────────────────────────┘
                                     │
                     ┌───────────────────────────────┐
                     │      receive & clear spi_irq     │
                     └───────────────────────────────┘
                                     │
                     ┌───────────────────────────────┐
                     │     write to 'SPI_Data' reg      │ - - Send the 7-bit slave address + "W" bit
                     └───────────────────────────────┘
                                     │
                     ┌───────────────────────────────┐
                     │      receive & clear spi_irq     │
                     └───────────────────────────────┘
                                     │
                     ┌───────────────────────────────┐
                     │  write 0x0004 to 'SPI_Ctrl' reg │ - - Generate WAC condition
                     └───────────────────────────────┘
                                     │
                     ┌───────────────────────────────┐
                     │      receive & clear spi_irq     │
                     └───────────────────────────────┘
                                     │
   ┌─────────────────────┐ yes  ◇───────────────────◇
   │ write to 'SPI_Data' reg │◀────── Send more data? ◇
   └─────────────────────┘      ◇───────────────────◇
                                     │ no
                     ┌───────────────────────────────┐
                     │  write 0x0010 to 'SPI_Ctrl' reg │ - - Generate STOP condition
                     └───────────────────────────────┘
                                     │
                              ┌─────────────┐
                              │     END     │
                              └─────────────┘
```
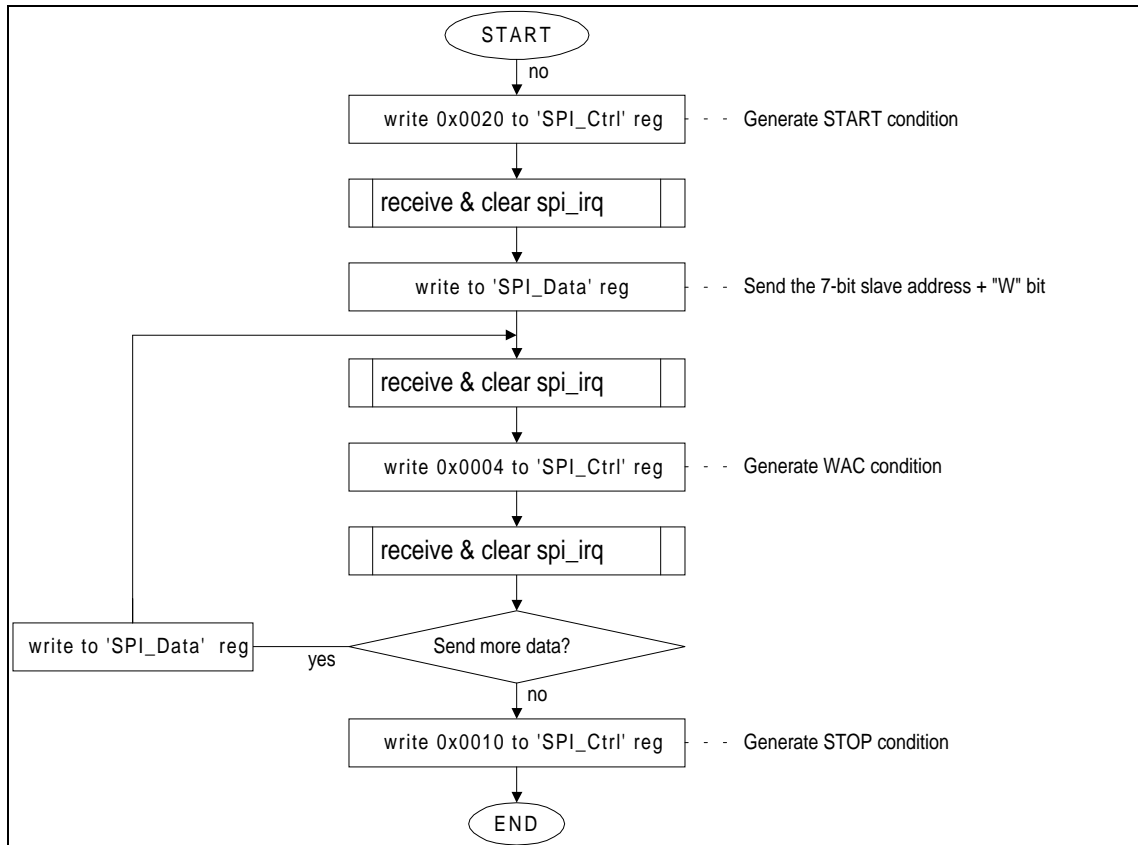
[spi_fw_tx.vsd]

Figure 7-27. Firmware Operation—Transmission

Reception:
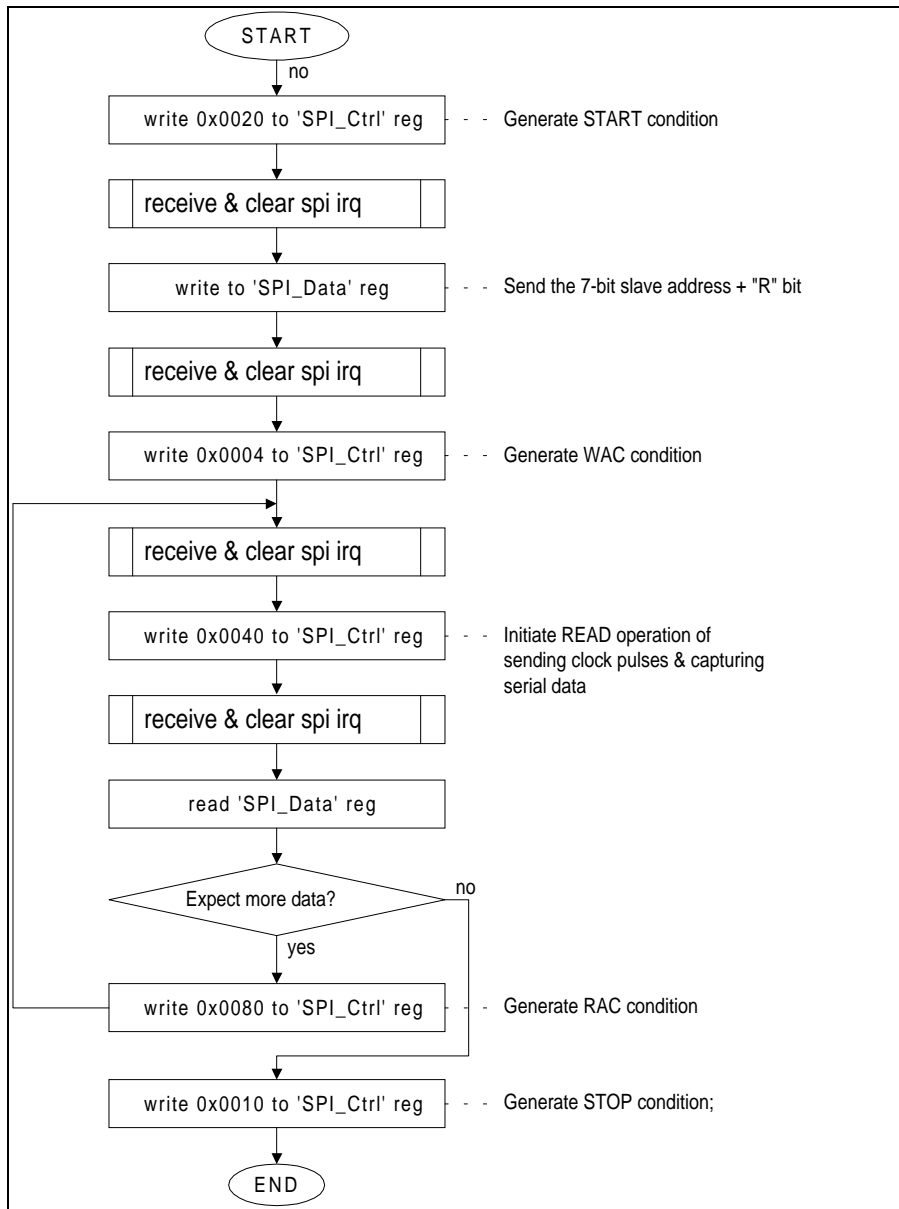
Figure 7-28. Firmware Operation—Reception

[spi_fw_rx.vsd]

7.3 Video Controller

Main features:

- Capture a frame from an external video decoder device
- Support ITU–R BT.656 or VESA Video Interface Port (VIP) format of YCrCb 4:2:2
- Store captured frame into memory in a non-interlace format
- Software interrupt at the end of a frame capture

ITU–R BT.656

- Data format: C_B, Y, C_R, Y, etc

Y	Y
C_B	
C_R	
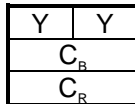
- Control structure: FF, 00, 00, XY

    XY contains the video timing reference code

    XY = {1, F, V, H, P3, P2, P1, P0}, where

    F = Field

    V = Vertical

    H = Horizontal (0 = SAV, 1 = EAV)

    P3, P2, P1, P0 = error protection bits, with P3 = V xor H, P2 = F xor H, P1= F xor V, P0 = F xor V xor H

XY	F	V	H	*type*
80	0	0	0	SAV
9D	0	0	1	EAV
AB	0	1	0	SAV
B6	0	1	1	EAV
C7	1	0	0	SAV
DA	1	0	1	EAV
EC	1	1	0	SAV
F1	1	1	1	EAV

- Control insertion:

    1. at the beginning of a line: SAV (start of active video)

    2. at the end of a line: EAV (end of active video)

- Number of pixels per line is 720.

- The following control codes are inserted for every valid line:

Field	Start Line	End Line
0	FF, 00, 00, 80	FF, 00, 00, 9D
1	FF, 00, 00, C7	FF, 00, 00, DA

VESA VIP

- Dummy byte/ invalid pixel code: 00 (between SAV and EAV)

- Number of pixels per line is not restricted to 720

- The most significant bit of XY byte is called task bit (T-bit), which can be 1 or 0.

Convention

- F = 0 (Field = 1) is an odd field

- F = 1 (Field = 2) is an even field

- Odd field is the top field

- Even field is the bottom field

7.3.1   Function Description

Pins connection to external video capture device:
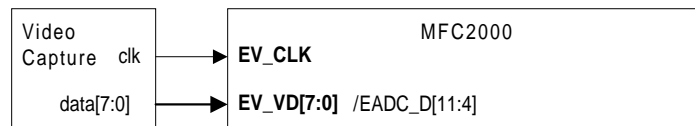
Figure 7-29. Connection to External Video Capture Device

7.3.2   Register Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Video Capture Control *(VidCaptureCtrl)* $01FF8549	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Video Error (read only)	Rst. Value xxxxxxx0b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Video Capture Control *(VidCaptureCtrl)* $01FF8548	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Start Capture	Rst. Value xxxxxxx0b Read Value 00h

Bit 8   Video Error   This status bit indicates that error has been detected in capturing the video signal. The error is detected based on the line length comparison between the specified number of bytes per line (i.e., the value in *VideoLineCfg* register) to the actual number of bytes captured. If any line within a frame does not match *VideoLineCfg* register, this bit will be set.

At the beginning of the next video capture, this error bit will be cleared.

Bit 0   Start Capture   By setting this bit, the hardware will start the process of capturing a frame of video data. Once a complete frame has been captured, the hardware will clear this bit. The status of capturing process can be known by reading this bit. Reading a 1 indicates that the hardware is busy.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Video Line Config *(VidLineCfg)* $01FF88C7	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	LineSize[10]	LineSize[9]	LineSize[8]	Rst. Value xxxxx000b Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Video Line Config *(VidLineCfg)* $01FF88C6	LineSize[7]	LineSize[6]	LineSize[5]	LineSize[4]	LineSize[3]	LineSize[2]	LineSize[1]	0	Rst. Value 00h Read Value 00h

Bit 10-1   LineSize[10:1]   This register defines the **maximum** number of bytes to be captured per line. The value in this register must be written prior to initiating the video capture. Bit 0 is a constant zero in order to ensure that line size is an even number, i.e., on the word boundary.

If the number of actual data is more than this value, then the actual data will be truncated. It should be noted that at the end of every line, one or two byte of zeros will be added in order to indicate the end of line, and at the same time the additional zeros will guarantee that the line length is on the word boundary. Whenever the actual data does not match the LineSize, the "Video Error" bit in "Video Capture Config" register will be set to 1.

> *Note: The jump or skip parameter in DMA setting must be* at least *1 halfword bigger than the "LineSizeCfg" in order to take into account the additional zeros at the end of line.*

Example of line *too short*

    VidLineCfg = 10 bytes
    Number of actual data from video decoder = 9 bytes
    Number of actual data stored into memory = 9 bytes
    Number of zeros inserted into memory = 1 byte
    Final line size = 10 bytes

Example of line *too long*

    VidLineCfg = 10 bytes
    Number of actual data from video decoder = 11 bytes
    Number of actual data stored into memory = 10 bytes
    Number of zeros inserted into memory = 2 byte
    Final line size = 12 bytes

Example of line *is just right*

    VidLineCfg = 10 bytes
    Number of actual data from video decoder = 10 bytes
    Number of actual data stored into memory = 10 bytes
    Number of zeros inserted into memory = 2 byte
    Final line size = 12 bytes

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Video Line Length Status *(VidLLStat)* **$01FF88C9**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	VidLL[10] (read only)	VidLL[9] (read only)	VidLL[8] (read only)	Rst. Value xxxxx000b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Video Line Length Status *(VidLLStat)* **$01FF88C8**	VidLL[7] (read only)	VidLL[6] (read only)	VidLL[5] (read only)	VidLL[4] (read only)	VidLL[3] (read only)	VidLL[2] (read only)	VidLL[1] (read only)	VidLL[0] (read only)	Rst. Value 00h Read Value 00h

Bit 10-0   VidLL[10:0]   This register indicates the number of bytes per line and not the number of pixels per line. The value in this register is updated continuously at the end of every line.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Video Odd Field Length Status *(VidOddFLStat)* **$01FF88CB**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	VidOddFL [8] (read only)	Rst. Value xxxxxxx0b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Video Odd Field Length Status *(VidOddFLStat)* **$01FF88CA**	VidOddFL [7] (read only)	VidOddFL [6] (read only)	VidOddFL [5] (read only)	VidOddFL [4] (read only)	VidOddFL [3] (read only)	VidOddFL [2] (read only)	VidOddFL [1] (read only)	VidOddFL [0] (read only)	Rst. Value 00h Read Value 00h

Bit 8-0   VidOddFL [8:0]   This register indicates the number of lines in odd field. The value in this register is updated continuously at the end of every odd field.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Video Even Field Length Status *(VidEvenFLStat)* **$01FF88CD**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	VidEvenFL [8] (read only)	Rst. Value xxxxxxx0b Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Video Even Field Length Status *(VidEvenFLStat)* **$01FF88CC**	VidEvenFL [7] (read only)	VidEvenFL [6] (read only)	VidEvenFL [5] (read only)	VidEvenFL [4] (read only)	VidEvenFL [3] (read only)	VidEvenFL [2] (read only)	VidEvenFL [1] (read only)	VidEvenFL [0] (read only)	Rst. Value 00h Read Value 00h

Bit 8-0   VidEvenFL [8:0]   This register indicates the number of lines in even field. The value in this register is updated continuously at the end of every even field.

7.3.3   Timing

Data is sampled at the rising edge.

The control codes indicate the beginning and the end of a valid line, with invalid pixels inserted.
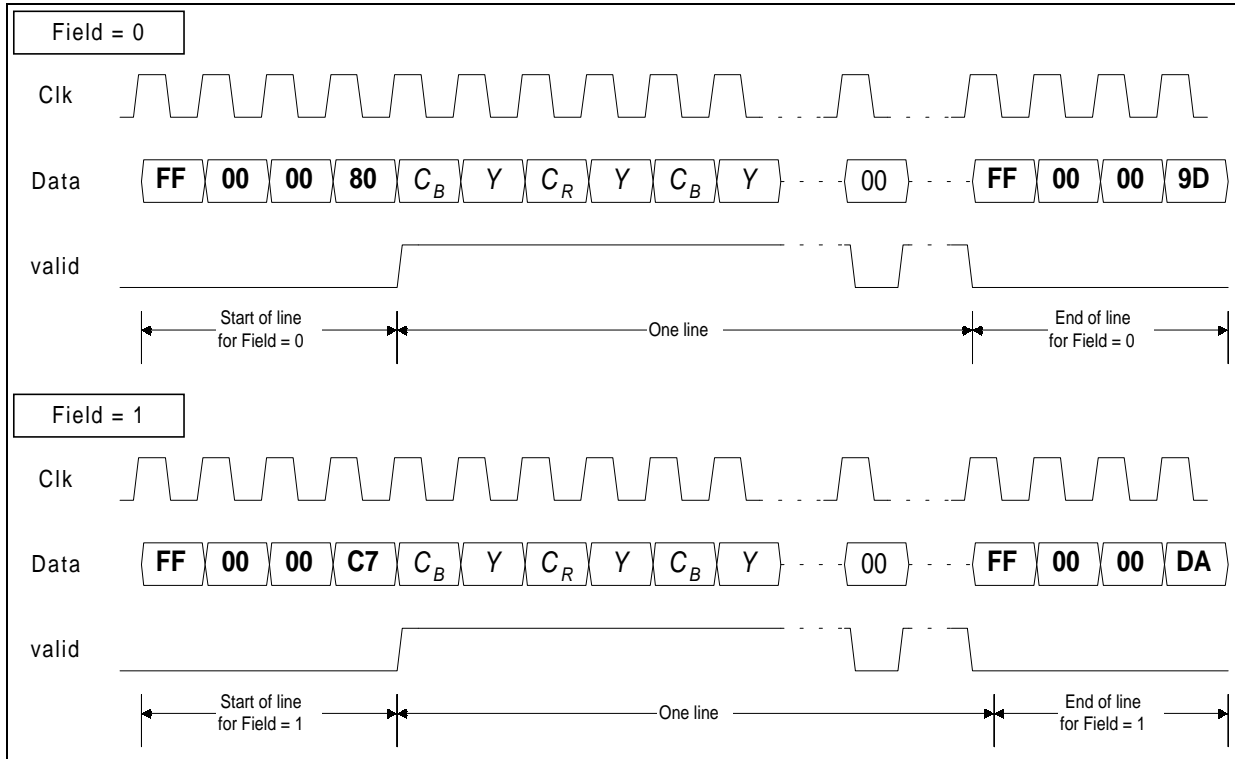
Figure 7-30. Untitled Timing Diagram

7.3.4   Firmware Operation

Firmware sequence of operations:

1. Setup the external video capture device via two-wire serial interface

2. Poll the status register in external video capture device via two-wire serial interface

3. Read *VidLLStat* and *VidOddFLStat* registers if video signal is detected

4. Setup DMA based on the values stored in *VidLLStat* and *VidOddFLStat* registers

5. Write to *VidLineSizeCfg* register based on the *VidLLStat*

6. Start the video capture by writing to *VidCaptureCtrl* register

DMA operation:
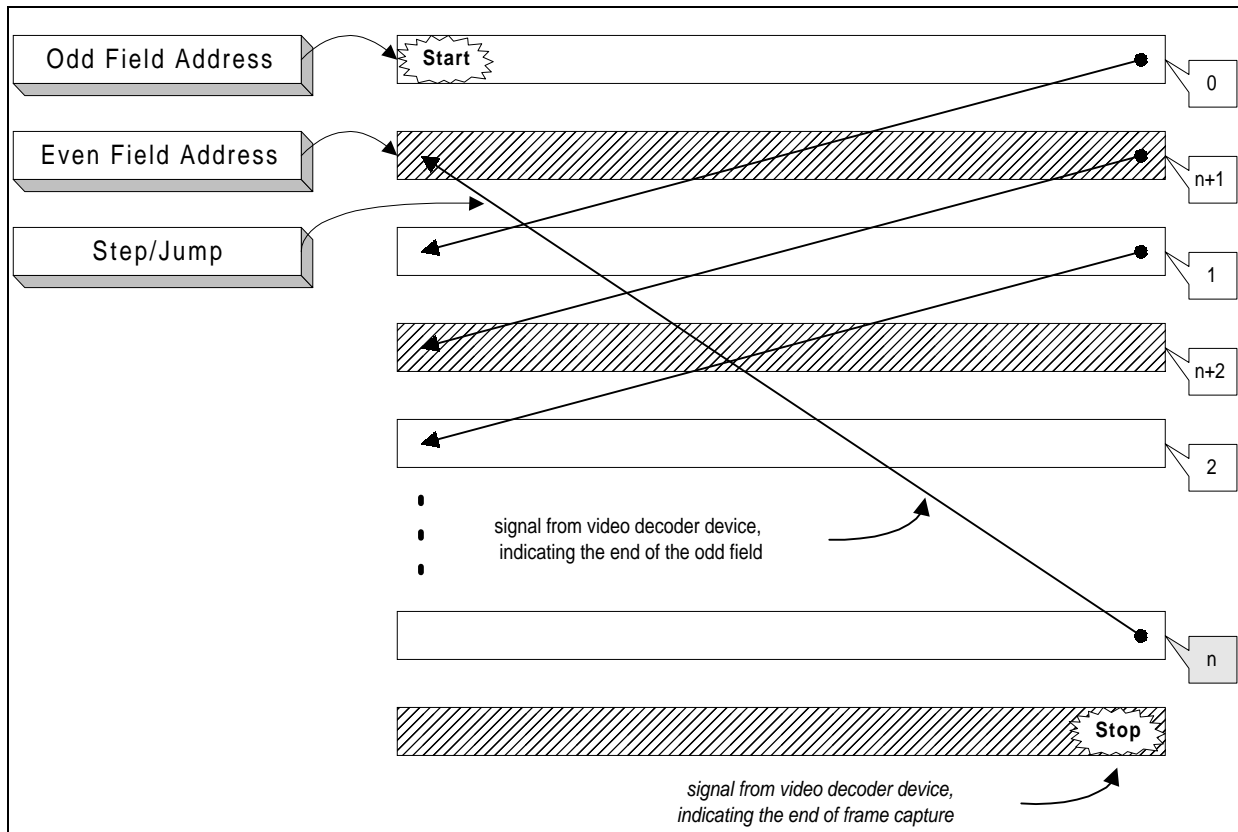
Figure 7-31. DMA Operation

DMA features:

*   clear internal counter after address/other registers are written
*   selection bit (scan/video)
    *   scan: 1 address will be used, and operation is simple
    *   video: 2 addresses will be used, and operation is complex as shown above

This page is intentionally blank.

8. ADC

8.1 PADC AND SCAN ANALOG FRONT END

8.1.1 General description
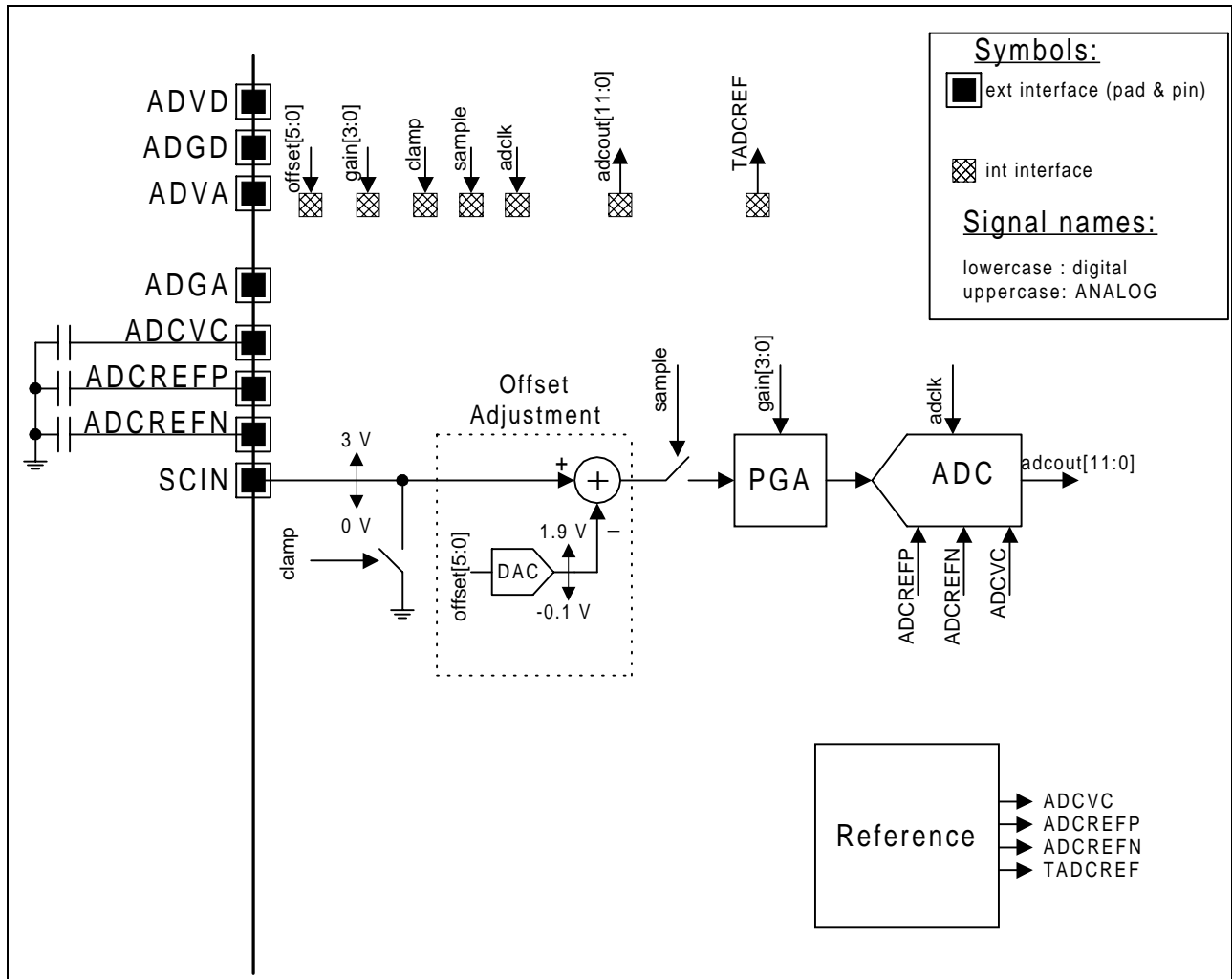
Figure 8-1. Untitled Figure

Table 8-1. Untitled Table

Name	Description
adclk	Clock for ADC circuit
	The maximum frequency of this clock is 5 MHz. This clock is guaranteed to be low while "sample" is high.
clamp	Clock for clamping circuit
	The maximum frequency of this clock is 5 MHz.
gain[3:0]	Selector for gain adjustment on PGA circuit
offset[5:0]	Selector for offset adjustment on DAC circuit
sample	Clock for sampling circuit
	The maximum frequency of this clock is 5 MHz. The minimum (high) pulse width is 50 ns.
adcout[11:0]	Digital data from ADC circuit
TADCREF	Analog signal as a reference voltage for TADC

Table 8-2. Untitled Table

Name	Description
ADCREFN	An internally generated reference voltage
ADCREFP	An internally generated reference voltage
ADGA	Ground return from analog circuits
ADGD	Ground return from digital circuits
ADVA	+3.3V power supply for analog circuits
ADVC	An internally generated reference voltage
ADVD	+3.3V power supply for digital circuits
SCIN	Analog input for scanner signal

8.1.2 Offset Adjustment

This offset adjustment contains DAC which takes digital input to configure the DAC and analog subtraction to subtract the incoming scanner signal with the analog signal from the DAC. This adjustment is used to remove/reduce the dark level offset on the scanner signal. The adjustment on DAC value is described below:

Table 8-3. Offset Adjustment on DAC

Parameter	Min	Typ	Max	Units
offset[5:0]				
000000		-0.1		V
11111		1.9		V

8.1.3   Programmable Gain Amplifier (PGA)

Table 8-4. Programmable Gain Amplifier (PGA)

Parameter	Min	Typ	Max	Units
Input Voltage Range (Gain = 1)	0		3	V
Gain[3:0] (in hex)				
0	-1.25	-1	-0.75	dB
1	-0.25	0	0.25	dB
2	0.75	1	1.25	dB
3	1.75	2	2.25	dB
4	2.75	3	3.25	dB
5	3.75	4	4.25	dB
6	4.75	5	5.25	dB
7	5.75	6	6.25	dB
8	6.75	7	7.25	dB
9	7.75	8	8.25	dB
A	8.75	9	9.25	dB
B	9.75	10	10.25	dB
C	10.75	11	11.25	dB
D	11.75	12	12.25	dB
E	12.75	13	13.25	dB
F	13.75	14	14.25	dB

Note: Scanner output will be sampled at the falling edge of the "sample" signal

8.1.4   Pipelined Analog-to-Digital Converter (PADC)

Table 8-5. Pipelined ADC (PADC)

Parameter	Min	Typ	Max	Units
RESOLUTION		12		Bits
SPEED			5	MSPS
ACCURACY				
Differential Nonlinearity (DNL)	-1	±0.5	+1	LSB
Integral Nonlinearity (INL)	-2	±1	+2	LSB
POWER SUPPLY REJECTION		±1		mV/V
OUTPUT FORMAT		✓		Binary unsigned
LATENCY			16	ADCLK

8.1.5   Timing Diagram
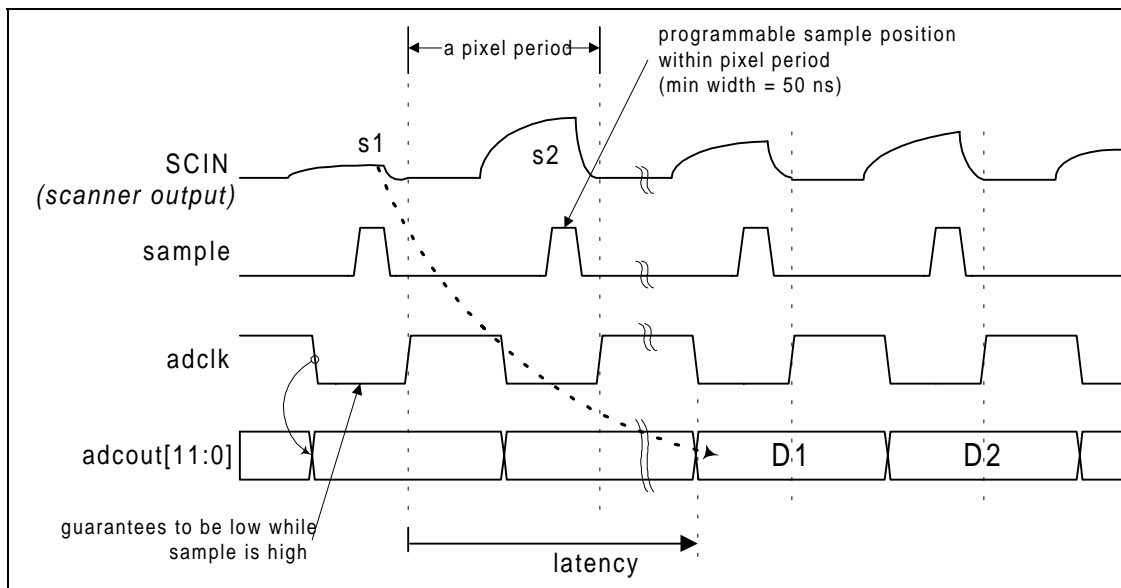
Table 8-6. PADC Timing Diagram

8.1.6   ADCVREFP, ADCVREFN, and ADCVC

The ADCVREFP and ADCVREFN voltages provide positive and negative references for ADCs while VC is used as common mode voltage in the design of ADC and PGA. These voltages are decoupled with external capacitors (0.1μF) to reduce thermal noise and also to provide high frequency ac current for sampling.

8.1.7   TADCREF

Parameter	Min	Typ	Max	Units
TADCREF	2.370	2.390	2.410	V

8.2   TADC

8.2.1   Function Description

Thermal ADC provides the analog to digital functionality for low speed analog signals. A simplified block diagram of TADC is shown below.
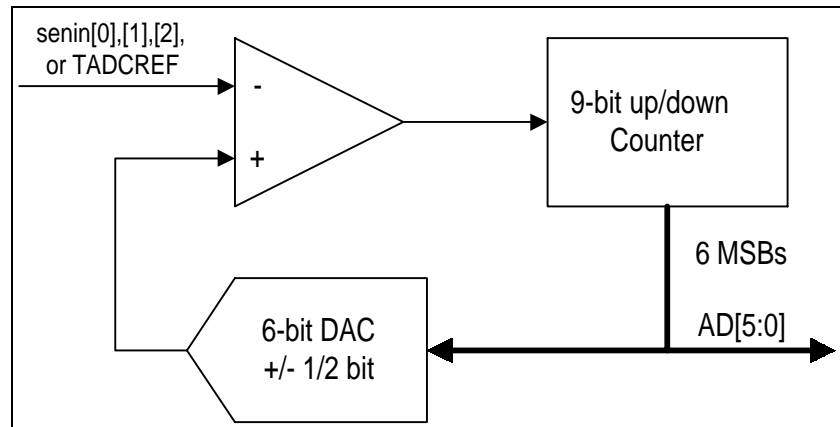
[tadc-simple.vsd]

Table 8-7. TADC Block Diagram

The TADC includes a 6-bit DAC, a comparator and a 9-bit self tracking up/down counter. The up/down counter is incremented or decremented based on the polarity of the comparator output. The most significant 6 bits of the counter are used to generate the DAC reference signal to the comparator. The comparator compares this reference voltage to the external analog input signals, SENIN[2:0], or TADCREF. After settling, the counter value (and thus the DAC digital value) represents the A-to-D conversion of the input signal. The least significant bits of the counter are used to average out the comparator fluctuations which occurs at the settling point. The counter is set to a mid-point value on the power-on reset so that it settles more quickly. The frequency of updating the counter is programmable.

The TADC can be operated in two modes. By default, the manual mode will be selected. In this mode, the firmware will select the analog input. Then, the firmware must wait for at least for 256 * tadcclk period to allow the TADC to settle prior to reading the data. In the automatic mode, the hardware will do the selection for each analog input in rotation. Each analog input will be selected periodically, and the data from each channel will be stored in different register at the same rate as channel selection.

8.2.2   Register Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
TADC Control *(TADCCtrl)* $01FF8001	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
TADC Control *(TADCCtrl)* $01FF8000	(Not Used)	Tadcclk Select[1]	Tadcclk Select[0]	Update Select[1]	Update Select[0]	Channel Select[1]	Channel Select[0]	Mode	Rst. Value x0000000b Read Value 00h

Bit 6-5   Tadcclk Select[1:0]   The frequency for tadc counter is programmable according to the following table.

Tadcclk Select[1:0]	Tadcclk Frequency	Tadcclk Frequency (assuming iclk = 10MHz)
0	freq(iclk)/64	156 KHz
1	freq(iclk)/16	625 KHz
2	freq(iclk)/8	1.25 MHz
3	freq(iclk)/4	2.5 MHz

Bit 4-3   Update Select[1:0]   In the auto mode, the frequency of updating each register is determined by the setting on these bits.

Update Select[1:0]	Update Frequency	Update Frequency (with tadcclk = 156 KHz)	Update Frequency (with tadcclk = 5 MHz)
0	freq(tadcclk)/2048	76 Hz	2.4 KHz
1	freq(tadcclk)/1024	152 Hz	4.8 KHz
2	freq(tadcclk)/512	304 Hz	9.8 KHz
3	freq(tadcclk)/256	609 Hz	19.5 KHz

Bit 2-1   Channel Select [1:0]   This channel selection is only valid for manual mode (i.e., Mode = 0).

Channel Select[1:0]	Selected Channel
0	senin[0]
1	senin[1]
2	senin[2]
3	TADCREF

Bit 0   Mode   When mode = 0, then the manual mode is selected. In this manual mode, the channel selected must be done by the firmware, by setting the desired channel selection in bit 2 & bit1. When mode = 1, then the automatic mode is selected. In this auto mode, the selection for the analog mux will be performed automatically by the hardware, and the data on three registers corresponding to each channel will be updated accordingly.

Note: *In automatic mode, only channel 0, 1, and 2 will be automatically selected. Thus, in order to obtain the value for TADCREF, manual mode must be used, and "Channel Select" must be set to 3.*

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
TADC Instant Data *(TADCInsData)* **$01FF8003**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
TADC Instant Data *(TADCInsData)* **$01FF8002**	(Not Used)	(Not Used)	Data[5] (Read Only)	Data[4] (Read Only)	Data[3] (Read Only)	Data[2] (Read Only)	Data[1] (Read Only)	Data[0] (Read Only)	Rst. Value xx000000b Read Value 00h

Bit 5-0   Data   The data in this register comes directly from TADC. Thus, it gives the instant value from TADC regardless of the TADC operation mode.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
TADC Ch 0 Data *(TADCCh0Data)* **$01FF8005**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
TADC Ch 0 Data *(TADCCh0Data)* **$01FF8004**	(Not Used)	(Not Used)	Data[5] (Read Only)	Data[4] (Read Only)	Data[3] (Read Only)	Data[2] (Read Only)	Data[1] (Read Only)	Data[0] (Read Only)	Rst. Value xx000000b Read Value 00h

Bit 5-0   Data   This data is only valid when the auto mode is selected.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
TADC Ch 1 Data *(TADCCh1Data)* **$01FF8007**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
TADC Ch 1 Data *(TADCCh1Data)* **$01FF8006**	(Not Used)	(Not Used)	Data[5] (Read Only)	Data[4] (Read Only)	Data[3] (Read Only)	Data[2] (Read Only)	Data[1] (Read Only)	Data[0] (Read Only)	Rst. Value xx000000b Read Value 00h

Bit 5-0   Data   This data is only valid when the auto mode is selected.

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
TADC Ch 2 Data *(TADCCh2Data)* **$01FF8009**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
TADC Ch 2 Data *(TADCCh2Data)* **$01FF8008**	(Not Used)	(Not Used)	Data[5] (Read Only)	Data[4] (Read Only)	Data[3] (Read Only)	Data[2] (Read Only)	Data[1] (Read Only)	Data[0] (Read Only)	Rst. Value xx000000b Read Value 00h

Bit 5-0   Data   This data is only valid when the auto mode is selected.

8.2.3   Firmware Operation

General setup:

1. Select the desired tadclk frequency
2. Select the desired mode of operation

In manual mode:

3_{mm}. Select the right channel ("TADCCtrl[2:1]")
4_{mm}. Wait for at least "tadclk" period * 256 for TADC output to be stable
5_{mm}. Read TADC data from "TADCInsData" register. (Then, go to step 3_{mm} to select other channel)

In automatic mode:

3_{am}. Select the desired update frequency ("TADCCtrl[4:3]")
4_{am}. Wait for at least TADC update period * 3 (only need to wait once after the above step. This wait guarantees each channel has been selected and its value has been stored in the corresponding register)
5_{am}. Read TADC data from "TADCCh1Data", "TADCCh2Data", "TADCCh3Data" at any time for latest data.

9. BI-LEVEL RESOLUTION CONVERSION

9.1   Functional Description

The bi-level resolution conversion logic in the MFC2000 performs expansion (converting from lower to higher resolution) and reduction (converting from higher to lower resolution) of horizontal image data. Vertical line ORing can also be performed for image reduction in the vertical direction. There are two data sources that can be selected for this bi-level resolution conversion block. One is T4/T6 Decompressor and another is the external memory. The data from the T4/T6 Decompressor is serially input to this bi-level resolution conversion logic. The data from the external memory is input to this bi-level resolution conversion logic through the DMA operation. The DMA operation for the data from the external memory uses the DMA channel 9. If the T4/T6 Decompressor is selected as the data source, the DMA channel 9 request is blocked. The data from this resolution conversion logic is output to the external memory also through the DMA operation. The DMA operation for the output data uses the DMA channel 8 (See Figure 9-1).

The MFC2K adds the following features to the BRC Block:

1.   A first and last black data detector to indicate the first and last halfword out of the BRC that contains printable data.
2.   The ability the change the ORing algorithm for hoizontal reduction.

Because the bi-level resolution conversion is a slave logic, the DMA channel 9 for inputting data from memory has the block size control function. Once the block size limit is reached during the DMA operation. The DMA Controller gives a signal to the bi-level resolution conversion logic that indicates the end of process. Then, the bi-level resolution conversion logic will deactivate the DMA request, disable the FIFO, and start the flush operation. After the flush operation the bi-level resolution conversion interrupt will be generated to the Interrupt Controller. Before CPU gets out of the interrupt routine, CPU needs to write the new block size into the DMA9BlockSize register of the DMA Controller (even the same block size) to clear the interrupt. Then, CPU needs to write a 1 and then, immediately write a 0 to bit 1 of the BiResConvCtrl register to tell hardware to convert a new line (Reset the Resolution Conversion Logic).

If the data from the T4/T6 Decompressor is serially input to this bi-level resolution conversion logic, this bi-level resolution conversion logic is treated as a part of T.4/T6 logic. The T.4 line length (instead of block size of DMA channel 9) and T.4 interrupt (instead of the bi-level resolution conversion interrupt) are used. Furthermore,, this bi-level resolution conversion logic is enabled by setting bit 5 in the 'T4Config' register in the T4/T6 Compression/Decompression Logic.

The flush operation depends on the FIFO direction. If the FIFO is being read by the local side and written by the system side, a flush will simply reset all of the internal FIFO pointers, thereby removing the data. If the FIFO is being written by the local side and read by the system side, a flush will cause a DMA request to be generated. In addition, if there is a single byte in the holding register, it will be treated as a halfword (the contents of the msbyte of the halfword are indeterminate). Because of this, it is important for firmware to first check for the presence of a single byte in the holding register (via the Byte Present bit in the FIFO Control register).

Each FIFO is composed of 4 halfwords, a holding register, and a control register. Firmware can randomly access all the FIFO registers under the following restrictions:

●   When the FIFO is enabled - reads may return invalid data, writes are blocked
●   When the FIFO is disabled - read/write operations are properly performed
●   Random access of the FIFO registers is useful in saving and restoring the state of the FIFO. This should only be done when the FIFO is disabled.

9.1.1   Horizontal Conversion

Conversion in the horizontal direction is performed in this logic block by adding or removing a programmable number of pixels at the position which decided by the bi-level resolution conversion algorithm. Pixels are added based on a programmable 7-bit Base-3 counter and the expansion mode bits (2 bits). Input pixel number needs to be a multiple of 16. The output pixel number of this block is also a multiple of 16.

Table 9-1. Untitled Table

The expected expansion ratio	the expansion mode	the programmed expansion ratio	the expansion result
100% $\leq$ X < 200%	0	X	$\approx$ X
200% $\leq$ X < 300%	1	X - 100%	$\approx$ X
300% $\leq$ X < 400%	2	X - 200%	$\approx$ X

Pixels are removed based on the same programmable 7-bit Base-3 counter used for the expansion. Data can be reduced down to 0.0457% (1/2187 x) and expanded up to 399.95% (~4x). To preserve image quality, it is recommended that an input line be reduced by no more than 2/3 times (66.67%) it's original resolution (this will ensure that 2 consecutive input pixels will never be removed).

The following block diagram depicts the BRC block diagram. The functions are broken up into the blocks that represent the imdividule VHDL that make up the BRC design. Note that the two blocks, Vertical ORING and BRC Out FIFO, outputs are tied together. This represents the fact that only one of these output modes are operational at time.
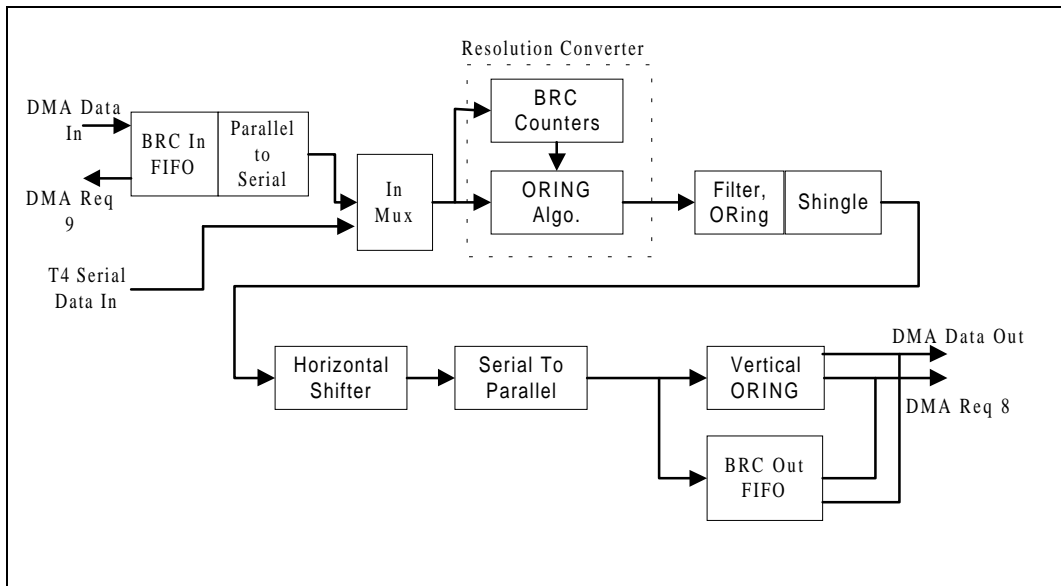
Figure 9-1: Bi-level Resolution Conversion Block Diagram

Conexant

9.1.2   Vertical Conversion

Expansion in the vertical direction is accomplished by the processor duplicating lines as needed to achieve the desired expansion ratio. Reduction in the vertical direction is accomplished by the processor either deleting lines or ORing lines together as needed to achieve the vertical reduction ratio. ORing of lines is also done by the resolution conversion logic. When ORing is enabled, each byte of the current line's line buffer data is ORed with the corresponding byte from the previous line. The DMA channel associated with the Resolution Conversion Block performs a read-modify (OR) -write operation. The DMA read and write accesses do not occur on consecutive bus cycles. The firmware should set the DMA channel start address for the line to be ORed equal to the address for the previous line. The threshold of the ORing function is selectable. If the threshold value is 0, all the black pixels in the current line will be used for the ORing function. In order to reduce the background image noise, the threshold value can be set from 1 to 7. If the threshold value is set to 2, the single black pixels and 2 contiguous black pixels in the current line will not be used for the ORing function.

9.1.3   Shingling

Shingling is a process in which color intensity was built up in the course of several passes of the head for the inkjet printing. The objective is to print a portion of the data on each of the passes, using a checkerboard mask to decide which pixels to print on each pass and which pixels to mask out. This allows time for dots of ink printed on one pass to dry somewhat before adjacent ink dots are printed on subsequent passes, thereby improving quality. The checkerboard varies with the level of shingling in use. This level is described as a percentage, with 50, 33, and 25 percent shingling levels having been discussed to date. Briefly:

50 percent shingling requires two passes of the head. The matrix below shows a section of the pixels on the page, with the number being the pass of the head on which the pixel in printed:

        1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2

        2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1

        1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2

        2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1

33 percent shingling requires three passes to build up full intensity:

        1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3

        3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2

        2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1

        1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3

25 percent shingling requires four passes:

        1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4

        2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3

        1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4

        2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3 2 4 1 3

Paper moves a portion of the height of the nozzle array between passes of the head, so different nozzles print different dots on a scanline. In this situation, the data for the scanline must be moved from the band buffers to the print buffers several times, with a mask applied along the way so that only part of the data gets through on each move. Note that printing is slower when shingling is in use, so bandwidth is not significantly affected. Also note that the mask changes between scanlines.

9.1.4   Horizontal Shift Function

Assume that the color inkjet head is used. There are 3 color groups. There are 16 nozzles for each color group on the inkjet head.

The image data is usually line-based. For the inkjet printing, we can't feed the line based data right into the inkjet head. The line-based image data needs to be rearranged to match the inkjet head firing order and will be no longer line-based. As you can see from the diagram at the next page, if nozzle 1 in the top nozzle group needs the 1st pixel of line 1, nozzle 2 in the top nozzle group needs the (NP)th pixel of line 2. If there is a total of 48 nozzles, we will need to group 3 halfwords (48 pixels) of data from different lines, different columns, and different color planes. Then, another 3 half-word group for the next firing cycle. These 3 half-word groups will be sent to the external print ASIC. The external print ASIC will map these 48 pixels to the right nozzle location and control the fire sequence and timing within one firing cycle.

In order to do inkjet printing, the line-based printing data from different color planes for each printing swath need to be prepared by Firmware. Under the assumption above, 48 lines (16 lines/color group) of data need to be prepared for one printing swath by Firmware. The vertical pitch (VP) in the diagram below is taken care of by the Firmware and the DMA operation.

Then, Bi-level Resolution Conversion block fetches the line-based printing data through DMA operation. The line-based printing data is serially processed in the Bi-level Resolution Conversion block. The process order is bi-level resolution conversion -> singling -> horizontal shift -> oring (optional). After these processes, the shifted line-based print data will be put back to the printing buffer by the DMA operation. The nozzle pitch (NP) and horizontal pitch (HP) in the diagram below are taken care of by the horizontal shifter in the Bi-level Resolution Conversion block. For example, the NP is 8 pixels and HP is 18 pixels. We want to prepare the shifted line-based printing data for nozzle 1 of the top group in the diagram below, the horizontal shifter will insert (8+18) zeros in the beginning of this line. If we want to prepare the shifted line-based printing data for nozzle 2 of the top group in the diagram below, the horizontal shifter will insert 18 zeros in the beginning of this line. If we want to prepare the shifted line-based printing data for nozzle 1 of the middle group in the diagram below, the horizontal shifter will insert 8 zeros in the beginning of this line. If we want to prepare the shifted line-based printing data for nozzle 2 of the middle group in the diagram below, the horizontal shifter will insert nothing in the beginning of this line. The actual shifting amount for each line needs to be changed according to the inkjet head which you are using and the mechanical orientation. All the pixels needed for one firing cycle will be at the same column after the horizontal shift process. The number of zeros inserted in the beginning of lines is programmable from 0 to 63. If extra zeros (>63) are needed, multiple of 16 zeros need to be prepared into memory by firmware.

Finally, the Bit Rotation block gets the shifted line-based printing data through the DMA operation. The Bit Rotation block groups 48 pixels at different lines and at the same column to 3-halfword data for one firing cycle (See the Bit Rotation section for details).
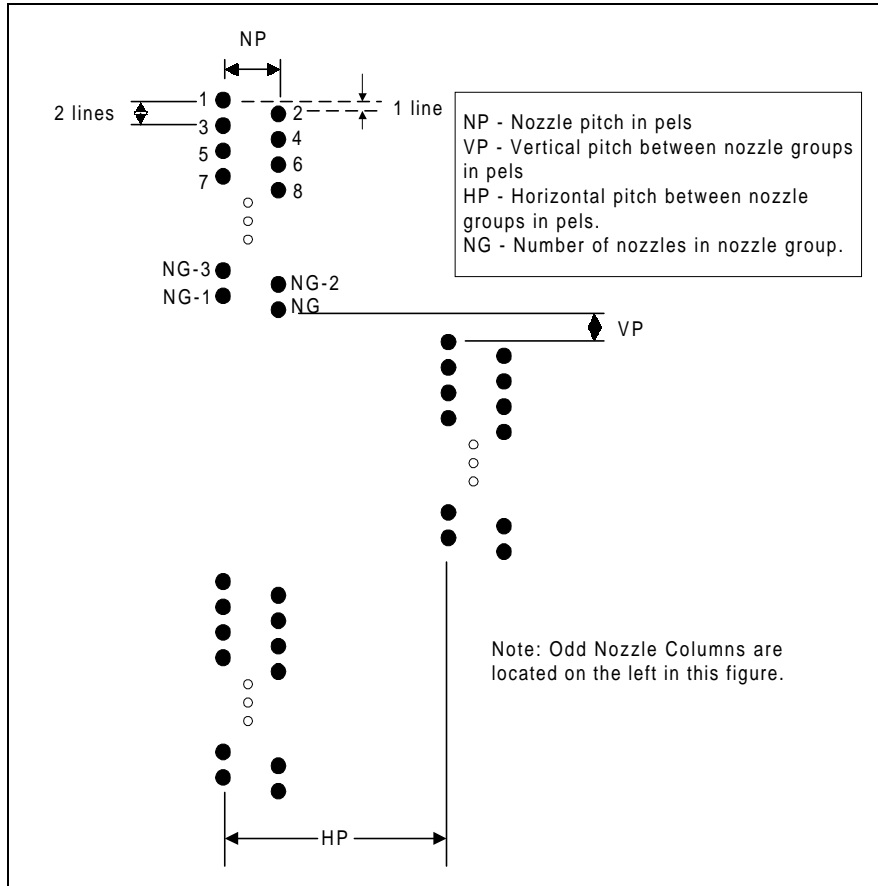
Figure 9-2. The Physical Nozzle Diagram for Typical Inkjet Heads

The boundary condition also needs to be considered. The line length (pixels/line) of the bi-level resolution conversion input and output data is a multiple of 16 (See the bi-level resolution conversion section for details). If we insert different number of zeros in the beginning of lines, all the lines will end at different columns (not end at the same halfword boundary). Therefore, control bits need to be used. It indicates the total number of halfwords of zeros that needs to be added either at the beginning and at the end of each line (See the diagram below). 0 - 4 halfwords can be set in the register.
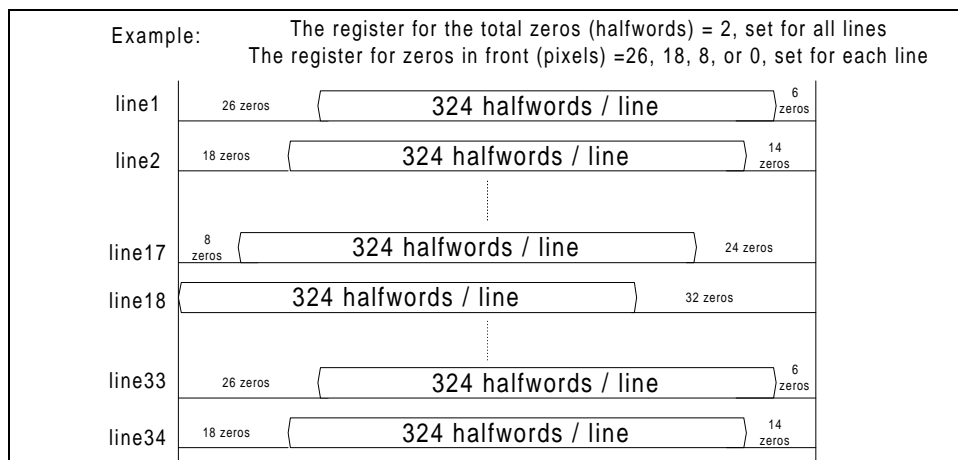
Figure 9-3. Untitled Figure

9.1.5   First and Last Black Data Detector Design

Below is a schematic representation of the Black Data Detector. The counter and registers get cleared at the start of each line. On the left side of the diagram is the half word counter that is incremented each time a half word is pushed into the FIFO (lcl_nxt). When the line starts, each time a lcl_nxt or lcl_wr shifts data into the output FIFO holding register, the serial data is checked for black data. If black data is detected, the *End Of White* flag is set. Once set, the logic waits until the counter is incremented, (the holding register is pushed into the FIFO data register). Then the new count value is loaded into the First Black Data Register.

As data continues to be shifted into the FIFO, the logic at the bottom of the Figure 1-2-2 continues to check for black data. Each time black data is detected the *Blk Data Det* flag is set. After that half word is loaded into the FIFO data register (counter counts), the counter value is loaded into the Last Black Data Register.

9.2   Register Description

External memory setup

• The external memory is selected as the data source by setting bit 2 in the BiResConvCtrl register to one in the Bi-level Resolution Conversion block.
• The block size of the data in the external memory for the Bi-level resolution conversion is set in the DMA9BlockSize register of the DMA Controller.
• Vertical Line ORing for the resolution converted data is enabled by setting bit 0 in the BiResConvCtrl register in the Bi-level Resolution Conversion block.
• The Bi-level resolution conversion is initiated by setting bit 1 (first to 1 and then, to 0) in the BiResConvCtrl register in the Bi-level Resolution Conversion block.

T4/T6 Control

• T4/T6 Decoder Bi-level resolution conversion is enabled by setting bit 5 in the 'T4Config' register in the T4/T6 Compression/Decompression Logic.
• Vertical Line ORing for the T4 Decoded data is enabled by setting bit 3 in the 'T4Control' register in the T4/T6 Compression/Decompression Logic. Vertical ORing for T4 decoded data is only valid if resolution conversion is also enabled.

Programming the Expansion/reduction Ratio

The Bi-level Resolution Conversion Ratio register (BiResConRatio) controls the horizontal resolution conversion expansion or reduction ratio. The BiResConRatio register is cleared on Reset. The value programmed into the register is the number of pixels to add or remove per 2187 pixels (expressed as a 7 digit base-3 number).

Input data FIFO Control register for the data from the external memory (for DMA channel 9):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Input Data FIFO Control *(BiRCInFIFOCtrl)* **$01FF807B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Bi-level Resolution Conversion Input Data FIFO Control *(BiRCInFIFOCtrl)* **$01FF807A**	(Not Used)	Holding Register Full	(Not Used)	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0x00000b Read Value 00h

Bit 15	This bit indicates that the FIFO is active and capable of generating DMA requests.
Bit 14	This bit is essentially a copy of the DMA Request output signal and indicates that the FIFO wishes to transfer data.
Bit 13	This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.
Bit 12-10	This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 0 to 4. Values greater than 4 are treated as 4.
Bit 9-8	This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.
Bit 6	This bit indicates that there is a full word in the holding register
Bit 4-2	This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.
Bit 1-0	This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

> *Note: It is strongly recommended that the FIFO be disabled before firmware writes to the FIFO Control register. The FIFO Control register contains data that is essential to the FIFO's operation. If this data is changed while the FIFO is operating, unpredictable operation will result.*

Input data Holding Register for the data from the external memory (for DMA channel 9):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Input Data Holding Register *(BiRCInHold)* **$01FF8079**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Bi-level Resolution Conversion Input Data Holding Register *(BiRCInHold)* **$01FF8078**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

Input data FIFO halfword[3:0] for the data from the external memory (for DMA channel 9):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Input Data FIFO *(BiRCInFIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Bi-level Resolution Conversion Input Data FIFO *(BiRCInFIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

Address assignment for input FIFO halfword[3:0]

The Input FIFO halfword	The register name	Address
Input FIFO halfword 3	*BiRCInFIFO3*	01FF8077-76
Input FIFO halfword 2	*BiRCInFIFO2*	01FF8075-74
Input FIFO halfword 1	*BiRCInFIFO1*	01FF8073-72
Input FIFO halfword 0	*BiRCInFIFO0*	01FF8071-70

Output data FIFO Control register for the data to the external memory (for DMA channel 8):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Output Data FIFO Control *(BiRCOutFIFOCtrl)* **$01FF808B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Bi-level Resolution Conversion Output Data FIFO Control *(BiRCOutFIFOCtrl)* **$01FF808A**	(Not Used)	Holding Register Full	(Not Used)	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0x00000b Read Value 00h

Bit 15	This bit indicates that the FIFO is active and capable of generating DMA requests.
Bit 14	This bit is essentially a copy of the DMA Request output signal and indicates that the FIFO wishes to transfer data.
Bit 13	This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.
Bit 12-10	This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 0 to 4. Values greater than 4 are treated as 4.
Bit 9-8	This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.
Bit 6	This bit indicates that there is a full word in the holding register
Bit 4-2	This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.
Bit 1-0	This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

> **Note**: *It is strongly recommended that the FIFO be disabled before firmware writes to the FIFO Control register. The FIFO Control register contains data that is essential to the FIFO's operation. If this data is changed while the FIFO is operating, unpredictable operation will result.*

Output data Holding Register for the data to the external memory (for DMA channel 8):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Output Data Holding Register *(BiRCOutHold)* **$01FF8089**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Bi-level Resolution Conversion Output Data Holding Register *(BiRCOutHold)* **$01FF8088**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

Output data FIFO halfword[3:0] for the data to the external memory (for DMA channel 8):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Output Data FIFO *(BiRCOutFIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Bi-level Resolution Conversion Output Data FIFO *(BiRCOutFIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

Address assignment for output FIFO halfword[3:0]

The output FIFO halfword	The register name	Address
Output FIFO halfword 3	*BiRCOutFIFO3*	01FF8087-86
Output FIFO halfword 2	*BiRCOutFIFO2*	01FF8085-84
Output FIFO halfword 1	*BiRCOutFIFO1*	01FF8083-82
Output FIFO halfword 0	*BiRCOutFIFO0*	01FF8081-80

Bi-level Resolution Converter Ratio Registers:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Resolution Conversion Ratio *(BiResConRatio)* **00FF807D**	Add/Delete 729 or 1458 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 729 pixels 3:= 1458 pixels		Add/Delete 243 or 486 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 243 pixels 3:= 486 pixels		Add/Delete 81 or 162 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 81 pixels 3:= 162 pixels		Add/Delete 27 or 54 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 27 pixels 3:= 54pixels		Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Bi-level Resolution Conversion Ratio Control *(BiResConRatio)* **01FF807C**	Add/Delete 9 or 18 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 9 pixels 3:= 18 pixels		Add/Delete 3 or 6 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 3 pixels 3:= 6 pixels		Add/Delete 1or 2 pixels per 2187 input pixels 0:= 0 pixels 1:= Not allowed 2:= 1pixels 3:= 2 pixels		Expansion/Reduction Mode 0=Exp. 1=Red.	Simple OR Enable	Rst Value 00h Read Value 00000000 b

Bit 15-2	Sets Ratio for Reduction/Expansion of image data
Bit 1	Sets Expansion or Reduction mode
Bit 0	Turns on simple oring mode of 2 pixels for horizontal reduction

Bi-level Resolution Conversion Control Register:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Bi-level Res. Conversion Control *(BiResConCtrl)* **01FF807F**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Flush FIFO (W) Flush Status (R)	Rst Value xxxxxxx0b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Bi-level Res. Conversion Control *(BiResConCtrl)* **01FF807E**	Expansion Mode 0 = 100% ≤ X < 200% 1 = 200% ≤ X < 300% 2 = 300% ≤ X < 400%		ORing Threshold Setting 0= no threshold 1-7= threshold value 1-7			Data Source Selection 0=T4/T6 1=Ext. Mem.	New line Write 1 and then, 0 → convert a new line	Enable Or of orig. line buffer with current scan line	Rst Value 00h Read Value 00h

Bit 8		For write, 1 → flush FIFO. For read, 1 → flush needed.
Bit 7-6		These two bits sets the resolution conversion hardware's expansion range. If the value of these two bits is 'n' and the expected expansion ratio is X, the ratio 'X-(n* 100%)' needs to be set into the BiResConRatio register.
Bit 5-3		n = the threshold value for the ORing function, n = 0-7. It means that Oring is done to (n+1) contiguous 1's only when bit 0 is set to 1.
Bit 2	Data Source selection	When this bit is set to 0, data is serially from the T4/T6 decompressor. When this bit is set to 1, data is from the external memory through the DMA operation (DMA channel 9).
Bit 1	Convert a new line	Resolution Conversion Logic is reset. Firmware needs to write a 1 and then, immediately write a 0 to tell hardware to convert a new line.
Bit 0		Enable OR of orig. line buffer with current scan line (ScanOrEnb) (This bit is only used when the external scan device is selected as the data source of the resolution conversion). 1 = enable Bi-level Vertical OR'ing of the scanner line buffer data. This bit can be used to accomplish normal mode OR'ing and vertical reduction.

> *Note*: *If you don't want to do image expansion or image reduction, the resolution conversion ratio is 100%. You need to program bits[7:6] of the BiResConCtrl register to '00b' and program the BiResConRatio register to '0000h'.*

Shingling Mask Register:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Shingling Mask Register *(ShinglingMask)* $01FF808D	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Mask bit 11	Mask bit 10	Mask bit 9	Mask bit 8	Rst Value xxxx1111h Read Value 0Fh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Shingling Mask Register *(ShinglingMask)* $01FF808C	Mask bit 7	Mask bit 6	Mask bit 5	Mask bit 4	Mask bit 3	Mask bit 2	Mask bit 1	Mask bit 0	Rst Value FFh Read Value FFh

This is a loadable circular right-shift register which firmware can load a mask. A 1 in a mask bit means to let that pixel pass through and A 0 in a mask bit means to block that pixel and output 0 for that pixel. The output of this shift register (LSB) gates with the serial data from the resolution convertor on each bi-level resolution conversion clock. The shift register is clocked in sync with the serial to parallel converter so that the mask rotates in sync with the serial data. This 12-bit shingling mask register supports 50, 33, and 25 percent shingling. For example, for the 33 percent shingling, firmware should load 001001001001, 010010010010, or 100100100100, depending on the scanline and pass in question.

The number of zeros inserted in a line for the Horizontal Shifter:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
The Number of Zeros in a line *(HSZeroNo)* $01FF808F	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	data bit 2	data bit 1	data bit 0	Rst Value xxxxx000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
The Number of Zeros in a line *(HSZeroNo)* $01FF808E	(Not Used)	(Not Used)	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xx000000b Read Value 00h

Bit 10-8   The total number of zeros (halfwords) in a line

0 = nothing is added
1 = 1 halfword (16 zeros)
2 = 2 halfwords (32 zeros)
3 = 3 halfwords (48 zeros)
4 = 4 halfwords (64 zeros)
5-7 = not used

Bit 5-0   The number of zeros (pixels) in the beginning of a line, from 0 to 63 zeros

Note: If value 5, 6, or 7 is put into this register, 4 halfword (64 zeros) will be used.

First Black Data:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
The First Black Data Location Count *(FirstBlkDatCnt)* $01FF806D	(Not Used)	(Not Used)	(Not Used)	(Not Used)	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxxx0000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
The First Black Data Location Count *(FirstBlkDatCnt)* $01FF806C	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00000000b Read Value 00h

Bit 11-0   The First Black Data Location   Gives the half word count that black data first appeared in output of the BRC. If the first half word contains black data the register will remain at 000H. If the second half word contains data the register will contain 0001H Etc. If black data in not present in the entire line (all white line), register will be equal to the line length.

First Half Word Location = Register Value + 1.

> *Note*: Starting a new line will reset the counter

Last Black Data:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
The Last Black Data Location Count *(LastBlkDatCnt)* $01FF806F	(Not Used)	(Not Used)	(Not Used)	(Not Used)	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxxx0000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
The Last Black Data Location Count *(LastBlkDatCnt)* $01FF806E	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00000000b Read Value 00h

Bit 11-0   The Last Black Data Location   Gives the half word count that black data last appeared in output of the BRC. If black data in not present in the entire line (all white line), register will read 000H. If only the first half word contains black data the register will read 001H. If the second half word contains data the register will contain 0002H Etc. If the last half word contains black data the register will be equal to the total number of half words out of the BRC for that line. Last Half Word Location = Register Value.

> *Note*: Starting a new line will reset the counter

Total Colored Pixel Counter:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Total Colored Pixels Printer *(TotalBlkDatCntr)* **$01FF806B**	(Not Used)	(Not Used)	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxxx0000b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Total Colored Pixels Printer *(TotalBlkDatCntr)* **$01FF806A**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00000000b Read Value 00h

Bit 13-0   Total Colored Pixel Counter   Reading this register will return the total number of colored pixels printed since the last time the register was cleared. Writing to this register will clear it.

9.3   Resolution Conversion Programming Examples

Figure 9-4 is a flowchart that shows the procedure for determining the resolution conversion register values for both expansion and reduction modes:
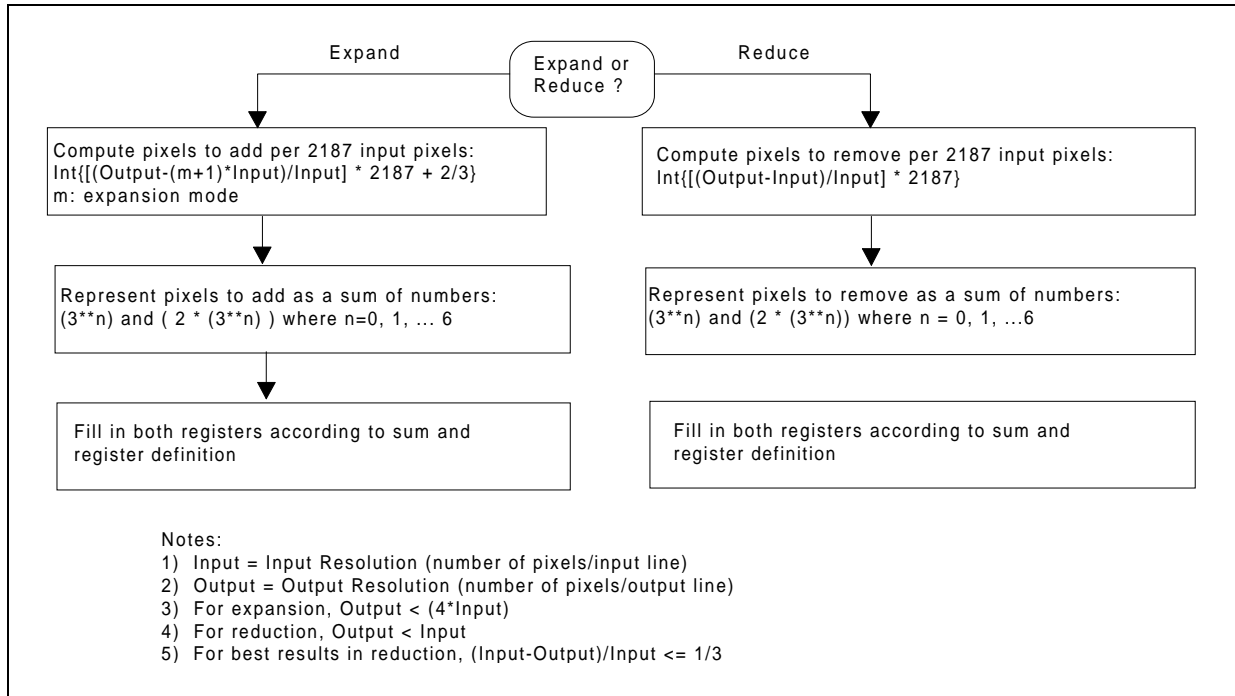
Figure 9-4: Resolution Conversion Programming

The number of pixels to add in expansion mode is determined from the integer portion of:

Pixels to add per 2187 input pixels =
{[(Output pixels per line - (m+1) * Input pixels per line)/input pixels per line] * 2187 +2/3}

m: the expansion mode

> **Note:**   2/3 is added in this equation in order to guarantee that the truncated result will be greater than or equal to the desired number of output pixels per line.

The number of pixels to remove in reduction mode is determined from the integer portion of:

Pixels to remove per 2187 input pixels =
{[(Input pixels per line - Output pixels per line)/input pixels per line] * 2187}

> **Note:**   For best results, the number of pixels removed (per 2187 pixels) in reduction mode should be less than or equal to 729. {i.e., [(Input Pixels - Output pixels)/Input pixels] < = 1/3}

The number of pixels to add or remove per 2187 pixels is next converted to a 7 digit base-3 number. Each digit in the Base-3 number is weighted as:

	MSD						LSD
Digit n	0	1	2	3	4	5	6
Weight	1/3	1/9	1/27	1/81	1/243	1/729	1/2187

Note: *MSD = most significant digit, LSD = lease significant digit.*

Each digit can take on 3 possible values: 0, 1, or 2; corresponding to adding or removing $(0, 1$ or $2) * 3^{(6-n)}$ pixels for every 3^7 input pixels or, equivalently, adding or removing 0,1, or 2 pixels for every $3^{(n+1)}$ pixels.

Note: *The actual number of pixels output from the resolution conversion block will always be a multiple of 16 pixels.*

Example: Reduction of 2048 pixel/line to 1728 pixel/line (B4 to A4 Reduction)

Given:

Input Resolution = 2048 pixel/line

Output Resolution = 1728 pixel/line

Notes:
1.   Output Resolution must be a multiple of 16.
2.   (Input Resolution - Output Resolution)/Input Resolution < = 1/3 for best results (i.e., consecutive pixels will not be removed))

In this case, (Input Resolution - Output Resolution)/Input Resolution = 320/2048 = 5/32. Therefore, ideally 5 pixels will be removed for every 32 input pixels.

Compute the number of pixels to remove per 2187 input pixels:

Int{ [(Input-Output)/Input]*(2187) }   = Int{ [(2048-1728)/2048]*(3**7) }
                                         = Int { [320/2048]*2187 }
                                         = Int { 341.71875 }
                                         = 341

Therefore, 341 pixels will be removed for every 2187 input pixels.

Represent the number of pixels to add per 2187 input pixels using powers of 3:

Using Figure 9-1 determine which combination of powers of 3 digits are required to represent 341. The table should be filled from top to bottom. The sum of the enabled digits should equal 341 as illustrated below:

$$341 = 243 + 81 + 9 + 6 + 2$$

Therefore, digits 243, 81, 9, 6, and 2 should be enabled in the table.

Table 9-2: Procedure to determine Pixels to remove

Pixels to Remove per 2187 Input Pixels	Yes/No
1458	N
729	N
486	N
243	Y
162	N
81	Y
54	N
27	N
18	N
9	Y
6	N
2	Y
1	N

Table 9-3 lists common resolution conversion values.

Table 9-3: Resolution Conversion Examples

Conversion dots/inch	Input dots/line	Output dots/line	Ideal # of pixels/line to insert (remove)	Pixels to insert (remove) per 2187	ResConMSD (hex value)	ResConLSD (hex value)
200 to 300	1728	2456	728	922	$8E	$28
200 to 300	1728	2400	672	851	$8A	$AC
200 to 360	1728	2952	1224	1549	$C8	$88
200 to 600	1728	3280	1552	1964	$F0	$CC
200 to 720	1728	3280	1552	1964	$F0	$CC
200 to 300	2048	2456	408	436	$2E	$28
200 to 360	2048	2952	924	966	$8F	$E0
200 to 600	2048	3280	1232	1316	$B8	$CC
200 to 720	2048	3280	1232	1316	$B8	$CC
200 to 200	2048	1728	(320)	(341)	$28	$BE

This page is intentionally blank

10. External Print ASIC Interface

The interface signals used between the MFC2000 and the external Print ASIC are described in this section. They are basically the external ARM system bus interface signals. Conexant's Alcore chip is an external Print ASIC and it has the P1284 master port to talk to the P1284 slave port in the Printer.

10.1 Interface Between the MFC2000 and External Print ASIC

The MFC2000 ASIC is the master that controls the system bus and all other control signals. The MFC2000 firmware configures and controls the external Print ASIC by setting registers in the external Print ASIC through the system bus. See Figure 10-1.

AUXCLK can be used as the clock base for the external Print ASIC. No matter AUXCLK or external OSC is used for the external Print ASIC, the external Print ASIC needs to resynchronize all the signals and accesses for the metastable problems caused by different clocks. The MFC2000 internally uses SIUCLK and ICLK which are different from the external Print ASIC.

The interrupt PRTIRQn is used to tell the MFC2000 that further settings or actions are needed for the external Print ASIC. The register bits in the external Print ASIC may be used to indicate which settings or actions need to be performed. If the extra interrupt channel is required, the external IRQ13n and IRQ11 can be also used for the external Print ASIC. These three interrupt input signals are resynchronized twice for the metastable issues.

The external Print ASIC needs to be designed with the compatible CPU read/write bus timing and DMA bus timing used in the MFC2000 ASIC.

The external DMA channel between the MFC2000 ASIC and the external Print ASIC is DMA channel 2. The external Print ASIC may get the print data from the bit rotation block if the bit rotation block is used or get the print data from the external memory if the bit rotation is done by host or the external Print ASIC.

1. The external Print ASIC gets the print data from the external memory. When the external Print ASIC generates the DMA request (DMAREQ2) to request the printing data, the MFC2000 will respond with the DMA acknowledge (DMAACK2) at the valid access cycle. At the same cycle, the MFC2000 will generate the read strobe, address, and the external chip select signals to read one byte or halfword of print data out of the external memory and put on the data bus. At the rising edge of the read strobe when DMAACK2 is high (DMAACK acts as a chip select), the external Print ASIC latches this one byte or halfword printing data in. If additional data bytes or halfwords are required by the external ASIC, it may leave the DMAREQ2 active. If no additional data is required, the ASIC should pull the DMAREQ2 inactive immediately after DMAACK goes low. DMA address will be incremented, decremented, or jumped during the DMA operation according to the fetch order of the print data in the external memory (See the DMA Controller Section).

> *Note: No matter the data bus is byte or halfword wide, the memory data bus size must match with the data bus size of the external Print ASIC. The Endian control bit for the DMA2 needs to be 0. The Endian mode can't be changed during the DMA operation.*

2. The external Print ASIC gets the print data from the bit rotation block.

   When the external Print ASIC generates the DMA request (DMAREQ2) to request the printing data, the MFC2000 will respond with the DMA acknowledge (DMAACK2) at the valid access cycle after the print data ready in the output register of the Bit Rotation Block. At the same cycle, the MFC2000 will generate the read strobe, address, and internal chip select signals to read one halfword of print data out of the output register of the Bit Rotation block and put on the data bus. The MFC2000's SIU may generate two access cycles for the byte-wide external data bus (the bit 8 of the DMA2Config register needs to be set to 0) or one access cycle for the halfword-wide external data bus (the bit 8 of the DMA2Config register needs to be set to 1). At the rising edge of the each read strobe when DMAACK2 is high (DMAACK acts as a chip select), the external Print ASIC latches this one byte or halfword printing data in. If additional data bytes or halfwords are required by the external ASIC, it may leave the DMAREQ2 active. If no additional data is required, the ASIC should pull the DMAREQ2 inactive immediately after DMAACK goes low. The DMA address is fixed during the DMA operation (See the DMA Controller section and the Bit Rotation section).

- CS4n, CS3n, or CS2n may be used as the chip select signal for accessing registers in the external Print ASIC (See Memory Map Section)
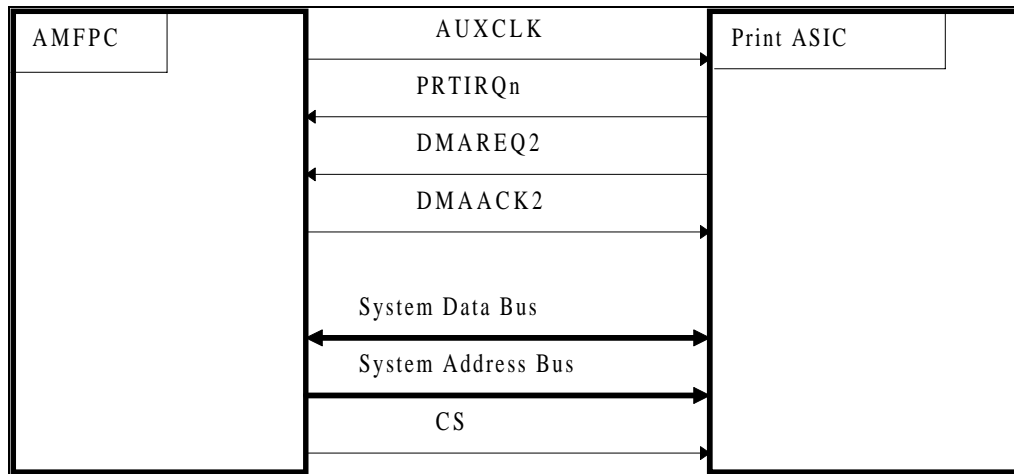
Figure 10-1. Print ASIC Interface

10.1.1 Examples

10.1.2 Motor Control Examples of the Inkjet Print Head Module

Vertical Movement

The vertical motor stepping control logic in the MFC2000 may be used for the Inkjet printing (See Section 12.1).

For Laser printing, this internal vertical motor stepping control logic is not used.

Horizontal Movement

The horizontal stepper motor control logic or the horizontal DC motor control logic should be in the external print ASIC.

Whenever the external print ASIC needs the new timer value(s) or the PWM value(s) which is used to control the time interval between steps or the DC motor speed, the external print ASIC generates the interrupt (PRTIRQn) to the MFC2000 ASIC. Then, the MFC2000 CPU will write the new timer value(s) to the register(s) in the external print ASIC in the interrupt subroutine.

11. Bit Rotation Logic

11.1 Functional Description

Bit Rotation Logic is responsible for preparing line-based image data to be used for a color or mono inkjet head. It is designed to be completely universal, so that any inkjet head may be supported regardless of nozzle configuration, provided there is only one nozzle per line of the printed image. An example of a typical 3-color inkjet head is shown.
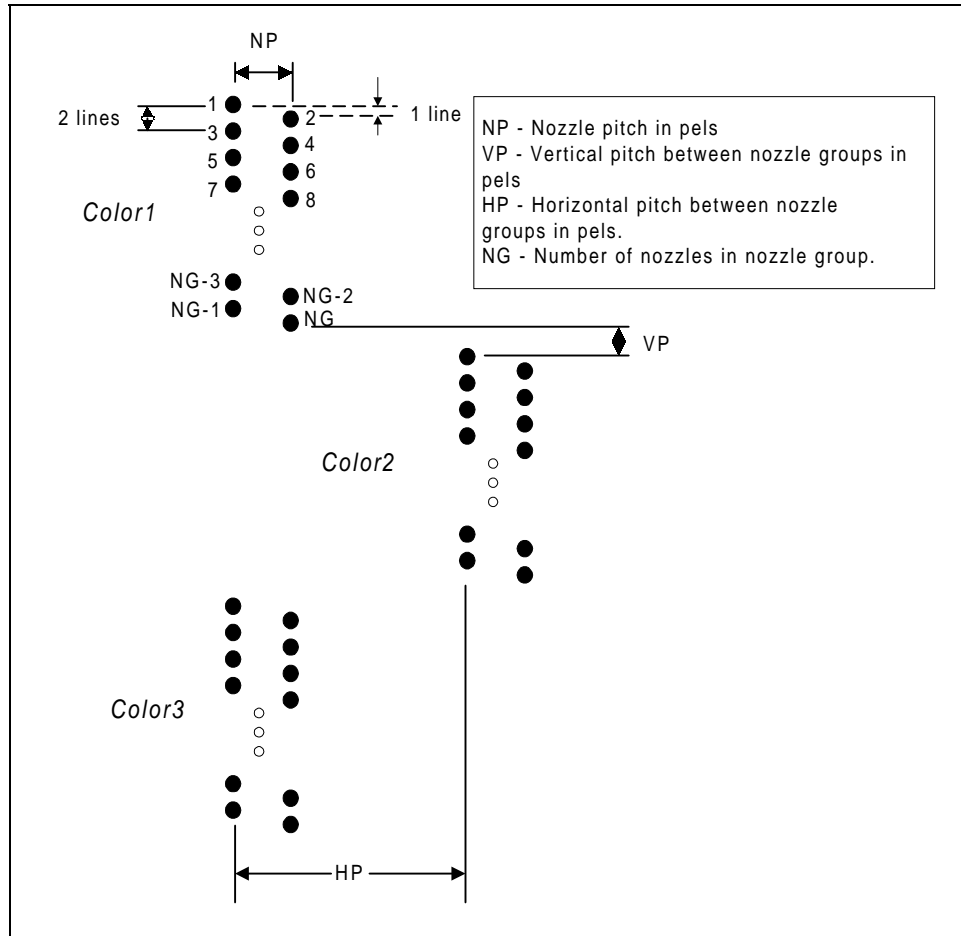
Figure 11-1. Nozzle Diagram of a Typical Programmable Inkjet Head

The inkjet head shown is only an example. There are many other configurations that may be supported. The Bi-Level Resolution Conversion block works in conjunction with firmware to ensure that all parameters that correspond to nozzle shifts are removed before the image data is deposited in the swath buffer for removal by the Bit Rotation Block. In addition, the Bit Rotation block does not need to distinguish the difference between mono or inkjet color modes. A few examples of nozzle configurations that can be supported are shown in the figures. These are only to show examples of the universal design.
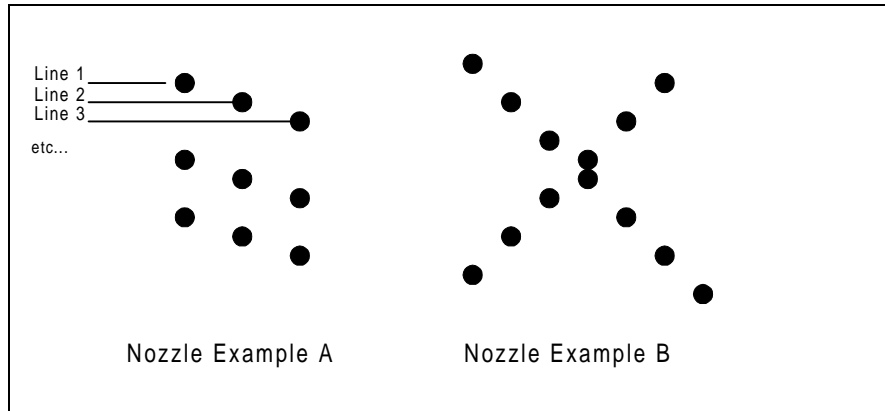
Figure 11-2. Examples of Nozzle Head Configurations

Because the firmware and Bi-Level Resolution Conversion block work in conjunction to remove all nozzle shift parameters, the Bit Rotation Block may always view the data in such a way that it views the nozzles as always being one column of N nozzles, where N corresponds to the number of nozzles on the inkjet head itself. In fact, the only configuration information that the Bit Rotation block needs to know is the number of nozzles on the inkjet head, the shuttle direction, the width of the line being printed, and the warp. The figure shows how the Bit Rotation Block views the swath regardless of the actual physical nozzle configuration.
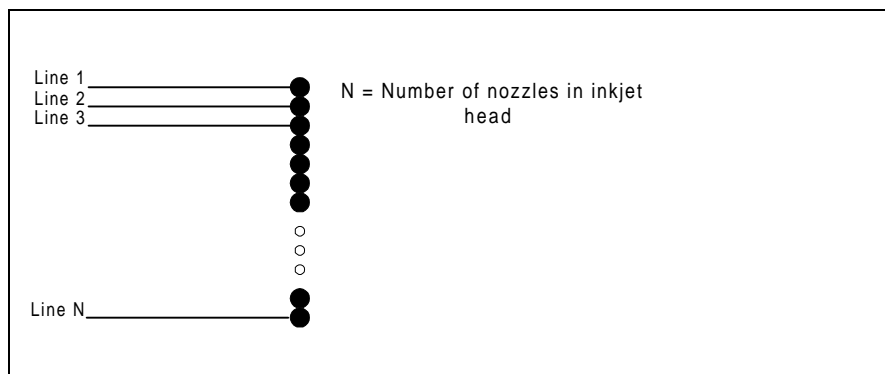
Figure 11-3. Nozzle Configuration by Bit Rotation Block (Regardless of Physical Nozzle Configuration)

The BRB works in conjunction with 2 DMA channels of the DMA controller. Within the MFC2000, the DMA channels used are channels 2 and 3. Channel 3 is used to "fetch" data from the external swath buffer that resides in the system's image memory, and is referred to as the Swath Fetcher DMA channel from here on. DMA Channel 2 is used to transfer "packed" or rotated image data to the External Print ASIC, and will be referred to as the Bit Packer DMA channel from here on.

The BRB uses a local memory (in the BRB block) to perform its rotation operation. Words are "fetched into the local memory, and then bits are picked out of the halfword and then "packed" or shifted into an output register, in effect, rotating the swath image data by 90 degrees. The internal buffer size within the Bit Rotation Block is 512 x 1 halfword (16 bits). This buffer is further subdivided into two separate 256 x 16 bit "banks" for reasons of system performance. While one 256 x 16-bit buffer is being operated upon, the other 256x16-bit buffer may be filled by the Fetcher DMA channel. Because of this, the system only needs to keep up with the print rate, which in the worst case scenario, is 256 bits every fire cycle if 256 nozzles are used.

Once the data is prepared for output by the Bit Rotation Block, it may be retrieved by the External Print ASIC via the Bit Packer DMA channel. The BRB issues the ready signal to indicate that a halfword of data is ready for retrieval to the DMA Controller. Then, the DMA Controller will grant the DMA channel 2 access cycle by activating the DMAACK2 when system bus is available if DMAREQ2 is active and BRB output ready signal is active.

11.2 Block Diagram

A high-level block diagram of the Bit Rotation Block is shown. Note that the internal RAM is double-banked, so that these operations of fetching and packing can occur simultaneously (however, they are not allowed to occur to/from the same bank at the same time).

brb_regs—This block is used to hold the internal registers of the BRB. The registers are split into setup registers, operation registers, and test registers. Each group has a separate chip select sent by the SIU.

brb_memif—This block has three separate interfaces: the SIU, the Swath Fetcher, and the Bit Packer. If a bit rotation operation is in progress (rotateStart = 1 in the RotCtrl register), SIU local memory access is locked out, and only the swath Fetcher and Bit Packer blocks may access the memory at this time. If the rotateStart bit is not set, then the SIU has access to the 1 kB memory through locations 01FF9000h-01FF93FFh.

Swath Fetcher—This block works in conjunction with channel 3 of the DMA Controller to transfer image data from the swath buffer to the local SRAM for Bit Rotation.

Bit Packer—This block is responsible for the actual rotation of image data. It performs this rotation by picking bits out the local SRAM and constructing the final rotated output data for the External Print ASIC.

brbsync—The brb_regs and Swath Fetcher blocks operate at the SIUCLK frequencies, while the Bit Packer and the brb_memif blocks operate at IHSCLK frequencies. The purpose of this block is to synchronize communications between the internal blocks of the BRB.
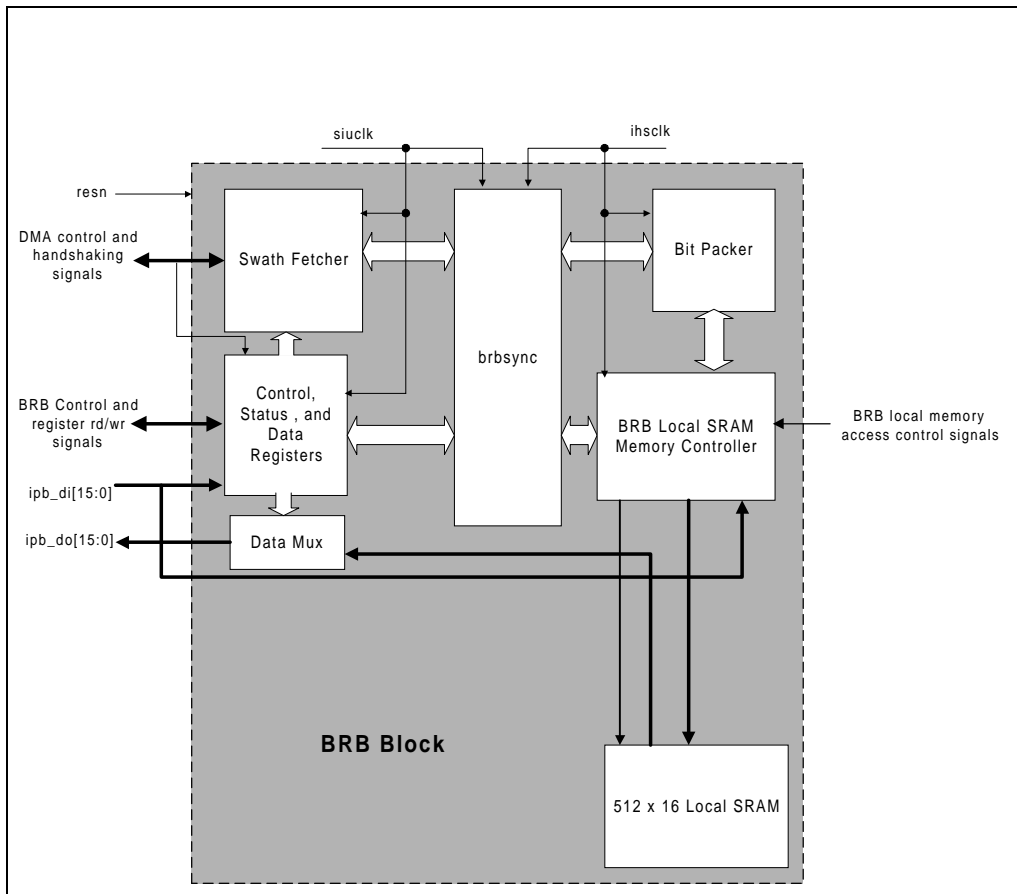
Figure 11-4. MFC2000 Bit Rotation Block Diagram

11.2.1 Swath Fetcher

The Swath Fetcher block is responsible for initiating as well as controlling the order of the transfer of data from the swath buffer to the internal RAM of the Bit Rotation Block. There are several control signals provided to the Swath Fetcher DMA channel for controlling what order the channel will read data out of image memory. The Fetcher DMA will not support burst mode. This is due to the fact that in order to use burst mode, the DRAM that comprises the image memory must be accessed using page mode. When accessing DRAM for the Bit Rotation block, the sequential data stream rarely contains elements out of the same page of DRAM. Because of this, in most cases, only single-access cycles can be realized when reading data out of the image memory for the Bit Rotation Block.

The order in which data is fetched out of the swath buffer depends on shuttle direction only. The way data is fetched out of the swath buffer is best described pictorially. In the figure, each cell corresponds to a halfword (16 bits) in memory. The gray portion represents the printable image region, and a full line corresponds to the warp of the swath buffer. The general order in which halfwords are fetched from the swath buffer is indicated by the arrows. The starting address value that the firmware must program into the Swath Fetcher DMA channel depends on the shuttle direction. While printing in a left-to-right fashion (when viewing the print on the page), the value must correspond to the halfword of the upper left-hand corner of the image in the swath buffer (in the example shown, this address would be base+6H. While printing in a right-to-left fashion, the value must correspond to the upper right-hand corner of the image (in the figure, this would be base+AH). The value that would be programmed in the Rotation Line Length (RotLine) register in this example would be 0005H (one less than the actual number of bytes that comprise the image).
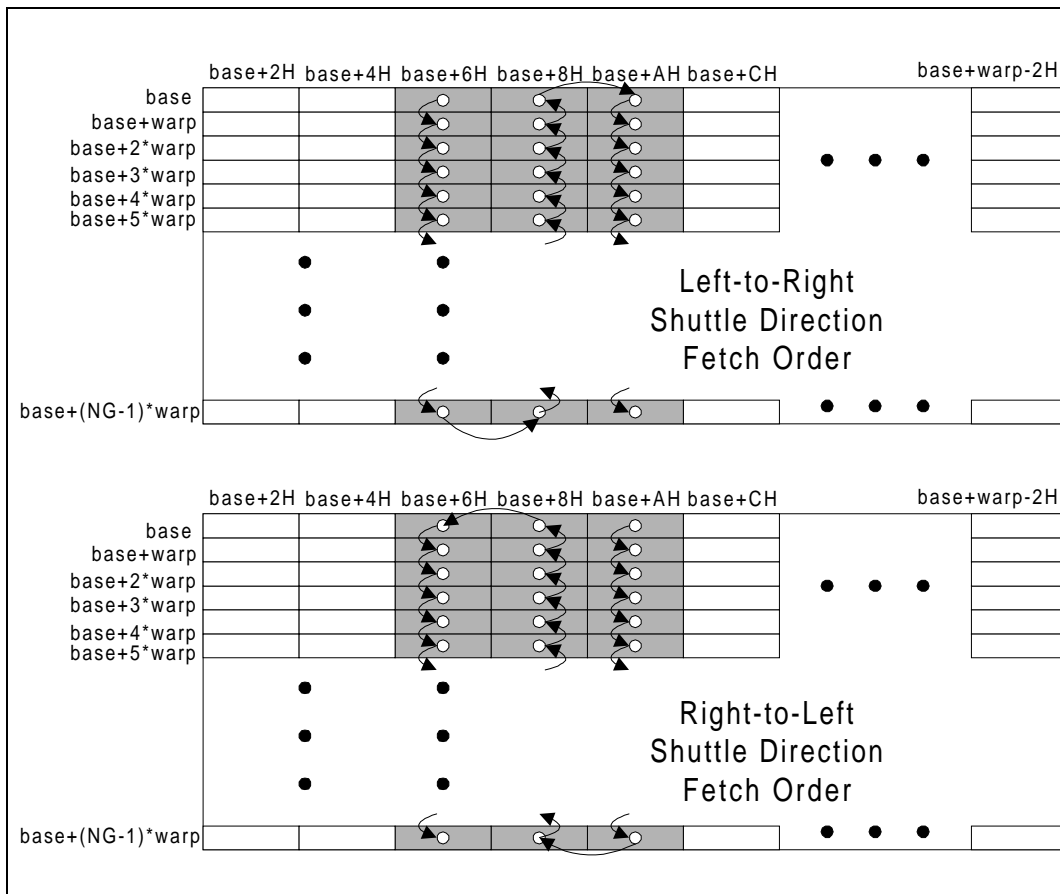
Figure 11-5. Fetcher DMA Channel Fetch Order

11.2.2 Bit Packer

The Bit Packer block is responsible for performing the actual rotation of the image data. It accomplishes this task by "picking" the bits out of the internal Bit Rotation RAM one bit at a time and packing them into the appropriate format such that the final output image is rotated 90 degrees. The Bit Packer circuits also contain the bit counter that describes the total number of printable bytes contained in the swath. The overall control of the print operation is controlled by this block. The parameters that are required by this block are: the Line Length (LL), the Nozzle Number (NN), the shuttle direction, and the packing order. In addition, this circuit also detects when there is an underrun condition. An underrun condition occurs when the system is not able to keep up with the print rate. Finally, the interface to the External Print ASIC is provided by this block.

11.2.3 Local Bit Rotation RAM

This memory is 512x16. Functionally, this memory consists of 2 banks of 256x16 memories, and only one of these buffers is used at a time for rotating image memory data. The double-buffering is used for reasons of system performance. When using the double-buffering mechanism, the system only needs to transfer the data at a rate that corresponds to the print rate. This is due to the fact that while one bank is being loaded with swath data, the other bank is being operated upon, or rotated and shipped to the External Print ASIC. This memory is accessible by three resources: the SIU, the Swath Fetcher, and the Bit Packer. When a bit rotation operation is in progress (when rotateStart in the ROTCTRL register is set to 1), then the SIU may not access this memory. During this time, both the Swath Fetcher and the Bit Packer may access this memory, but not the same bank simultaneously. When no bit rotation operation is in progress (the rotateStart bit is set to 0), then only the SIU has access to this memory through address locations 01FF9000h-01FF93FFh.

11.2.4 BRB Local SRAM Memory Controller (brb_memif)

The brb_memif block is responsible for controlling all accesses to Local BRB SRAM. During Bit Rotation operation, it will lock out all SIU accesses. At this time, it will allow both the Bit Packer and the Swath Fetcher blocks to access the SRAM. The Swath Fetcher block is a write-only resource, and the Bit Packer is a read-only resource. Arbitration to the SRAM by each of these resources is controlled with a rotating priority to ensure that neither the Swath Fetcher or the Bit Packer get locked out for any length of time.

11.2.5 Control, Status, and Data Registers (brb_regs)

This block provides all required registers used in Bit Rotation operation. These registers are described in detail later in section 11.3. The functional registers are spilt into two separate areas: Operation registers, and Setup registers. The Operation registers are those registers that are meant for access during Bit Rotation. The Setup registers are those register that are intended to be used once to set up the next print swath for printing. The Operation registers for the Bit Rotation block are located at 01FF8060h-01FF806Fh, however, only locations 01FF8060h-01FF8063h are used. The Setup Registers are located at 01FF8870h-01FF888Fh, however, only locations 01FF8870h-01FF8875h are used. The registers are summarized.

Register Address	Register Name	Summarized Description
01FF8060h	Rotation Control - *RotCtl*	Provides programmability of parameters, initiates Rotation, and provides status.
01FF8874h	Rotation Line Length - *RotLine*	Contains the value of the line length in terms of bytes.
01FF8870h	Nozzle Number - *RotNN*	Provides programmability of number of nozzles on inkjet head.
01FF8872h	BRB Warp - *BRBWarp*	Provides programmability of swath warp in increments of 32 bytes, up to 4096 bytes.
01FF8062h	Packing Register - *RotPack*	Provides the data for the External Print ASIC

11.2.6 BRB Synchronization

The Swath Fetcher and the brb_regs blocks operate at SIUCLK frequencies, while the brb_memif, and the Bit Packer circuits operate at ihsclk frequencies. The reason for this is to control power consumption and noise, as the higher SIUCLK frequency is not required for operation of the Bit Packer and the BRB Local SRAM Controller. Because of these different frequencies, a synchronization block is required to allow these blocks to communicate in a reliable fashion.

11.3 Register Description

Rotation Control Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Rotation Control Register (*RotCtrl*) **01FF8061**	Rotate Start (W) Rotate Status (R), 1:busy	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value 0xxxxxxxb Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Rotation Control Register (*RotCtrl*) **01FF8060**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Packing Order 0: odd then even 1: even then odd	Underrun 0: No underrun 1: Underrun occurred.	Packed-byte status 0: not ready 1: ready	Shuttle Direction 0: Left to Right 1: Right to left	Rst. Value xxxx0000b Read Value 00h

Bit 15

To start the bit rotation function, the CPU should write a 1 to this bit location. Reading this bit location indicates the status of the block rotation function. 0 = Not busy, 1 = busy. This bit will be set to 0 at the end of a print swath, and all bit rotation circuits will be reset with the exception of the BRB registers. If an unrecoverable error condition occurs within the system, then the BRB block may be reset by creating a falling edge on the RotateStart bit.

Bit 14-4

Not used

Bit 3

The packing order allows the order of the groups of odd and even bits to be switched.

Bit 3 = 0 => Odd bit is output first.
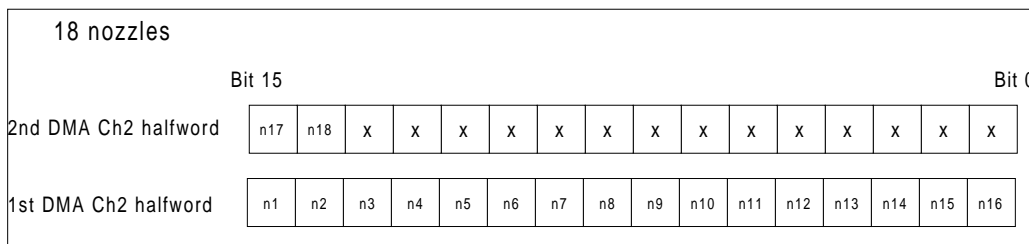
Bit 3 = 1 => Even bit is output first

```
18 nozzles

                       Bit 15                                                              Bit 0

2nd DMA Ch2 halfword   | n18 | n17 | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  |

1st DMA Ch2 halfword   | n2  | n1  | n4 | n3 | n6 | n5 | n8 | n7 | n10| n9 | n12| n11| n14| n13| n16| n15|
```
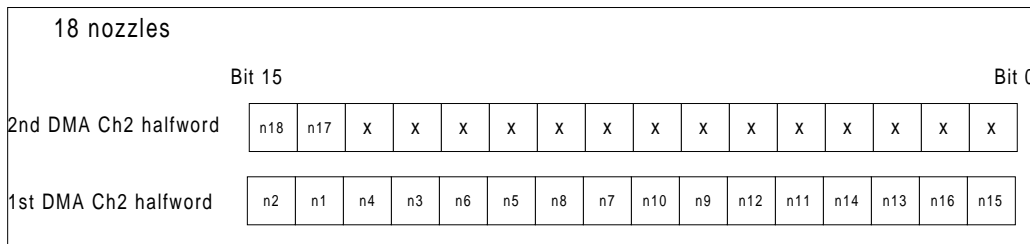
Bit2	This bit indicates whether a print underrun condition has occurred. Basically, an underrun condition exists if the system is not able to keep up with the print rate.
Bit 1	Once a halfword has been packed and is ready for output to DMA channel 2, a 1 will be indicated in this bit of the register. This bit directly reflects the state of the output signal brb_ch2rdy, and is intended only for status purposes. Once the data has been read, the bit is cleared automatically. The bit is also reset by creating a falling edge on the RotateStart bit.
Bit 0	This bit is written by the CPU and indicates which direction the print head is moving as viewed when looking at the print on the page. 0 = left to right shuttle motion 1 = right to left shuttle motion

Rotation Line Length Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Rotation Line Length Register *(RotLineLength)* **01FF8875**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	LL10	LL9	LL8	Rst. Value xxxxx000b Read Value 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Rotation Line Length Register *(RotLineLength)* **01FF8874**	LL7	LL6	LL5	LL4	LL3	LL2	LL1	LL0	Rst. Value 00h Read Value 00h

| Bits 15-11 | Not used |
| Bits 10-0 | The value in this field indicates the length in bytes of the actual printable data. The value programmed into this register is one less than the actual value in bytes. The printable data in a swath can be between 1 and 2048 bytes. |

Rotation Nozzle Number Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Rotation Nozzle Number Register *(RotNN)* **01FF8871**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Rotation Nozzle Number Register *(RotNN)* **01FF8870**	NN7	NN6	NN5	NN4	NN3	NN2	NN1	NN0	Rst. Value 00h Read Value 00h

Bits 15-8   Not used

Bits 7-0   This value describes the total number of nozzles on the inkjet head. The value written into this register is one less than the actual number of physical nozzles. The total number of nozzles that can be supported is between 1 and 256.

BRB Warp

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
BRB Warp Register *(BRBWarp)* **01FF8873**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	BW11	BW10	BW9	BW8	Rst. Value xxxx0000b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
BRB Warp Register *(BRBWarp)* **01FF8872**	BW7	BW6	BW5	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value 000xxxxxb Read Value 00h

Bits 15-12   Not used

Bits 11-5   These bits are used to define the warp of the swath, or the actual jump line size. The smallest resolution of the warp is 32 bytes. The value that is written into this register is the actual value that will be used in the jump. The warp value can range from 0000h to 0FE0h.

Bits 4-0   Not used

Packing Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Rotation Packed Data Register *(RotPackedData)* **01FF8063**	RP15	RP14	RP13	RP12	RP11	RP10	RP9	RP8	Rst. Value 00h Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Rotation Packed Data Register *(RotPackedData)* **01FF8062**	RP7	RP6	RP5	RP4	RP3	RP2	RP1	RP0	Rst. Value 00h Read Value 00h

Bits 15-0   This register shows the contents of the halfword content that will be sent to the External Print ASIC via the Bit Packer DMA controller channel.

11.4 Firmware Operation

11.4.1 Bit Rotation Setup and Operation

The following describes how to setup and operate the Bit Rotation Block for printing of image swaths. The following example assumes that there is a 8Mbyte DRAM residing in bank 0 of the system in question. It is also assumed that the swath buffer that resides in this DRAM is located at the base address of this device and grows up. Since the memory grows down from address 03000000h, the base address for the DRAM is at 02800000. In addition, I am assuming that the image that is to be printed is 72 bytes wide, has a warp of 96 bytes, shuttle direction is right-to-left, and the number of nozzles is 56.

Program the DMA Controller - Channel 2 and 3

1. Program the address of the Bit Rotation Packing Register (01FF8062h) into the DMA High Address Counter (01FF818Ah) and the DMACNT2LO (01FF8188h) and the DMA2CNTHI (01FF818Ah) registers:

    (DMACNT2LO) <= 8062h;
    (DMACNT2HI) <= 01ffh;

2. Program the DMA2CONFIG register (01FF81B2h). It must know that it is being used for Bit Rotation as well as for reads and halfword mode:

    (DMA2CONFIG) <= 0180h;

3. Program the DMA Channel 2 block size. Here, we set up for unlimited block size. Bit 15 is the enable bit for this channel.

    (DMA2BLKSIZE) <= 8000h;

4. The only registers that you need to program the DMA Channel 3 controller with is the address of the image memory: Since the shuttle direction is right-to-left, we must initialize these registers with the upper right-hand corner of the swath buffer.

    (DMA3CNTLO) <= 0046h;
    (DMACNT3HI) <= 0280h;

Program the BRB

1. First, load the Rotation Line Length Register (ROTLINE - 01FF8874h) with one less than the length of the image which would be 72 bytes - 1 = 71 = 47h.

    (ROTLINE) <= 0047h;

2. Program the BRBWARP (01FF8872h) register with the actual warp value. Here, the warp is 96 bytes = 60h.

    (BRBWARP) <= 0060h;

3. Program the Nozzle Number into ROTNN (01FF8870h) with the number of nozzles -1. In this case, the actual number of nozzles is 56. So, we need to program the ROTNN with 56 - 1 = 55 = 37h.

    (ROTNN) <= 0037h;

4. Program the ROTCTRL (01FF8060h) with the proper value. In this case, the shuttle direction is right-to-left.

    (ROTCTRL) <= 8001h;

Once the Rotation bit has been set, then Bit Rotation starts. The status bit (bit 15 of ROTCTRL) may then be polled to determine when the swath is completed with printing. On the next shuttle stroke, Only the address in channel 3 of the DMA controller needs to be changed as well as the shuttle direction bit within the ROTCTRL register. These should be the only parameters that need to change, once the initial parameters have been set up.

11.4.2 Swath Loader Requirements (Software Issues)

Due to the fact that the color nozzles are staggered as well as the fact that there is a physical separation VP of the color groups themselves, there are very special requirements imposed on the swath loader. In a situation where the mode is black only, then a minimum of two block buffers within image memory are required. This is due to the fact that while one block buffer is being printed, the other buffer may be simultaneously filled by the image loader.

Color mode is quite different from black-only print mode. In color mode, three separate colors are used: yellow, magenta, and cyan. The order of these colors may differ from one printhead to another. This is why the colors have been assigned simply as "Color1", "Color2", and "Color3" in Figure 11-1. On the first print swath of a page, Color3 is printed. On the next print block, Color3, and NG-VP + 1 nozzles of Color2 are printed. On the third pass, Color3, Color2, and NG-(2*VP) + 1 nozzles of Color1 are fired. In looking at this pattern, we see that two buffers are required to deal with printing Color3 (one block history required), three buffers are required in printing Color2 (two block history required), and four buffers are required in printing Color1 (three block history required).

Peerless PCL has a limitation that the color blocks that comprise the image data cannot be easily split up in order to take into account the VP parameter. The color data must be contiguous within the blocks. Due to this fact (along with the fact that the CPU may need to perform some preprocessing of the image prior to printing), the color buffers that are provided for use by the Bit Rotation Block are prepared by the CPU in the background. The figure shows the typical preparation of these buffers in the background by the CPU.

> *Note:*   *Before scanning starts, the two prepared color buffers must be zero-filled by the CPU for reasons of top-of-page boundary conditions. If, at the end of the printed page, there are fewer lines than the number in the prepared buffers, then the remainder of the prepared buffers must be zero-filled to account for bottom-of-page boundary conditions.*

Note: Each Buffer is NG Lines deep

Note: This example shows what these buffers should look like if printing swaths 3 and 4 starting with swath 1.

Data Buffers Dynamically Allocated by CPU

Two buffers prepared in background by CPU

Color3, Buffer 0

Color3, Buffer 1

Color 3, Buffer 0, lines 0 to NG

Color 2, Buffer 0 Lines NG-VP+1 to NG

Color 2, Buffer 1 Lines 0 to NG-VP

lines 0 to (2*VP) - 1 are zero-filled.

Color 1, Buffer 0, lines 0 to NG - (2*VP) .

for swath 3

Color2, Buffer 0

Color2, Buffer 1

Color 2, Buffer 2

for swath 4

Color 3, Buffer 1, lines 0 to NG

Color 2, Buffer 1, lines NG - (VP+1) to NG.

Color 2, Buffer 2, lines 0 to NG-VP.

Color1, Buffer 0

Color1, Buffer 1

Color1, Buffer 2

Color1, Buffer 3

Note: NG is one less than the physical number of nozzles in the color group. VP is the vertical distance between nozzle groups

Color 1, Buffer 0, lines NG - (2*VP) + 1 to NG.

Color 1, Buffer 1, lines 0 to NG - (2*VP) .

Figure 11-6. CPU Background Print Data Preparation

During black-only printing, the two buffers that are prepared for color printing simply become the two buffers that are used for the black-only print mode. Each of these buffers will contain as many lines as there are nozzles on the printhead.

Image Format

The data that is presented to the Bit Rotation Block is always in little-endian format. The data within the system is in 2 separate formats: big-endian and little-endian. However, if the data in image memory is in big-endian format, then the SIU will translate that data to a little-endian format during the DMA operation with the endian conversion bit enabled, to ensure that the Bit Rotation Block only needs to deal with little-endian formatted data. The little endian format that the Bit Rotation Block expects to see is shown in the following illustration:
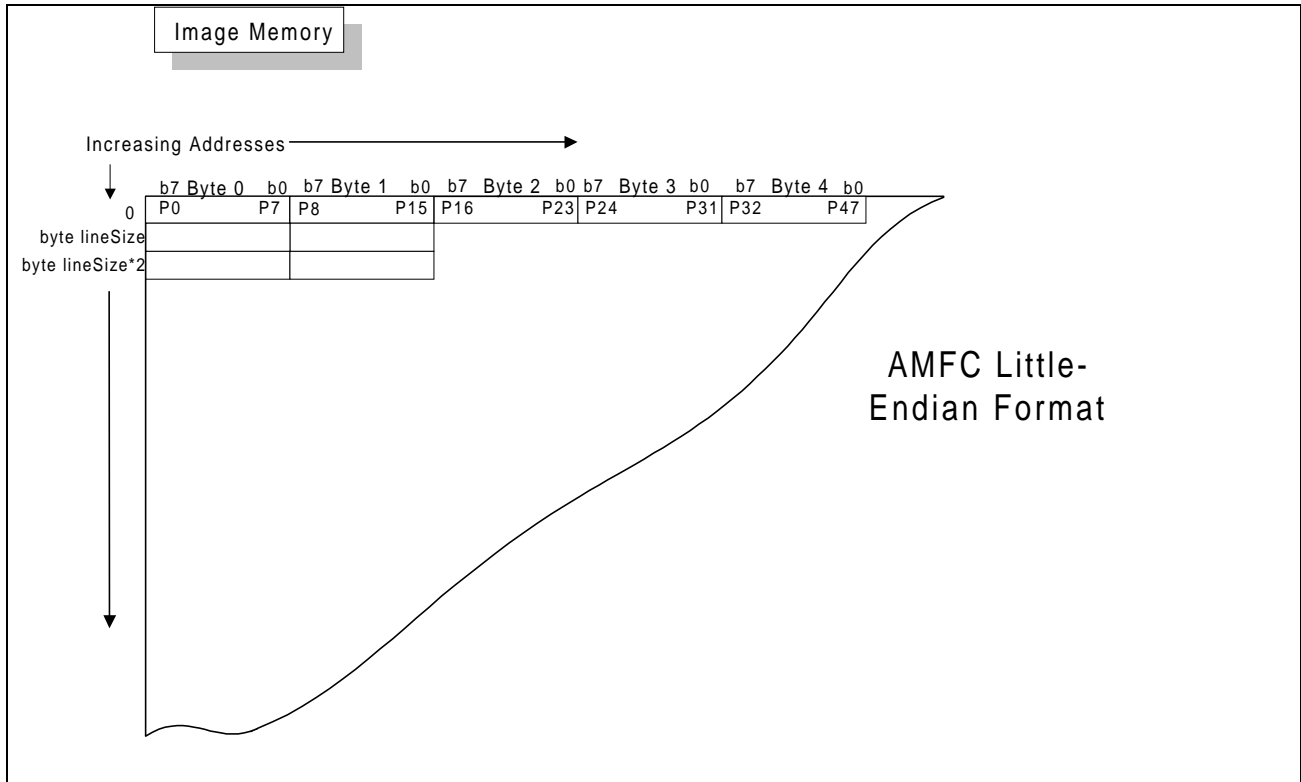
Figure 11-7. MFC2000 Little-Endian Format

This page is intentionally blank

12. Printer and Scanner Stepper Motor Interface

12.1 Vertical Print Stepper Motor Interface

12.1.1 Vertical Print Stepper Motor Control Description

The stepper motor can be controlled by this block to run at constant speed, increasing speed (acceleration), or decreasing speed (deceleration).

The control logic is based on a loadable step timer and a step timer register. A pulse and interrupt are generated when the step timer expires (timed-out) and at the same time, the content of the step timer register is automatically loaded into the step timer. The pulse is used to change the phase pattern to the next one in the Motor Pattern Control Sub-block and the interrupt is used to inform CPU to load the new step timer value to the step timer register if necessary for the step after the next step (See Figure 12-1).
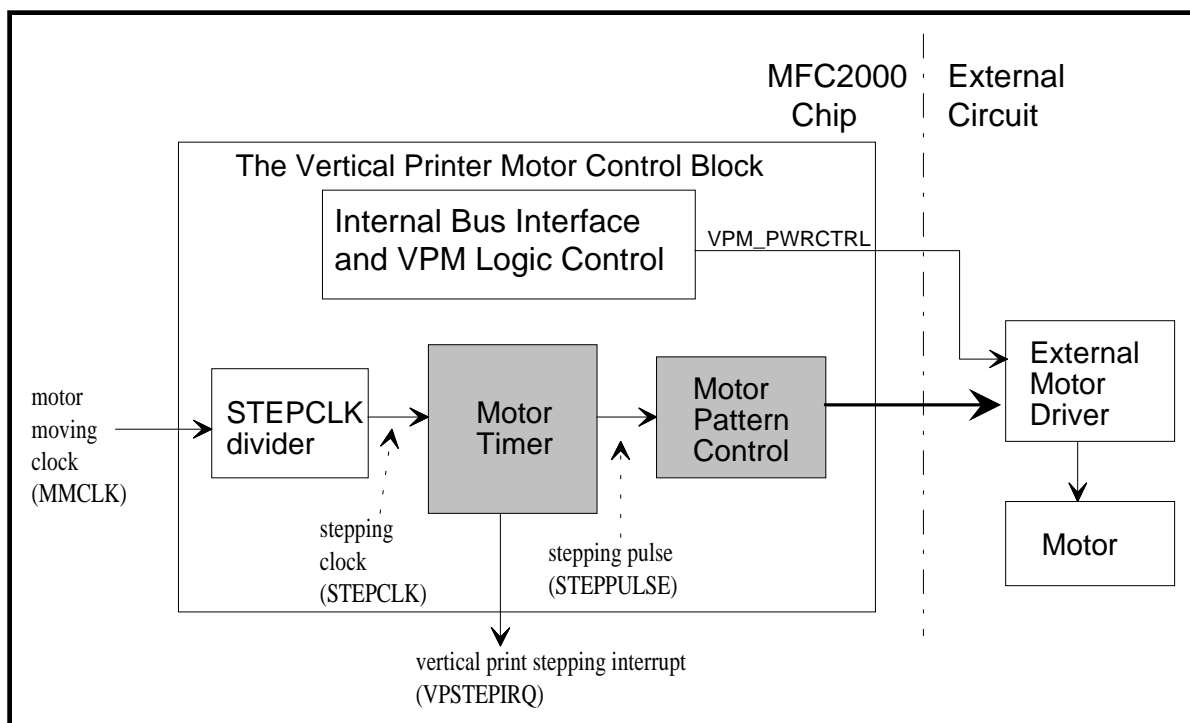
Figure 12-1. Vertical Printer Motor Control Block Diagram

The printer motor control lines are the four pins PM[3:0]/GPO[3:0]. These pins are selected as printer motor control lines when bit 0 in the GPIOconfig register is set to 0 (default), and are selected as GPO's when this bit is set to 1. The printer pattern or GPO data is output on pins PM[3:0]/GPO[3:0] from bits [3:0] (VPMF[3:0]) of the VPMPattern register.

When selected as GPO's, data written by CPU to the VPMPattern register is immediately output on the GPO pins and output data are not changed unless CPU writes another new data to the VPMPattern register.

When selected as printer motor control lines, data is not allowed to be written to the VPMPattern register no matter whether the motor stepping is disabled or enabled.

At the beginning of the motor moving process, CPU needs to write the stepping mode and the stepping direction to the VPStepCtrl register, the initial vertical print motor pattern to the VPMPattern register, and the 1st step timer value to the VPStepTimer register (used to define the time interval between enabling stepping and the 1st step).

> ***Caution:*** *The vertical print motor pattern register (VPMPattern register) can only be written when it is used as the GPO function. In other words, bit 0 of the GPIOConfig register is set to 1.*

Then, CPU sets bit 0 of the GPIOConfig register to 0 and enables the vertical print motor stepping. The content (the 1st step timer value) of the step timer register is automatically loaded into the step timer and at the same time an interrupt is generated (the step pulse is blocked at this time). CPU writes the 2nd step timer value into the step timer register in the stepping interrupt routine. When the step timer expires (timed-out), A pulse is generated to change the phase pattern to the next one (the 1st step) and the content (the 2nd step timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated to inform CPU to load the 3rd step timer value to the step timer register.

At the near end of the motor stepping, a pulse is generated to change the phase pattern to the next one (the 'n-2' step, n: the last step) and the content of the step timer register (the time interval between the 'n-2' and 'n-1 steps) is automatically loaded into the step timer and an interrupt is generated to inform CPU to load the last step timer value to the step timer register when the step timer expires (timed-out). When the step timer expires again (timed-out), A pulse is generated to change the phase pattern to the next one (the 'n-1' step) and the content (the last step timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated to inform CPU to load the large dummy timer value to the step timer register. Finally, a pulse is generated to change the phase pattern to the next one (the last step) and the content (the large dummy timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated to inform CPU to disable the motor stepping. The step timer is cleared to 0 and the stepping pulse is blocked immediately after CPU disables the motor stepping.

The step enable bit in the VPStepCtrl register only controls if the motor pattern is allowed to change to the next one. The motor pattern in the VPMPattern register always goes out. CPU has the responsibility to write value '00h' to the VPMPattern register or set bit 4 of the VPStepCtrl register to a proper value according to the external power control circuit for not driving the vertical print motor. Bit 4 value of the VPStepCtrl register directly goes out of the MFC2000 chip through the GPIO[13]/SASTXD/PMPWRCTRL pin.

> ***Caution:*** *The GPIO[13]/SASTXD/PMPWRCTRL pin can only be used as the PMPWRCTRL pin when bit11 of the GPIOConfig1 register is '0' and bit 12 of the GPIOConfig1 register is '1'.*

The time interval between two adjacent stepping pulses (for example, t1 and t2) is determined by the timer value. The min. time difference between two adjacent stepping pulses (It is called the timer resolution, for example, $\Delta t = | t2-t1 |$.) is determined by the frequency of the stepping clock (See Figure 12-2).
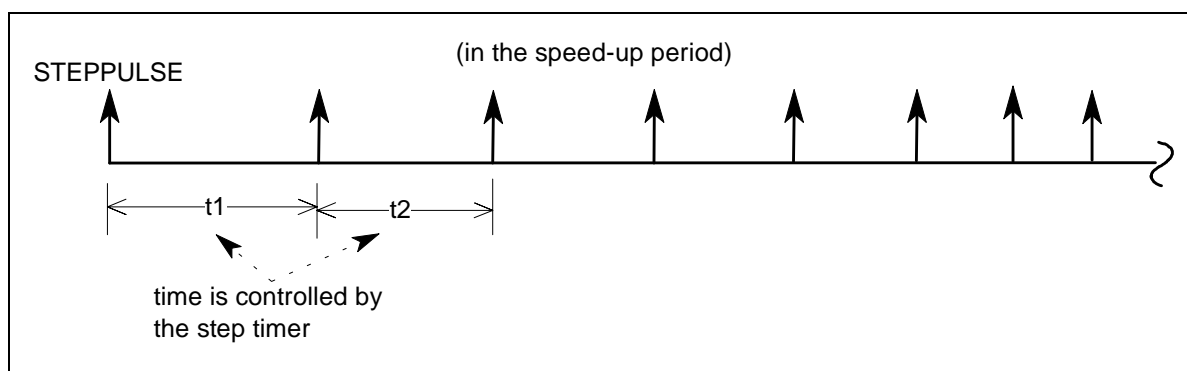
Figure 12-2. Stepping Timing

The change sequence of the phase pattern in the Motor Pattern Control Sub-block is as follows: after the CPU gives the initial phase pattern (any column in the following tables), the phase pattern will automatically change to the next one after getting the time-out pulse of the timer (the stepping pulse) according to the clockwise (CW) or counterclockwise (CCW) rotation in the following tables.

Table 12-1. Full Step/Single Phase Excitation

-------------------------> CW rotation

	Step pattern 1	Step pattern 2	Step pattern 3	Step pattern 4	Step pattern 1
VPM [0](phase A)	1	0	0	0	1
VPM [1](phase B)	0	1	0	0	0
VPM [2](phase C)	0	0	1	0	0
VPM [3](phase D)	0	0	0	1	0

<------------------------ CCW rotation

Table 12-2. Full Step/Two Phase Excitation

-------------------------> CW rotation

	Step pattern 1	Step pattern 2	Step pattern 3	Step pattern 4	Step pattern 1
VPM [0](phase A)	1	0	0	1	1
VPM [1](phase B)	1	1	0	0	1
VPM [2](phase C)	0	1	1	0	0
VPM [3](phase D)	0	0	1	1	0

<------------------------ CCW rotation

Table 12-3. Half-Step Excitation

------------------------> CW rotation

	Step FP1	Step HP2	Step FP3	Step HP4	Step FP5	Step HP6	Step FP7	Step HP8
VPM [0]	1	0	0	0	0	0	1	1
VPM [1]	1	1	1	0	0	0	0	0
VPM [2]	0	0	1	1	1	0	0	0
VPM [3]	0	0	0	0	1	1	1	0

<------------------------ CCW rotation

> **Note:**   *FPx: full step pattern and x: 1-4, HPx: half step pattern and x:1-4*

Programmability:

- The frequency of STEPCLK is programmable (around 1.25 MHz—39 KHz) if the ICLK (internal) frequency is 10 MHz.
- The step timer value is programmable (around 1us—400ms) if the ICLK (internal) frequency is 10 MHz.
- The Motor moving direction (clockwise or counterclockwise) and the stepping mode (full step or half step) is programmable.
- The motor phase pattern is programmable.
- Control bits for clearing the stepping interrupt and enabling the stepping logic are included.
- A control bit for the vertical print motor power on/off.

12.1.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Vertical Printer Motor Pattern Register *(VPMPattern)* **01FF805D**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Vertical Printer Motor Pattern Register *(VPMPattern)* **01FF805C**	VPMH[3]	VPMH[2]	VPMH[1]	VPMH[0]	VPMF[3]	VPMF[2]	VPMF[1]	VPMF[0]	Rst. Value 00h Read Value 00h

For the full step excitation, the phase pattern VPM[3:0] (See Table 12-1 and Table 12-2) needs to be put into VPMF[3:0] of this VPMPattern register. Bit[7:4] of this register are 'don't care'.

For the half step excitation, two adjacent phase patterns (See Table 12-3) need to be put into the VPMPattern register: VPM[3:0] at FP[x] column needs to be put into VPMF[3:0] and VPM[3:0] at HP[x] column needs to be put into VPMH[3:0]. The current output at PM[3:0] is VPMF[3:0].

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Clock Register *(VPStepClk)* **01FF8853**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Step Clock Register *(VPStepClk)* **01FF8852**	(Not Used)	(Not Used)	(Not Used)	bit 4 of the step clock divider	bit 3 of the step clock divider	bit 2 of the step clock divider	bit 1 of the step clock divider	bit 0 of the step clock divider	Rst. Value xxx00000b Read Value 00h

- STEPCLK = ICLK/(8 * (n+1)), n (bit[4:0] of the step clock divider): from 0 to 31

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Control Register *(VPStepCtrl)* **01FF8059**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Step Control Register *(VPStepCtrl)* **01FF8058**	(Not Used)	(Not Used)	(Not Used)	Vertical Print Motor Power Control	Stepping Mode 0: full step 1: half step	Motor Direction 0: CW 1: CCW	Step Enable 0: disable 1: enable	STEPIRQ Clear 1:clear	Rst. Value xxx00000b Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Timer Register *(VPStepTimer)* **01FF805B**	(Not Used)	(Not Used)	bit 13 of the step timer	bit 12 of the step timer	bit 11 of the step timer	bit 10 of the step timer	bit 9 of the step timer	bit 8 of the step timer	Rst. Value xx000000b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Step Timer Register *(VPStepTimer)* **01FF805A**	bit 7 of the step timer	bit 6 of the step timer	bit 5 of the step timer	bit 4 of the step timer	bit 3 of the step timer	bit 2 of the step timer	bit 1 of the step timer	bit 0 of the step timer	Rst. Value 00h Read Value 00h

- The Time interval between two adjacent step pulses = (n+1)/STEPCLK, n (bit[13:0] of the step timer): from 0 to 16383

12.2 Scanner Stepper Motor Interface

12.2.1 Scanner Stepper Motor Control Description

The stepper motor can be controlled by this block to run at constant speed, increasing speed (acceleration), or decreasing speed (deceleration).

The Motor Timer Sub-block contains two sets of loadable timers and timer registers. One is used for the step control logic and the other is used for the delay control logic.

The step control logic is based on a loadable step timer and a step timer register. A pulse and interrupt are generated when the step timer expires (timed-out) and at the same time, the content of the step timer register is automatically loaded into the step timer. The pulse is used to change the phase pattern to the next one in the Motor Pattern Control Sub-block and the interrupt is used to inform CPU to load the new step timer value to the timer register if necessary for the step after the next step (See Figure 12-3).
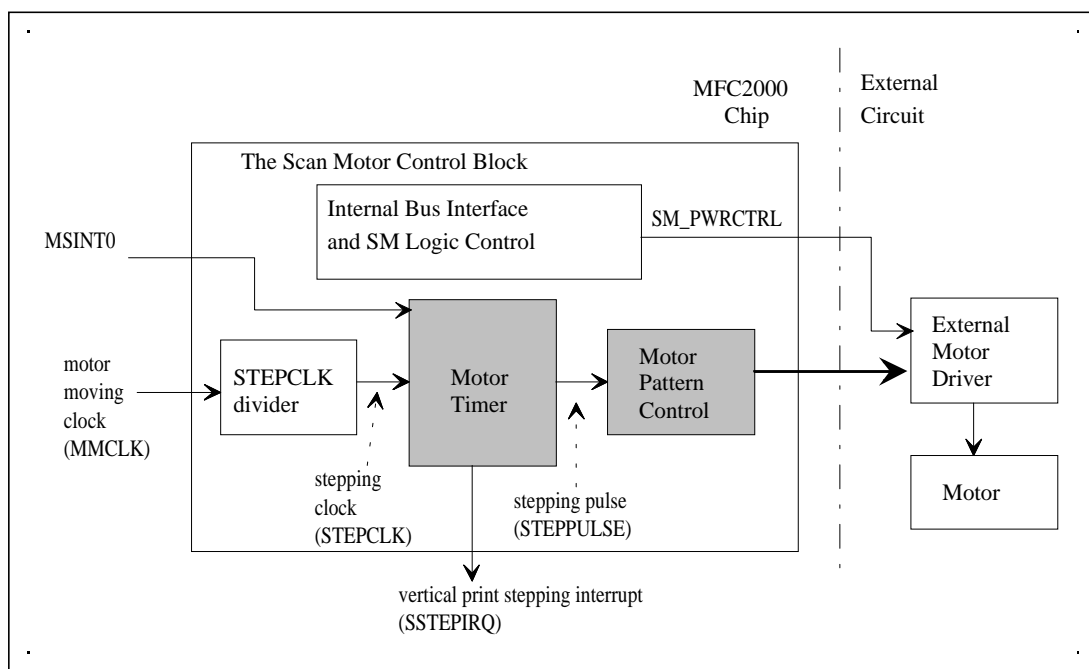
Figure 12-3. Scan Motor Control Diagram

The scan motor stepping needs to synchronize to the scan cycle during scanning. Signal MSINT0 is used to indicate the beginning of the scan cycle. MSINT0 synchronization can be enabled or disabled through the register setting. This is the main difference between the Vertical Print Motor Control block and the Scan Motor Control block.

The delay control logic is based on a loadable delay timer and a delay timer register. The MSINT0 and the delay timer control when the Motor Timer Sub-block sends out the first stepping pulse after CPU enables the scan motor stepping. Once, CPU enables the scan motor stepping, the Motor Timer Sub-block waits for the MSINT0 pulse to load the delay timer value from the delay timer register and activate the delay timer if the MSINT0 synchronization is enabled. Otherwise, the Motor Timer Sub-block loads the new timer value from the delay timer register and activates the delay timer immediately. The first stepping pulse will be sent to the Motor Pattern Control Sub-block when the delay timer expires (timed-out). The delay timer only works once after CPU enables the scan motor stepping for the first step.

The scan motor control lines are the four pins SM[3:0]/GPO[7:4]. These pins are selected as scanner motor control lines when bit 1 in the GPIOconfig register is set to 0 (default), and are selected as GPO's when this bit is set to 1. The scanner motor pattern or GPO data is output on pins SM[3:0]/GPO[7:4] from bits [3:0] of the SMPattern register.

When selected as GPO's, data written by CPU to the SMPattern register is immediately output on the GPO pins and output data are not changed unless CPU writes another new data to the SMPattern register.

When selected as scanner motor control lines, data is not allowed to be written to the SMPattern register no matter whether the motor stepping is disabled or enabled.

> *Caution: The vertical print motor pattern register (SMPattern register) can only be written when it is used as the GPO function. In other words, bit 1 of the GPIOConfig register is set to 1.*

At the beginning of the motor moving process, CPU needs to write the stepping mode and the stepping direction to the SStepCtrl register, the initial scan motor pattern to the SMPattern register, the delay timer value to the SDelayTimer register, and the 1st step timer value to the SStepTimer register (used to define the time interval between the 1st and 2nd steps).

Then, CPU sets bit 1 of the GPIOConfig register to 0 and enables the scan motor stepping. If the MSINT0 synchronization is disabled, the content of the delay timer register is immediately loaded into the step timer. If the MSINT0 synchronization is enabled, the delay timer loading will be delayed until the beginning of the scan (seeing the MSINT0 pulse). After Motor Timer Sub-block loads the new timer value from the delay timer register, It activates the delay timer immediately. The first stepping pulse will be sent to the Motor Pattern Control Sub-block when the delay timer expires (timed-out).

At the same time, the content (the 1st step timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated. CPU writes the 2nd step timer value into the step timer register in the stepping interrupt routine. When the step timer expires (timed-out), A pulse is generated to change the phase pattern to the next one (the 2nd step) and the content (the 2nd step timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated to inform CPU to load the 3rd step timer value to the step timer register.

At the end of the motor stepping, a pulse is generated to change the phase pattern to the next one (the 'n-2' step, n: the last step) and the content of the step timer register (the time interval between the 'n-2' and 'n-1 steps) is automatically loaded into the step timer and an interrupt is generated to inform CPU to load the last step timer value to the step timer register when the step timer expires (timed-out). When the step timer expires again (timed-out), A pulse is generated to change the phase pattern to the next one (the 'n-1' step) and the content (the last step timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated an interrupt is generated to inform CPU to load the large dummy timer value to the step timer register. Finally, a pulse is generated to change the phase pattern to the next one (the last step) and the content (the large dummy timer value) of the step timer register is automatically loaded into the step timer and an interrupt is generated to inform CPU to disable the motor stepping. The step and delay timers are cleared to 0 and the stepping pulse is blocked immediately after CPU disables the motor stepping.

The step enable bit in the SStepCtrl register only controls if the motor pattern is allowed to change to the next one. The motor pattern in the SMPattern register always goes out. CPU has the responsibility to write value '00h' to the SMPattern register or set the bit 5 of the SStepCtrl register to a proper value according to the external motor power control circuit for not driving the scan motor. Bit 5 value of the SStepCtrl register directly goes out of the MFC2000 chip through the GPIO[12]/SASCLK/SMPWRCTRL pin.

> ***Caution:***   *The GPIO[12]/SASCLK/SMPWRCTRL pin can only be used as the SMPWRCTRL pin when bit 9 of the GPIOConfig1 register is '0' and bit 10 of the GPIOConfig1 register is '1'.*

The step time interval between two adjacent stepping pulses (for example, t1 and t2) is determined by the timer value. The min. time difference between two adjacent stepping pulses (It is called the timer resolution, for example, $\Delta t = | t2-t1 |$.) is determined by the frequency of the stepping clock (See Figure 12-4).
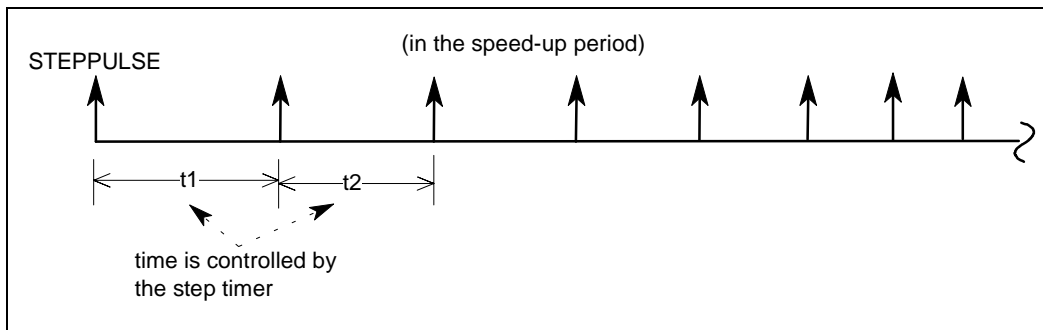
Figure 12-4. Stepping Timing

The change sequence of the phase pattern in the Motor Pattern Control Sub-block is as follows: after the CPU gives the initial phase pattern (any column in the following tables), the phase pattern will automatically change to the next one after getting the time-out pulse of the timer (the stepping pulse) according to the clockwise (CW) or counterclockwise (CCW) rotation in the following tables.

Table 12-4. Full Step/Single Phase Excitation

------------------------> CW rotation

	Step pattern 1	Step pattern 2	Step pattern 3	Step pattern 4	Step pattern 1
SM [0](phase A)	1	0	0	0	1
SM [1](phase B)	0	1	0	0	0
SM [2](phase C)	0	0	1	0	0
SM [3](phase D)	0	0	0	1	0

<------------------------ CCW rotation

Table 12-5. Full Step/Two Phase Excitation

------------------------> CW rotation

	Step pattern 1	Step pattern 2	Step pattern 3	Step pattern 4	Step pattern 1
SM [0](phase A)	1	0	0	1	1
SM [1](phase B)	1	1	0	0	1
SM [2](phase C)	0	1	1	0	0
SM [3](phase D)	0	0	1	1	0

<------------------------ CCW rotation

Table 12-6. Half-Step Excitation

------------------------> CW rotation

	Step FP1	Step HP2	Step FP3	Step HP4	Step FP5	Step HP6	Step FP7	Step HP8
SM [0]	1	0	0	0	0	0	1	1
SM [1]	1	1	1	0	0	0	0	0
SM [2]	0	0	1	1	1	0	0	0
SM [3]	0	0	0	0	1	1	1	0

<------------------------ CCW rotation

Note: *FPx: full step pattern and x: 1-4, HPx: half step pattern and x:1-4*

Programmability:

- The frequency of STEPCLK is programmable (around 1.25 MHz—39 KHz) if the ICLK (internal) frequency is 10 MHz.
- The step timer value is programmable (around 1us—400ms) if the ICLK (internal) frequency is 10 MHz.
- The delay timer value is programmable (around 1us—400ms).
- The Motor moving direction (clockwise or counterclockwise) and the stepping mode (full step or half step) is programmable.
- The motor phase pattern is programmable.
- Control bits for clearing the stepping interrupt and enabling the stepping logic are included.
- The MSINT0 synchronization can be enabled and disabled.
- A control bit for the scan motor power on/off.

12.2.2 Scan Motor Current Control

The current control logic is added to this block. Two current control output signals (SMI[1:0]) are needed to control 4 different current modes. Three different current modes can be used within the time period of one step (See Figure 12-5). T1 and T2 are programmable in one step period. The current mode in each current period of one step period is programmable. CPU can load T1 and T2 values and current modes of three current periods within one step for the next step at the same time when CPU loads the next step timer value by using the scan step interrupt (SSTEPIRQ). In other words, T1, T2, and current modes are double buffered.

SMI[0] output value is 1 and SMI[1] output value is 1 after reset and when SM[3:0] pins are used as GPO's. The default current mode is also programmable when SM[3:0] pins are used as motor control pins and motor stepping is disabled.
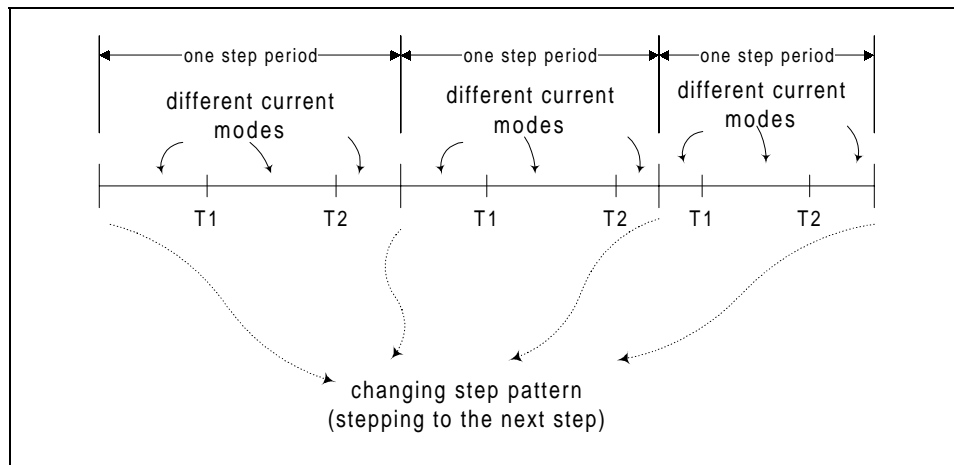
Figure 12-5: Current Control Diagram

Two current control signals (SMI1 and SMI0) share same pins with PM[3]/GPO[3] and PM[2]/GPO[2]. There is a SMI[1:0] output select bit in the SStepCtrl Register to select which signals output on PM[3:2] pins.

12.2.3 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Scanner Motor Pattern Register *(SMPattern)* **01FF8057**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Scanner Motor Pattern Register *(SMPattern)* **01FF8056**	SMH[3]	SMH[2]	SMH[1]	SMH[0]	SMF[3]	SMF[2]	SMF[1]	SMF[0]	Rst. Value 00h Read Value 00h

For the full step excitation, the phase pattern SM[3:0] (See Table 12-4 and Table 12-5) needs to be put into SMF[3:0] of this SMPattern register. Bit[7:4] of this register are 'don't care'.

For the half step excitation, two adjacent phase patterns (See Table 12-6) need to be put into the SMPattern register: SM[3:0] at FP[x] column needs to be put into SMF[3:0] and SM[3:0] at HP[x] column needs to be put into SMH[7:4]. The current output at PM[3:0] is SMF[3:0].

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Clock Register *(SStepClk)* **01FF8851**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Step Clock Register *(SStepClk)* **01FF8850**	(Not Used)	(Not Used)	(Not Used)	bit 4 of the step clock divider	bit 3 of the step clock divider	bit 2 of the step clock divider	bit 1 of the step clock divider	bit 0 of the step clock divider	Rst. Value xxx00000b Read Value 00h

- STEPCLK = ICLK/(8 * (n+1)), n (bit[4:0] of the step clock divider): from 0 to 31

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Control Register *(SStepCtrl)* **01FF8051**	I31	I30	I21	I20	I11	I10	DI1	DI0	Rst. Value FFh Read Value FFh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Step Control Register *(SStepCtrl)* **01FF8050**	(Not Used)	SMI Output Select	Scan Motor Power Control	MSINT0 Sync 0: disabled 1: enabled	Stepping Mode 0: full step 1: half step	Motor Direction 0: CW 1: CCW	Step Enable 0: disable 1: enable	STEPIRQ Clear 1:clear	Rst. Value x0000000b Read Value 00h

Bits 15-14	The third current control values go out to the scan current control pins (SMI1 and SMI0) within one step time period.
Bits 13-12	The second current control values go out to the scan current control pins (SMI1 and SMI0) within one step time period.
Bits 11-10	The first current control values go out to the scan current control pins (SMI1 and SMI0) within one step time period.
Bits 9-8	The default current control values go out to the scan current control pins (SMI1 and SMI0) when the motor stepping is disabled and SM[3:0] is used for motor control.
Bits 6	Control which signal output to PM[3:2] pins (SMI[1:0] output signals or PM[3]/GPO[3] and PM[2]/GPO[2] output signals). 0: Output PM[3]/GPO[3] and PM[2]/GPO[2] signals, 1: Output SMI[1:0] signals.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Timer Register *(SStepTimer)* **01FF8053**	(Not Used)	(Not Used)	bit 13 of the step timer	bit 12 of the step timer	bit 11 of the step timer	bit 10 of the step timer	bit 9 of the step timer	bit 8 of the step timer	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Step Timer Register *(SStepTimer)* **01FF8052**	bit 7 of the step timer	bit 6 of the step timer	bit 5 of the step timer	bit 4 of the step timer	bit 3 of the step timer	bit 2 of the step timer	bit 1 of the step timer	bit 0 of the step timer	Rst. Value 00h Read Value 00h

- The Time interval between two adjacent step pulses = (n+1)/STEPCLK, n (bit[13:0] of the step timer): from 0 to 16383

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Step Delay Timer Register *(SSDelayTimer)* **01FF8055**	(Not Used)	(Not Used)	bit 13 of the delay timer	bit 12 of the delay timer	bit 11 of the delay timer	bit 10 of the delay timer	bit 9 of the delay timer	bit 8 of the delay timer	Rst. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Step Delay Timer Register *(SSDelayTimer)* **01FF8054**	bit 7 of the delay timer	bit 6 of the delay timer	bit 5 of the delay timer	bit 4 of the delay timer	bit 3 of the delay timer	bit 2 of the delay timer	bit 1 of the delay timer	bit 0 of the delay timer	Rst. Value 00h Read Value 00h

- The delay time interval before stepping = (n+1)/STEPCLK, n (bit[13:0] of the step delay timer): from 0 to 16383

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Current Timer 1 Register *(SSCurTimer1)* **01FF804D**	(Not Used)	(Not Used)	bit 13 of the current timer	bit 12 of the current timer	bit 11 of the current timer	bit 10 of the current timer	bit 9 of the current timer	bit 8 of the current timer	Rst. Value xx000000b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Current Timer 1 Register *(SSCurTimer1)* **01FF804C**	bit 7 of the current timer	bit 6 of the current timer	bit 5 of the current timer	bit 4 of the current timer	bit 3 of the current timer	bit 2 of the current timer	bit 1 of the current timer	bit 0 of the current timer	Rst. Value 00h Read Value 00h

- This register defines when to change the first driving current to the second driving current within one step time period. The current timer 1 value in this register must be less than the step timer value set in the SStepTimer register for the same step and greater than 0. Otherwise, the current will not be changed. Once the step timer counts down to the current timer 1 value, the current control signals changes to I20 and I21 from I10 and I11 in the SStepCtrl register. I10 and I11 are the initial values of the current control signals within a step.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Current Timer 2 Register *(SSCurTimer2)* **01FF804F**	(Not Used)	(Not Used)	bit 13 of the current timer	bit 12 of the current timer	bit 11 of the current timer	bit 10 of the current timer	bit 9 of the current timer	bit 8 of the current timer	Rst. Value xx000000b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Current Timer 2 Register *(SSCurTimer2)* **01FF804E**	bit 7 of the current timer	bit 6 of the current timer	bit 5 of the current timer	bit 4 of the current timer	bit 3 of the current timer	bit 2 of the current timer	bit 1 of the current timer	bit 0 of the current timer	Rst. Value 00h Read Value 00h

- This register defines when to change the second driving current to the third driving current within one step time period. The current timer 2 value in this register must be less than the current timer 1 value set in the SSCurTimer1 register for the same step and greater than 0. Otherwise, the current will not be changed. Once the step timer counts down to the current timer 2 value, the current control signals changes to I30 and I31 from I20 and I21 in the SStepCtrl register.

This page is intentionally blank

13. General Purpose Inputs/Outputs (GPIO)

13.1 GPIO Signals

The AMFPC provides 31 GPIO lines (GPIO[30:0]) typically used for I/O control, each of which is multiplexed with other signals. The direction of each GPIO line (input or output) is programmable using the hardware registers shown in Section 13.3.

GPIO0	GPIO0 is multiplexed with the flash memory write enable signal (FWRn) for the NAND-type flash memory, this GPIO0 pin is also multiplexed with W_Rn.
	This pin is selected as a GPIO when bit 2 of the GPIOConfig1 register is set to 0 and Emulator signals are not enabled, in this case, the direction of this pin is controlled by bit 0 of the GPIODir register. If GPIODir[0] is set to one, this pin will ouput the value of the bit 0 from GPIOData register, and, if GPIODir[0] is set to zero, the value of the GPIO0 can be read back by reading GPIOdata[0].
	This pin is selected as FWRn when the GPIOConfig1 bit is set to 1, in this case, this pin becomes an output pin regardless of the value in the GPIODir[0].
	This pin is selected as W_Rn when EMUSIG_EN is set to 1, refer to Testmgr section for EMUSIG_EN signal definition.
GPIO1	GPIO1 is multiplexed with the flash memory read enable signal (FRDn) for the NAND-type flash memory, this GPIO1 pin is also multiplexed with XAKn.
	This pin is selected as a GPIO when bit 2 of the GPIOConfig1register is set to 0 and, in this case, the direction of this pin is controlled by bit 1 of the GPIODir register. If GPIODir[1] is set to one, this pin will ouput the value of the bit 1 from GPIOData register, and, if GPIODir[1] is set to zero, the value of the GPIO1 can be read back by reading GPIOdata[1].
	This pin is selected as FWRn when the GPIOConfig1 bit is set to 1, in this case, this pin becomes an output pin regardless of the value in the GPIODir[1].
	This pin is selecetd as XAKn when EMUSIG_EN is set to 1, refer to Testmgr section for EMUSIG_EN signal definition.
GPIO2	GPIO2 is multiplexed with SSCLK2, this pin is also used for the DMA request input from DMA channel 1 (DMAREQ1). If this pin is used as DMAREQ1, the direction of this pin needs to be set as input.
	If SSCLK2 function is used, the bit 0 of GPIOConfig2 register needs to be set to 1. If this pin is used as SSCLK2, the GPIO2 direction and data setting for this pin is no longer valid.
	When bit 16 of GPIOCofig register is set to 0, this pin can be used as GPIO2, in this case, the direction of this pin is controlled by bit 2 of the GPIODir register. The GPIO2 input/output value is controlled by bit 2 of the GPIOData register.

GPIO3	GPIO3 is multiplexed with the DMA acknowledge of the DMA channel 1 (DMAACK1), it is also used as SSRXD2 input. If this pin is used as SSRXD2, the direction of this pin needs to be set as input and the DMAACK1 function must be disabled.

If this pin is used as DMAACK1, the GPIO direction and data setting for this pin is no longer valid. This function is enabled by setting EXTDMA1SEL to 1.

If this pin is used as GPIO, the direction of this pin is controlled by bit 3 of the GPIODir register. The GPIO3 input/output value is controlled by bit 3 of the GPIOData register. |
| GPIO[7:4] | GPIO[7:4] are multiplexed with general chip selects on pins GPIO7/CS5n, GPIO6/CS4n, GPIO5/CS3n, and GPIO4/CS2n. These pins are selected as GPIO when the corresponding bits in the GPIOConfig1 register are set to 0. The direction of these pins is controlled by the bits 7-4 in the GPIODir register. The GPIO[7:4] input/output values are controlled by the bits 7-4, respectively, of the GPIOData register.

In addition, the GPIO[5]/CS3n pin is also multiplexed with the PWM3 signal. See the register description in the PWM section for settings. GPIO6 can also used as input pin for EV_CLK and EADC_D[3], if this function is selected, the direction control for GPIO6 must be set to 0. |
| GPIO8 | GPIO8 is multiplexed with SSSTA1 and SC_CLK1/2B, this pin is also used as the interrupt request 11 (IRQ11) input. If this pin is used as IRQ11, the direction of this pin needs to be set as input. If this pin is not used as IRQ11, the interrupt channel 11 needs to be disabled by setting the IRQEnable register in the Interrupt Controller.

If SSSTA1 function is selected, the bit 15 of GPIOConfig1 register must be set to 1. To enable SC_CLK1/2B function, bit 15 of GPIOConfig1 must be set to 0 and bit 18 of this register set to 1. Only when both bit15 and bit2 of the GPIOConfig2 register are set to 0, the GPIO8 pin is enabled as GPIO.

When GPIO function is enabled, the direction of this pin is controlled by bit 8 of the GPIODir register. The GPIO8 input/output value is controlled by bit 8 of the GPIOData register. |
| GPIO9 | GPIO9 is also used as the interrupt request 13 (IRQ13n) input or EADC_D[2] input. If this pin is used as IRQ13n or EADC_D[2], the direction of this pin needs to be set as input. If this pin is not used as IRQ13n, the interrupt channel 13 needs to be disabled by setting the IRQEnable register in the Interrupt Controller.

The direction of this pin is controlled by bit 9 of the GPIODir register. The GPIO9 input/output value is controlled by bit 9 of the GPIOData register. |

GPIO10   GPIO10 is multiplexed with the PWM[4] signal, this pin is also used for P80_PB3 as input. When P80_PB3 function is used, the direction of this pin needs to be set as input. This pin is selected as a GPIO when bit 7 of the GPIOConfig1 register is set to 0. The direction of this pin is controlled by bit 10 of the GPIODir register, and the input/output value is controlled by bit 10 of the GPIOData register.

The PWM[4] function is enabled by setting bit 7 of the GPIOConfig1 register to 1, additional control is required to use PWM[4], refer to PWM section for detail.

GPIO11   GPIO11 is also used for the Calling Party Control Input (CPCIN) pin and multiplexed with the ALTTONE output signal on pin GPIO11/CPCIN/ALTTONE. This pin is selected as a GPIO by setting bit 8 of the GPIOConfig1 register to 0. The direction of this pin is controlled by bit 11 in the GPIODir register, and the input/output value is controlled by bit 11 in the GPIOData register. The ALTTONE output signal is selected by setting bit 8 of the GPIOConfig1 register to 1 and setting bit 11 of the GPIODir register to 1. If this pin is used as the CPCIN signal input pin, the bit 11 of the GPIODir register is set to 0.

In addition, the GPIO11/CPCIN/ALTTONE pin is also multiplexed with the PWM0 signal. See the register description in the PWM section for settings.

GPIO[14:12]   GPIO[14:12] are multiplexed with the scan/print motor power control output signals, the RINGER output signal, and the SASIF signals on pin GPIO14/SASRXD/RINGER, GPIO13/SASTXD/PMPWRCTRL, and GPIO12/SASCLK/SMPWRCTRL. These pins are selected as a GPIO by setting bit[12:9] in the GPIOConfig1 register to zero. The direction of these pins are controlled by bit[14:12] of the GPIODir register, and the input/output value is controlled by bit[14:12] of the GPIOData register. These pins are selected as the SASIF signals or RINGER/PMPWRCTRL/SMPWRCTRL signals by setting bit[14:9] in the GPIOConfig1 register (refer to GPIOConfig1 register description). Firmware needs to program GPIO[14:12] pins to use SASRXD, SASTXD, and SASCLK for the sync mode. If the async mode of SASIF is used, firmware only needs to program GPIO[14:13] pins to use SASRXD and SASTXD. GPIO[12] can be still used as GPIO.

GPIO15   GPIO15 is multiplexed with SC_CLK1/2C signals, this pin is also used as IRQ16 input. If this pin is used as IRQ16, the direction of this pin needs to be set as input. If this pin is not used as IRQ16, the interrupt channel 16 needs to be disabled by setting the IRQEnable register in the Interrupt Controller.

If SC_CLK1/2C function is selected, the bit 3 of GPIOConfig2 register must be set to 1. To enable GPIO15 function, bit 3 of GPIOConfig2 must be set to 0.

When GPIO function is enabled, the direction of this pin is controlled by bit 15 of the GPIODir register. The GPIO15 input/output value is controlled by bit 15 of the GPIOData register.

GPIO[21:16]

GPIO[21:16] are multiplexed with six P80 modem IA signals; in addition, GPIO19 can be used for MIRQn input signal, and GPIO20 is further multiplexed with MCSn. Six P80 modem IA signals are M_TXSIN, M_CLKIN, M_RXOUT, M_SCK, M_STROBE and M_CNTRL_SIN, they are multiplexed with GPIO16 through GPIO21 respectively. The control signals of this multiplexing are EXTIA_MODE and EXTIA_SEL, they are generated from SSD_P80. When EXTIA_MODE is 1, the six P80Modem signals will appear on the GPIO pins, these signals can be used to either interface to Data IA (when EXTIA_SEL is 0) or Voice IA (when EXTIA_SEL is 1). When EXTIA_MODE is 0, the GPIO[21:16] become GPIO and they are controlled by GPIODir and GPIOData registers as usual except GPIO20 can become MCSn if bit 5 GPIOConfig2 is set to 1 and GPIO19 can be used as MIRQn input.

GPIO22

GPIO22 is multiplexed with EADC_Sample signals. If EADC_Sample function is selected, the bit 6 of GPIOConfig2 register must be set to 1. To enable GPIO22 function, bit 6 of GPIOConfig2 must be set to 0.

When GPIO function is enabled, the direction of this pin is controlled by bit 22 of the GPIODir register. The GPIO22 input/output value is controlled by bit 22 of the GPIOData register.

GPIO[30:23]

GPIO[30:23] are multiplexed with PIOD[7:0] signals respectively; If GPIO function is selected, bit 8 of GPIOConfig2 register must be set to 1, else, PIOD function is selected.

When GPIO function is enabled, the direction of this pin is controlled by bit 30-23 of the GPIODir register. The GPIO[30:23] input/output value is controlled by bit 30-23 of the GPIOData register.

13.2 GPO/GPI Signals

The MFC2000 also provides eight GPO lines (GPO[7:0]), which are multiplexed with printer and scanner motor control signals.

GPO[3:0]	GPO[3:0] are multiplexed with the printer motor control lines on the four pins PM[3:0]/GPO[3:0]. These pins are selected as GPO's when bit 0 in the GPIOConfig1 register is set to 1, and are selected as printer motor control lines when this bit is set to 0. When these pins are selected as GPO's, the GPO[3:0] output values are controlled by the lowest 4 bits in the VPMPattern register, and appear on the output pins immediately.
GPO[7:4]	GPO[7:4] are multiplexed with the scanner motor control lines on the four pins SM[3:0]/GPO[7:4]. These pins are selected as GPO's when bit 1 in the GPIOConfig1 register is set to 1, and are selected as scanner motor control lines when this bit is set to 0. When these pins are selected as GPO's, the GPO[7:4] output values are controlled by the lowest 4 bits in the SMPattern register, and appear on the output pins immediately.
GPO[13:8]	If the PIO Interface is not enabled, the pins for PIODIR, BUSY, ACKn, SLCTOUT, PE and FAULTn will become GPIO8-GPIO13 respectively. These pins are selected as GPO function when bit8 of GPIOConfig2 register is set to 1, otherwise, PIO interface will be selected. The GPO[13:8] data is controlled by bit 5-0 of GPOData register.
GPO[17:14]	GPO[17:14] are multiplexed with AO3, AE3, AO2 and AE2 respectively, in addition, GPO14 is multiplexed with SSTXD2 and GPO15 is with SSSTA2. The multiplexing selection is controlled by bit 0 and bit 1 of the GPIOConfig2 register. When bit0 of the GPIOConfig2 register is 1, the SSTXD2/SSSTA2 function is selected, when bit 0 is 0 and bit 1 is 1, the GPO[17:14] function is enabled and the output value of the GPO[17:14] is controlled by bit 9-6 of GPOData register. To select AO[3:2]/AE[3:2], both bit 0 and bit 1 of the GPIOConfig2 register must be 0.
GPI[3:0]	Four PIO Interface input pins can be used as GPI when PIO interface is not used, the value on these GPI pins can be read from bit 15-12 of the GPOData register.

13.3 GPIO Control and Data Registers

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPIO Configuration 1 *(GPIOConfig1)* **01FF8831**	0=GPIO[8] 1=SSSTAT1	GPIO[14] Select bits		GPIO[13] Select bits		GPIO[12] Select bits		0=GPIO11 1=ALTTONE	Rst Value xxx00000b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
GPIO Configuration 1 *(GPIOConfig1)* **01FF8830**	0=GPIO10 1= PWM4 Enable	0=GPIO7 1=CS5n	0=GPIO6 1=CS4n	0=GPIO5 1=CS3n	0=GPIO4 1=CS2n	0=GPIO[1:0] 1=FRDn and FWRn	0=scan mot. SM[3:0] 1= GPO[7:4] Select	0=Prt Mot. PM[3:0] 1= GPO[3:0] Select	Rst Value 00h Read Value 00h

Register Description: GPIO Configuration1 Register

Bit 14-13:   GPIO[14] or ringer or SASRXD is selected.

Bit 14	Bit 13	the GPIO[14]/Ringer/SASRXD pin
0	0	GPIO[14]
0	1	SASRXD
1	0	Ringer
1	1	(Not used)

Bit 12-11:   GPIO[13] or PMPWRCTRL or SASTXD is selected.

Bit 12	Bit 11	the GPIO[13]/PMPWRCTRL/SASTXD pin
0	0	GPIO[13]
0	1	SASTXD
1	0	PMPWRCTRL
1	1	(Not used)

Bit 10-9:   GPIO[12] or SMPWRCTRL or SASCLK is selected.

Bit 10	Bit 9	the GPIO[12]/SMPWRCTRL/SASCLK pin
0	0	GPIO[12]
0	1	SASCLK
1	0	SMPWRCTRL
1	1	(Not used)

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPIO Configuration 2 *(GPIOConfig2)* **01FF8833**	Not Used	Not Used		Not Used		Not Used		0=PIO Select 1=GPO[13:8]/GPIO[30:23]	Rst Value xxx00000b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
GPIO Configuration 2 *(GPIOConfig2)* **01FF8832**	0=PM[1:0] 1= SPI_SIC/SPI _SID	0=GPIO22 1=EADC_S ample	0=GPIO2 0 1=MCSn	Not Used	0=GPIO15 1=SC_CLK1/ 2C	0=GPIO8 1=SC_CLK1/ 2B	0=AE[3:2]/AO [3:2] 1=GPO[17:14] Select	0=GPIO[3:2] 1= SS2 Select	Rst Value 00h Read Value 00h

Register Description: GPIO Configuration2 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPI/GPO Data *(GPOData)* **01FF883D**	GPI3 Data	GPI2 Data	GPI1 Data	GPI0 Data	Not Used	Not Used	GPO17 Data	GPO16 Data	Rst Value 00000000b Read Value xxxx00xxb
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
GPI/GPO Data *(GPOData)* **01FF883C**	GPO15 Data	GPO14 Data	GPO13 Data	GPO12 Data	GPO11 Data	GPO10 Data	GPO9 Data	GPO8 Data	Rst Value 00h Read Value xxh

Register Description: GPO Data Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPIO Data 1 *(GPIOData1)* **01FF8835**	GPIO15 Data	GPIO14 Data	GPIO13 Data	GPIO12 Data	GPIO11 Data	GPIO10 Data	GPIO9 Data	GPIO8 Data	Rst Value x0000000b Read Value 0xxxxxxxb
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
GPIO Data 1 *(GPIOData1)* **01FF8834**	GPIO7 Data	GPIO6 Data	GPIO5 Data	GPIO4 Data	GPIO3 Data	GPIO2 Data	GPIO1 Data	GPIO0 Data	Rst Value 00h Read Value xxh

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPIO Data 2 *(GPIOData2)* **01FF8837**	(Not Used)	GPIO30 Data	GPIO29 Data	GPIO28 Data	GPIO27 Data	GPIO26 Data	GPIO25 Data	GPIO24 Data	Rst Value x0000000b Read Value 0xxxxxxxb
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
GPIO Data 2 *(GPIOData2)* **01FF8836**	GPIO23 Data	GPIO22 Data	GPIO21 Data	GPIO20 Data	GPIO19 Data	GPIO18 Data	GPIO17 Data	GPIO016 Data	Rst Value 00h Read Value xxh

Register Description: GPIO Data 1/2 Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPIO Direction 1 *(GPIODi1)* **01FF8839**	GPIO15 0=in 1=out	GPIO14 0=in 1=out	GPIO13 0=in 1=out	GPIO12 0=in 1=out	GPIO11 0=in 1=out	GPIO10 0=in 1=out	GPIO9 0=in 1=out	GPIO8 0=in 1=out	Rst Value x0000000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
GPIO Direction 1 *(GPIODir1)* **01FF8838**	GPIO7 0=in 1=out	GPIO6 0=in 1=out (Must be set to 0 if sartsel=1)	GPIO5 0=in 1=out	GPIO4 0=in 1=out	GPIO3 0=in 1=out	GPIO2 0=in 1=out	GPIO1 0=in 1=out	GPIO0 0=in 1=out	Rst Value 00h Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
GPIO Direction 2 *(GPIODir2)* **01FF883B**	(Not Used)	GPIO30 0=in 1=out	GPIO29 0=in 1=out	GPIO28 0=in 1=out	GPIO27 0=in 1=out	GPIO126 0=in 1=out	GPIO25 0=in 1=out	GPIO24 0=in 1=out	Rst Value x0000000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
GPIO Direction 2 *(GPIODir2)* **01FF883A**	GPIO23 0=in 1=out	GPIO22 0=in 1=out	GPIO21 0=in 1=out	GPIO20 0=in 1=out	GPIO19 0=in 1=out	GPIO18 0=in 1=out	GPIO17 0=in 1=out	GPIO16 0=in 1=out	Rst Value 00h Read Value 00h

Register Description: GPIO Direction 1/2 Register

14. Compressor and Decompressor

14.1 Functional Description

The T.4 data compression/decompression process is primarily implemented in hardware within the MFC2000 (see Figure 14-2). Hardware provides compression and decompression of data within a line, and adds fill bits at the end of a line to cause the number of bits (coded plus fill) to be a multiple of 16 (halfword justified) for T.4 MH and MR. For coding (compression), firmware (T.4 subsystem task) appends end-of-line characters and, if required, fill bits to support the minimum line times of CCITT T.4. On the receive side, firmware passes the coded line to the T.4 hardware. Data flow control, to prevent buffer overflow/underflow, is also the responsibility of the firmware.

Line times less than 5 msec per line are supported by the MFC2000s' T.4 hardware; the line time supported by the overall system depends on other system factors (e.g. resolution, concurrency, etc). T4 line lengths up to 2046 bytes are supported.

There are three FIFOs contained in the compressor/decompressor block. All FIFOs operate identically from both a hardware and firmware standpoint. The FIFO structure is illustrated in Figure 14-2. The FIFOs consist of a quad halfword FIFO structure and a single halfword holding register. The quad FIFO structure is emptied and filled by either DMA or CPU operations. The holding register is emptied and filled by the compressor/decompressor logic.

All locations within the FIFO can also be arbitrarily read and written by firmware. This allows firmware to save the entire FIFO state and restore it at a later point in time, thereby permitting an unlimited number of data streams to be concurrently processed.
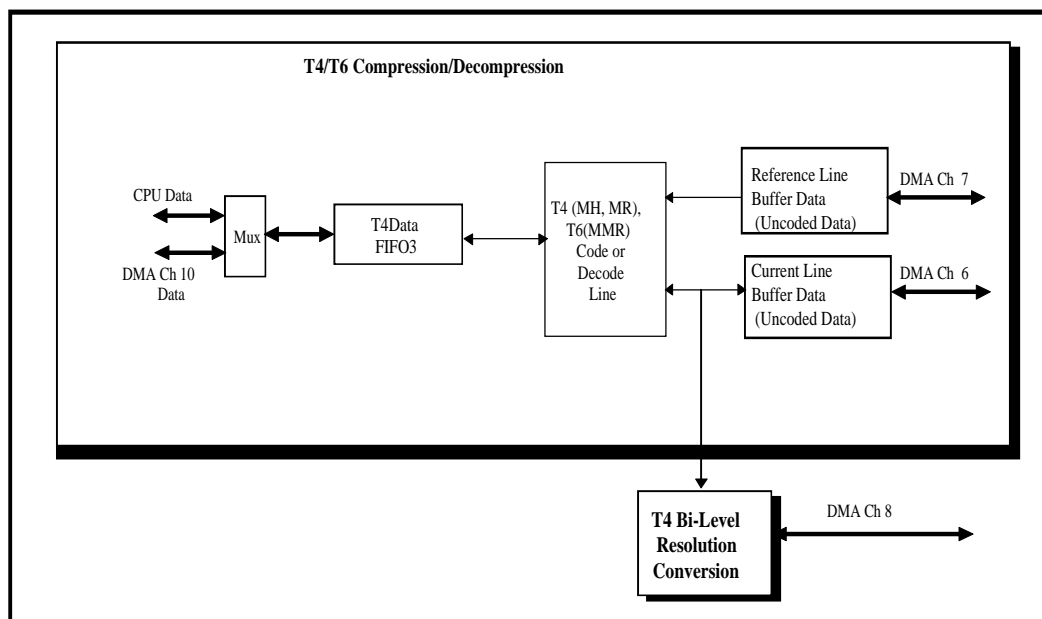
Figure 14-1. Data Flow for Compression/Decompression
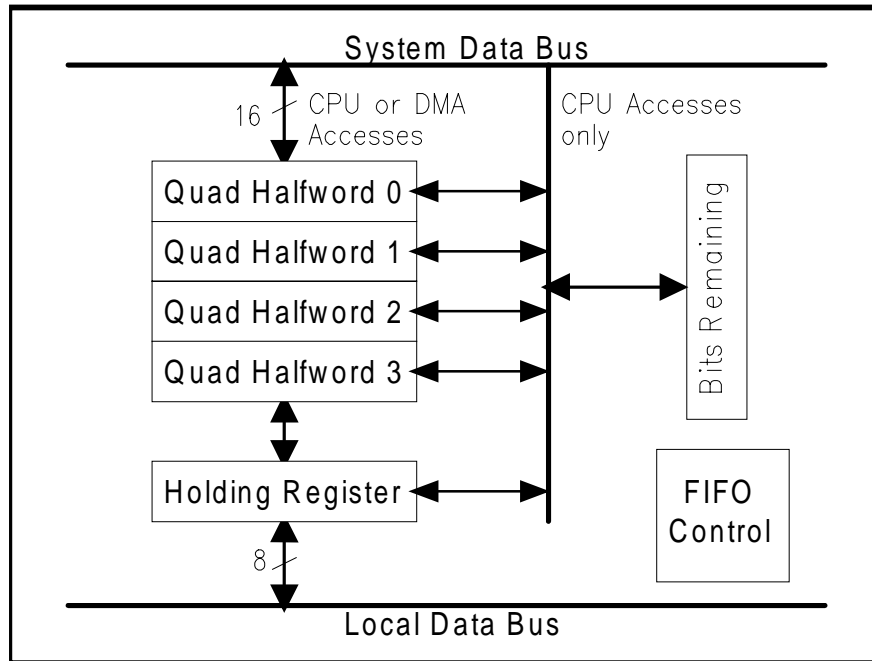
Figure 14-2. Compressor/Decompressor FIFO Structure

14.2 Register Description

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Configuration *(T4Config)* **$01FF8171**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Disable Hardware DMA CH10	Disable Hardware DMA CH6	Disable Hardware DMA CH7	Rst. Value xxxxx000b Read Value: 00000000b

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
T4 Configuration *(T4Config)* **$01FF8170**	(Not Used)	Enable smart EOL operation	Enable non-inverted flush (zero fill)	Compressed data src./dest. 00=T4Data0 01=T4Data1 10=T4Data2 11=T4Data3		Enable Decompression	T4 Coding/Decoding Mode 00=MH 01=MR 1-dimensional 10-MR 2-dimensional 11=MMR		Rst. Value x0000000b Read Value: 00000000b

Bit 15-11:   Not used

Bit 10:   Disable coded hardware DMA.   Setting this bit to 1 disables hardware DMA accesses to the coded data FIFOs (DMA CH10). Clearing this bit allows hardware DMA to the coded data FIFOs. This bit should be set when firmware DMA is to be carried out to the coded data FIFO.

Bit 9:   Disable uncoded hardware DMA.

                           Setting this bit to 1 disables hardware DMA accesses to the uncoded data FIFO (DMA CH6). Clearing this bit allows hardware DMA to the uncoded data FIFO. This bit should be set when firmware DMA is to be carried out to the uncoded data FIFO.

Bit 8:   Disable reference hardware DMA.

Setting this bit to 1 disables hardware DMA accesses to the reference data FIFO (DMA CH7). Clearing this bit allows hardware DMA to the reference data FIFO. This bit should be set when firmware DMA is to be carried out to the reference data FIFO.

Bit 7:   Not used

Bit 6:   Enable smart EOL operation.   Setting the smart EOL operation bit causes the posting of an EOL in the T4 Status register to be delayed until the selected T4Data FIFO has transferred all of its data. Clearing the bit causes EOL to be set as soon as the EOL has been processed, regardless of the condition of the FIFO.

Bit 5:   Enable Non-inverted flush.   Setting the non-inverted flush bit causes the flush operation to use zero as fill bits rather than the inverted value of the last CW bit. Inverted fill allows the user to determine the number of fill bits added to halfword align the last CW.

Bit 4-3:   Compressed data source/destination.

00=T4Data0 (default)
01=T4Data1
10=T4Data2 11=T4Data3

Warning: *No distinction is made between FIFO accesses by either the DMA channel or the CPU. This means it is possible for both sources to modify the FIFO contents, resulting in unpredictable operation of the compressor/decompressor. Firmware must ensure that the CPU does not modify the FIFO contents without first disabling the DMA channel.*

Bit 2:   Enable Decompression

Bits 1-0:   T4 Coding/Decoding Mode   00=MH01=MR 1-d
10=MR 2-d11=MMR

When MMR compression is selected, 2-d coding will occur (similar to MR 2-d), however, the last CW in the line will not be halfword aligned by adding fill bits. Code words from the next compressed line will be concatenated to whatever is already in the selected T4Data FIFO. MMR decompression is similar to MR 2-d decoding, except that the end-of-line condition is based on the number of decoded bits rather than an EOL code word.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Control *(T4Control)* **$01FF8173**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value: 00h

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
T4 Control *(T4Control)* **$01FF8172**	Reset T4 Block (w)	Terminating Code Error	Enable Bi-level Resolution Conversion	Decode Output Enable	Vertical OR enable (decompress mode only)	Reset/Halt Coding or decoding	Start coding or decoding	Flush CW (completes byte)	Rst. Value x0000000b Read Value 00h

Bit 15-8:   Not used

Bit 7:   Writing a 1 to this bit causes the T4 logic to be reset, including all internal state machines and FIFOs. This bit is useful when switching between compression and decompression modes.

Bit 6:   A Terminating Code Error is indicated when a *run length = 0* code is expected but not received at the end of a line, and the Terminating Code Error also sets the Line error bit

Bit 5:   Enable Bi-level Resolution Conversion.

                         Bi-level resolution conversion is only valid in decompression mode. When this bit is set, horizontal resolution conversion will be performed on the decoder output. Both the unconverted line and the converted line will be output to the line buffer. The converted line will be written to the line buffer via DMA channel 8.

Bit 4:   Decode output enable.   Decode Output Enable allows decoded data to be output to the line buffers. This bit should not be set in MH or MR decoding until the first EOL has been found. The decoder output should also be disabled after an error line until the next EOL is found. This bit should always be set for the MMR decoder.

Bit 3:   Vertical OR enable.   The Vertical OR bit is only valid in decompression mode when the bi-level resolution conversion bit in the T4Config register is also set. When the Vertical OR bit is set, the resolution converted decoded output data is ORed with the corresponding byte from the previous line before storing.

Bit 2:   Reset or halt the coding or decoding.

                         Setting the Reset/Halt bit immediately halts coding or decoding of a line, resets all the T4 Status bits, and clears the active T4Data FIFO. All other bits in this register are ignored if the Reset/Halt bit is set. When coding or decoding is in progress, writing to any bit in this register except Reset/Halt is ignored.

Bit 1:   Start coding or decoding.   The Start bit begins the coding or decoding of the line according to the configuration selected per the T4Config register.

Bit 0:   Flush the code word.   The Flush CW bit is only valid during MMR compression mode. Writes to this bit are ignored when coding/decoding is in progress, or when decompression is selected. When the Flush CW bit is set at the end of coding a line, it causes any data in the active T4Data FIFO to be filled in order to make a complete halfword which can then be read by the CPU or DMA controller. The fill bits are the opposite value of the last CW bit, unless the non-inverted flush bit in the T4Config register is set. In this case, the data is flushed with zeros. One halfword is always output when the flush bit is set even if the last CW is already halfword aligned.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Status *(T4Status)* $01FF8174 (R)	T4 Datax DMA Page Boundary	T4 Datax FIFO DMA Acknowledge	T4 Datax FIFO DMA Request	Write T4Datax FIFO Ready	Read T4Datax FIFO Ready	Write Uncoded FIFO Ready	Read Uncoded FIFO Ready	Write Reference FIFO Ready	Rst. Value 00h Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
T4 Status *(T4Status)* $01FF8175 (R)	EOL Condition	Tag Bit	non-Consecutive EOL/ MMREOL	Line Error	Line Error Type			All White Line	Rst. Value 00h Read Value 00h

Bit 15:   This a copy of the hardware signal from the DMA channel indicating when the DMA block size has been reached and is used by the FIFO to prevent its DMA request from being asserted if Smart EOL Operation has been enabled via the T4 Configuration register.

Bit 14:   This is a copy of the hardware signal from the DMA channel indicating a DMA transfer is in progress.

Bit 13:   This is a copy of the hardware signal to the DMA channel indicating the desire to initiate a DMA transfer.

Bit 12:   The WrT4DataFIFORdy bit is set during decompression mode when the active T4Data FIFO is not full and is cleared when the FIFO is full.

Bit 11:   The RdT4DataFIFORdy bit is set when compressed data is available in the active T4Data FIFO. This bit is cleared when there is no data available.

Bit 10:   The WrUncodedFIFORdy bit is set during compression mode when the Uncoded Line FIFO is not full and is cleared when the FIFO is full.

Bit 9:   The RdUncodedFIFORdy bit is set when uncompressed data is available in the Uncoded Line FIFO. This bit is cleared when there is no data available.

Bit 8:   The WrRefFIFORdy bit is set during either MR/MMr compression or decompression mode when the Reference Line FIFO is not full and is cleared when the FIFO is full.

Bit 7:   End of Line Conditions.   For coding, the End-of-line Condition occurs when all data in the line has been compressed and the last codeword has been written to the T4Data FIFO.

For decoding, the End-of-line Condition occurs when the decoding of a line is complete, and all decoded data has been written to the line buffer. For MH and MR modes, decoding is complete when the EOL codeword (and Tag bit if in MR mode) have been decoded. For MMR mode, decoding is complete when the number of bytes specified by the T4Bytes register have been decoded.

Bit 6:   The Tag bit is only valid when the End-of-line Condition bit is also set.

Bit 5:   The NonConsecutiveEOL/MMREOL is only valid when the End-of-line Condition bit is also set.

Bit 4:   Line Error.   Line errors apply to the decoding mode only. If a line error occurs while decoding a line, the T.4 hardware will set the LineError bit. Decoding will continue until an EOL code is found (MH and MR modes only), but data decoded following the error will not be output to the line buffer.. The type of line error can be determined by examing the Line Error Type bit field.

Bit 3-1:   Line Error Type.   When a line error occurs, the type of line error is placed in this bit field. The following line errors are detected by the T.4 hardware:

LineError Code:	Error:	Description:
000b	No error	
001b	RTC (Return to Control) Error	Fill bits occurred between consecutive EOL codewords.
010b	Terminating Code Error	Codewords for white or black run lengths equal to zero which occur at the end of a line are missing. When this error occurs the TC error bit (T4Status register) is set as well as the normal error bit; this is to allow the user to ignore these errors if desired.
011b	Reserved	
100b	Line Too Short	EOL code found before the number of bytes specified in the T4Bytes register have been decoded.
101b	Line Too Long	The number of data bytes decoded before the EOL code is received exceeds the line length specified in the T4Bytes register.
110b	Code Word Error	An invalid code word was decoded.
111b	Reserved	

Bit 0:   Indicate the decoded line is all white when this bit is set. This bit is reset when the decoding of a line is started.

Notes:

1. *An IRQ is generated whenever any of the shaded bits shown in the T4 Status register (bits 4, 7–9) and their associated bits in the T4IntMask register are both set.*
2. *Bits 0–6, and 9 only apply to decompression mode.*
3. *Bits 0–7 are cleared at the start of a new line.*

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Line Length *(T4Halfwords)* $01FF8179	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Line Length bits 9-8		Rst. Value xxxxxx00b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
T4 Line Length *(T4Halfwords)* $01FF8178	Line Length bits 7-0								Rst. Value 00h Read Value 00h

Bit 15-10:   Not used

Bit 9-0:   Line Length   The T4 Line Length register defines the number of bytes in the line to be coded or decoded. Line length values from 001h through 3ffh correspond to 1 through 1023 halfwords of data.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Interrupt Mask *(T4IntMask)* $01FF8177	(Not Used)	(Not Used)	(Not Used)	Enable Wr T4Data FIFO Rdy	Enable Rd T4Data FIFO Rdy	Enable Wr Uncoded FIFO Rdy	Enable Rd Uncoded FIFO Rdy	Enable Wr Ref FIFO Rdy	Rst. Value xxx00000b Read Value: 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
T4 Interrupt Mask *(T4IntMask)* $01FF8176	Enable End of Line	(Not Used)	(Not Used)	Enable Line Error	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value 0xx0xxxxb Read Value 00h

Bit 15-13:   Not used

Bit 12:   WrT4DataFIFORdy Enable Interrupt

Bit 11:   RdT4DataFIFORdy Enable Interrupt

Bit 10:   WrUncodedFIFORdy Enable Interrupt

Bit 9:   RdUncodedFIFORdy Enable Interrupt

Bit 8:   WrRefFIFORdy Enable Interrupt

Bit 7:   End of Line Enable Interrupt.

Bit 6-5:   Not used

Bit 4:   Line Error Enable Interrupt.

Bit 3-0:   Not used

The four interrupt enable bits correspond to bits in the T4 Status register. The position of each interrupt enable bit is in the same place as its corresponding bit in the T4 Status register (bit 7 – End of Line Enable Interrupt corresponds to bit 7 – EOL Condition).

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 FIFO Bits Remaining Register *(T4FIFOBitRem)* **$01FF817B**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value: 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
T4 FIFO Bits Remaining Register *(T4FIFOBitRem)* **$01FF817A**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Bits remaining				Rst. Value xxxx0000b Read Value x0000000b

Bit 15-4: Not used

Bit 3-0: Number of bits remaining in FIFO (holding register), with values ranging from 0 to 15. The interpretation of bits remaining depends on the direction of data flow. If the compressor/decompressor is emptying the FIFO, bit remaining is the number of that contain data that have yet to be processed. If the compressor/decompressor is filling the FIFO, bits remaining is the number of bits that are empty.

T.4 data FIFO halfword[3:0] for the coded data to/from the external memory (for DMA channel 10):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Data FIFO *(T4Data1FIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
T.4 Data FIFO *(T4Data1FIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

- Address assignment for T.4 Data FIFO halfword[3:0]

FIFO halfword	The register name	Address
FIFO halfword 3	*T4DataFIFO3*	01FF8117-16
FIFO halfword 2	*T4DataFIFO2*	01FF8115-14
FIFO halfword 1	*T4DataFIFO1*	01FF8113-12
FIFO halfword 0	*T4DataFIFO0*	01FF8111-10

T.4 Ref. line Data FIFO halfword[3:0] for the uncoded data to/from the external memory (for DMA channel 7):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Reference Line Data FIFO *(T4RefDataFIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
T.4 Reference Line Data FIFO *(T4RefDataFIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

- Address assignment for T.4 Ref. Line Data FIFO halfword[3:0]

FIFO halfword	The register name	Address
FIFO halfword 3	*T4RefDataFIFO3*	01FF8157-56
FIFO halfword 2	*T4RefDataFIFO2*	01FF8155-54
FIFO halfword 1	*T4RefDataFIFO1*	01FF8153-52
FIFO halfword 0	*T4RefDataFIFO0*	01FF8151-50

T.4 Current line Data FIFO halfword[3:0] for the uncoded data to/from the external memory (for DMA channel 6):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Current Line Data FIFO *(T4CurDataFIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
T.4 Reference Line Data FIFO *(T4CurDataFIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

- Address assignment for T.4 Current Line Data FIFO halfword[3:0]

FIFO halfword	The register name	Address
FIFO halfword 3	*T4CurDataFIFO3*	01FF8167-66
FIFO halfword 2	*T4CurDataFIFO2*	01FF8165-64
FIFO halfword 1	*T4CurDataFIFO1*	01FF8163-62
FIFO halfword 0	*T4CurDataFIFO0*	01FF8161-60

T.4 Data FIFO Holding Register for the coded data to/from the external memory (for DMA channel 10):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Data FIFO Holding Register *(T4Data1Hold)* **$01FF8119**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Data FIFO Holding Register *(T4Data1Hold)* **$01FF8118**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

T.4 Ref. Line Data FIFO Holding Register for the Uncoded data to/from the external memory (for DMA channel 7):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Ref. Line Data FIFO Holding Register *(T4RefDataHold)* **$01FF8159**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Ref. Line Data FIFO Holding Register *(T4RefDataHold)* **$01FF8158**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

T.4 Current Line Data FIFO Holding Register for the Uncoded data to/from the external memory (for DMA channel 6):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Current Line Data FIFO Holding Register *(T4CurDataHold)* **$01FF8169**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Current Line Data FIFO Holding Register *(T4CurDataHold)* **$01FF8168**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

T.4 Data FIFO Control register for the coded data to/from the external memory (for DMA channel 10):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Data FIFO Control Register *(T4Data1FIFOCtrl)* **$01FF811B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Data FIFO Control Register *(T4Data1FIFOCtrl)* **$01FF811A**	(Not Used)	Holding Register Full	(Not Used)	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0x00000b Read Value 00h

Bit 15	This bit indicates that the FIFO is active and capable of generating DMA requests.
Bit 14	This bit is essentially a copy of the DMA Request output signal and indicates that the FIFO wishes to transfer data.
Bit 13	This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.
Bit 12-10	This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 0 to 4. Values greater than 4 are treated as 4.
Bit 9-8	This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.
Bit 6	This bit indicates that there is a full word in the holding register
Bit 4-2	This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.
Bit 1-0	This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

> *Note: It is strongly recommended that the FIFO be disabled before firmware writes to the FIFO Control register. The FIFO Control register contains data that is essential to the FIFO's operation. If this data is changed while the FIFO is operating, unpredictable operation will result.*

T.4 Ref. Line Data FIFO Control register for the Uncoded data to/from the external memory (for DMA channel 7):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Ref. Line Data FIFO Control Register *(T4RefDataFIFOCtrl)* **$01FF815B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Ref. Line Data FIFO Control Register *(T4RefDataFIFOCtrl)* **$01FF815A**	(Not Used)	Holding Register Full	(Not Used)	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0x00000b Read Value 00h

T.4 Current Line Data FIFO Control register for the Uncoded data to/from the external memory (for DMA channel 6):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Current Line Data FIFO Control Register *(T4CurDataFIFOCtrl)* **$01FF816B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Current Line Data FIFO Control Register *(T4CurDataFIFOCtrl)* **$01FF816A**	(Not Used)	Holding Register Full	(Not Used)	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0x00000b Read Value 00h

T.4 Data FIFO Data Port for the coded data to/from the external memory:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Data FIFO Data Port *(T4Data1Port)* **$01FF811D**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
T.4 Data FIFO Data Port *(T4Data1Port)* **$01FF811C**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T.4 Data FIFO Data Port Transfer *(T4Data1PortTfr)* **$01FF814F (DW)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value: xxh
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
T.4 Data FIFO Data Port Transfer *(T4Data1PortTfr)* **$01FF814E (DW)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value xxh

Bit 15-0:   Not used.

Writing any value to this register causes the data in FIFO to push or pop data through the FIFO data port and the FIFO Quad Halfword 1-4 registers. The FIFO operation depends on the direction of data flow. If the compressor/decompressor is emptying the FIFO, data is transferred from the FIFO data port register to the FIFO Quad Halfword 1 register, simultaneously with data transfer from Quad Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3, and Quad Halfword 3 to Quad Halfword 4, with the original data of Quad Halfword 4 being lost. If the compressor/decompressor is filling the FIFO, data is transferred from the FIFO Quad Halfword 1 register to the FIFO data port register, simultaneously with data transfer from Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the contents of Quad Halfword 4 becoming indeterminate.

T.4 Ref. Line Data FIFO Data Port for the Uncoded data to/from the external memory:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Ref. Line Data FIFO Data Port *(T4RefDataPort)* **$01FF815D**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
T.4 Ref. Line Data FIFO Data Port *(T4RefDataPort)* **$01FF815C**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Ref. Line Data FIFO Data Port Transfer *(T4RefDataPortTfr)* **$01FF815F (DW)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value: xxh
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
T4 Ref. Line Data FIFO Data Port Transfer *(T4RefDataPorTfrt)* **$01FF815E (DW)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value xxh

Bit 15-0:   Not used   Writing any value to this register causes the data in FIFO to push or
                                             pop data through the FIFO data port and the FIFO Quad Halfword 1-4
                                             registers. The FIFO operation depends on the direction of data flow. If
                                             the compressor/decompressor is emptying the FIFO, data is
                                             transferred from the FIFO data port register to the FIFO Quad
                                             Halfword 1 register, simultaneously with data transfer from Quad
                                             Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3,
                                             and Quad Halfword 3 to Quad Halfword 4, with the original data of
                                             Quad Halfword 4 being lost. If the compressor/decompressor is filling
                                             the FIFO, data is transferred from the FIFO Quad Halfword 1 register
                                             to the FIFO data port register, simultaneously with data transfer from
                                             Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad
                                             Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the
                                             contents of Quad Halfword 4 becoming indeterminate.

T.4 Current Line Data FIFO Data Port for the Uncoded data to/from the external memory:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
T.4 Current Line Data FIFO Data Port *(T4CurDataPort)* **$01FF816D**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
T.4 Current Line Data FIFO Data Port *(T4CurDataPort)* **$01FF816C**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value xxh Read Value xxh

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
T4 Current Line Data FIFO Data Port Transfer *(T4CurDataPortTfr)* **$01FF816F (DW)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value: xxh
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
T4 Current Line Data FIFO Data Port Transfer *(T4CurDataPortTfr)* **$01FF816E (DW)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value xxh

Bit 15-0:   Not used   Writing any value to this register causes the data in FIFO to push or
                                             pop data through the FIFO data port and the FIFO Quad Halfword 1-4
                                             registers. The FIFO operation depends on the direction of data flow. If
                                             the compressor/decompressor is emptying the FIFO, data is
                                             transferred from the FIFO data port register to the FIFO Quad
                                             Halfword 1 register, simultaneously with data transfer from Quad
                                             Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3,
                                             and Quad Halfword 3 to Quad Halfword 4, with the original data of
                                             Quad Halfword 4 being lost. If the compressor/decompressor is filling
                                             the FIFO, data is transferred from the FIFO Quad Halfword 1 register
                                             to the FIFO data port register, simultaneously with data transfer from
                                             Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad
                                             Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the
                                             contents of Quad Halfword 4 becoming indeterminate.

15. Synchronous/Asynchronous Serial Interface (SASif)

15.1 Functional Description

The SASIF is a Synchronous/Asynchronous Receiver/Transmitter which performs serial-parallel conversion on data received from a peripheral device, and parallel-to-serial conversion of data for transmission to a peripheral device. The interface consists of serial transmit data (SASTXD), serial receive data (SASRXD), and a serial clock (SASSCLK) signals. Figure 15-1 is a block diagram of the SASIF.

The SASIF includes a programmable baud rate generator for asynchronous and synchronous operations. Four interrupts can be generated for the ARM. These are Transmit Shift Register empty, Transmit Buffer Register Empty, or Receive Buffer Register full, and Receive FIFO Timeout. These interrupts can each be uniquely enabled or disabled.

The SASIF can also be set to FIFO mode. In this mode, the ARM will be relieved of excessive interrupt handling and will allow for an overall faster transmit and receive speed. A FIFO mode can be enabled for either the receive portion or the transmit portion of the SASIF. When the transmit FIFO is enabled, a 16-byte FIFO will be used to hold up to 16 bytes of data to be transmitted. The Transmit Buffer Empty interrupt will only be generated if the transmit FIFO has no new data to transmit (it is empty). When the receive FIFO is enabled, a 16-byte FIFO will be used to store up to 16 bytes of data which are received from the receive shift register. The Receive Buffer Full interrupt will only be generated if the receive FIFO has no room left for incoming data (it is full).

The receive FIFO also includes a timeout function. If the receive FIFO is not full, but the last newly received byte was longer than four byte times ago (based on the programmed baud rate), then a timeout interrupt will be generated. A "byte time" is the time in which it takes to receive one byte of data.

The receive and transmit data is double buffered to provide more time for the ARM to process receive data. The data shifting order, MSB to LSB or LSB to MSB, and the SASCLK polarity are programmable.

The synchronous communication mode enables the ARM to transmit and receive data synchronous referencing to the serial clock.

The ARM can read the status of the SASIF at any time during operation. Status includes IRQ source (SASTXD or SASRXD) and operation mode (synchronous or asynchronous). During the Input/Output mode the SASTXD, SASRXD, and SASSCLK values can also be read or written by the ARM. The contents of each byte within the transmit and receive fifo can be read any time during operation, as well as the input and output pointers to both fifos.

Following is the feature summary of the Serial Interface:

- Full duplex, three wire system: SASSCLK (serial clock), SASTXD (Transmit data), SASRXD (Receive data)
- Independent transmit data shift register and receive data shift register
- Double buffered receive and transmit data register
- Programmable 7 or 8 data bit asynchronous serial-interface with a start bit, a stop bit and no parity
- 8 data bit synchronous serial-interface with programmable data shifting order
- Programmable baud rate generator supports up to 14400 baud asynchronous transmit and receive (when in fifo mode is not on), and up to 2 ICLK synchronous transmit and receive
- Supports up to 115.2 Kbaud asynchronous transmit and receive when in FIFO mode
- Single interrupt generation for Transmit Data Shift Register empty, Transmit Data Buffer Register empty, Receive Data Buffer Register full, and Receive FIFO Timeout
- In the FIFO mode, transmitter and receiver are each buffered with 16 byte FIFO's to reduce the number of interrupts to the ARM.
- Maximum ±1% SASTXD Async transmit baud rate error
- Maximum ±2.5% SASRXD Async receive baud rate allowable error
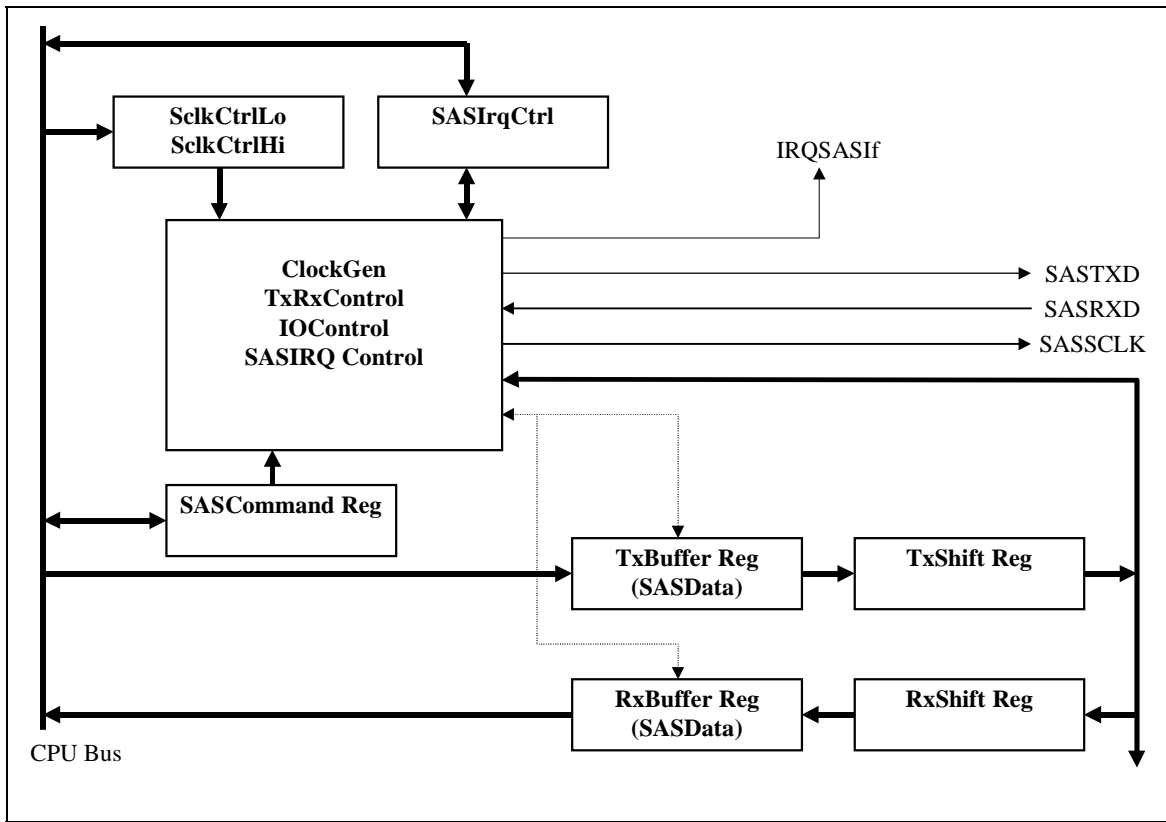
Figure 15-1. SASIF Block Diagram

15.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASCmd **01FF80F1**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
SASCmd **01FF80F0**	(Not Used)	TxFIFOEnb	RxFIFOEnb	L2MSB	SASSCLKPol	DataLen	SASMode		Rst. Value X0000000b Read Value 00h

Register Description: SASIF Command Register

Bit 15-7:   Not used

Bit 6:   TxFIFOEnb—Read/Write bit.   This bit controls whether or not the 16-byte transmit FIFO is to be used. When the TxFIFOEnb is cleared (default), the transmit FIFO will not be used and an interrupt will be generated each time a byte is sent. If TxFIFOEnb is set, the transmit FIFO will be used and an interrupt will be generated only when the FIFO is completely emptied. This should **ONLY** be written to during setup and not after the SASIF has started sending or receiving data, as data loss could occur.

Bit 5:   RxFIFOEnb—Read/Write bit.   This bit controls whether or not the 16-byte receive FIFO is to be used. When the RxFIFOEnb is cleared (default), the receive FIFO will not be used and an interrupt will be generated for each byte received. If RxFIFOEnb is set, then the receive FIFO will be used and an interrupt will only be generated if the FIFO is full or a receive FIFO timeout occurs. This should **ONLY** be written to during setup and not after the SASIF has started sending or receiving data, as data loss could occur.

Bit 4:   L2MSB—Read/Write bit.   This bit controls the shifting order of the transmit and receive registers. When the L2MSB bit is cleared (default) and the SASIF is set to the AsyncMode or SyncMode, the transmit and receive shift register will shift from the MSB to the LSB. When the L2MSB bit is set, the transmit and receive register will shift from the LSB to MSB.

Bit 3:   SASSCLKPol—Read/Write bit.

This bit controls the polarity of the SASSCLK pin for the sync mode. When the SASSCLKPol bit is cleared (default) and the SASIF is set to AsyncMode or SyncMode, the SASSCLK polarity will be non-inverted, which will enable the external logic to clock the SASTXD pin on the rising edge of the SASSCLK pin. When the SASSCLKPol bit is set and the SASIF is set to AsyncMode or SyncMode, the SASSCLK polarity will be inverted, which will enable the external logic to clock the SASTXD pin on the falling edge of the SASSCLK pin.

Bit 2:   DataLen—Read/Write bit.   This bit controls the number of bit for AsyncMode transmit. This bit is not effective for the SyncMode. When the DataLen is cleared (default) and the SASIF is set to the AsyncMode, the TxBuffer register bit 7 will be the MSB and bit 0 will be the LSB for asynchronous transmit. The transmit shifting order is controlled by the L2MSB bit. When the DataLen is set and the SASIF is set to the AsyncMode, the TxBuffer register bit 6 will be the MSB and bit 0 will be the LSB for asynchronous transmit.

Bit1-0:   SASMode—Read/Write bits.   These two bits control the SASIF operation modes.

   00: AsyncMode (default)
   01: SyncMode
   10 & 11: Reserved

> *Note: The TxFIFO enable and the RxFIFO enable should either be set or cleared once in the setup of the SASIF device. Enabling or disabling the FIFOs once the MFC is up and running could have unpredicted results if data is being received or sent by the shift registers.*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASRxData **01FF80F3**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASRxData **01FF80F2**	SASRx Data(7)	SASRx Data(6)	SASRx Data(5)	SASRx Data(4)	SASRx Data(3)	SASRx Data(2)	SASRx Data(1)	SASRx Data(0)	Rst. Value 00h Read Value 00h

Register Description: SASIF Rx Data Buffer Register

> *Note: This is the Receive buffer data register. Reading from this register will return the data in the Rxbuffer (for fifo disabled) or the oldest valid received data (for fifo enabled). However, after a read, a write MUST be done to this register in order to let the SASIF know that data has been read. This was done to help with emulation testing. Writing to the SASData Register will clear the RxBufFull or RxFifoThresh status bits.*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASTxData **01FF80FB**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASTxData **01FF80FA**	SASTx Data(7)	SASTx Data(6)	SASTx Data(5)	SASTx Data(4)	SASTx Data(3)	SASTx Data(2)	SASTx Data(1)	SASTx Data(0)	Rst. Value 00h Read Value 00h

Register Description: SASIF Tx Data Buffer Register (write only)

> *Note: This is the Transmit buffer data register. It is a write only register. Writing to this register placed data into the transmit buffer register (or the transmit fifo if TxFifo is enabled). Writing to the SASTxData Register will clear the TxBufEmpty or TxFifoThresh status bits.*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASFifoPtr **01FF80F7**			Tx Input Pointer				Tx Output Pointer		Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASFifoPtr **01FF80F6**			Rx Input Pointer				Rx Output Pointer		Rst. Value 00h Read Value 00h

Register Description: SASIF Fifo Pointer Register

Bit 15-12: Transmit FIFO input pointer   (Read Only bit)

Bit 11-8:   Transmit FIFO output pointer   (Read Only bit)

Bit 7-4:   Receiver FIFO input pointer   (Read Only bit)

Bit 3-0:   Receiver FIFO output pointer   (Read Only bit)

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASFifoQty **01FF80FD**	(Not Used)			Tx FIFO Quantity			TxFIFOThreshVal		Rst. Value X0000011b Read Value 00000011b
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASFifoQty **01FF80FC**	(Not Used)			Rx FIFO Quantity			RxFIFOThreshVal		Rst. Value X0000011b Read Value 00000011h

Register Description: SASFifoQty Register

Bit 15:   Not Used

Bit 14-10: TxFIFOQty (Transmit FIFO Quantity)—Read Only bits.

> These five bits hold the value of the amount of bytes stored within the transmit FIFO. As data is written to or removed from the TxFIFO, the quantity will be adjusted accordingly.

Bit 9-8:   TxFIFOThreshVal (Transmit FIFO Threshhold value)—Read/Write bits.

> Depending on how these two bits are set, will affect when the transmit FIFO considers itself empty enough to cause an interrupt.

Bit Value	# of bytes left in TxFIFO which should cause interrupt
"00"	12
"01"	8
"10"	4
"11"	2

Bit 7:   Not Used

Bit 6-2:   RxFIFOQty (Receive FIFO Quantity)—Read Only bits.

These five bits hold the value of the amount of bytes stored within the receive FIFO. As data is written to or removed from the RxFIFO, the quantity will be adjusted accordingly.

Bit 1-0:   RxFIFOThreshVal (Receive FIFO Threshhold value)—Read/Write bits.

Depending on how these two bits are set, will affect when the receive FIFO considers itself full enough to cause an interrupt.

Bit Value	# of bytes filled in RxFIFO which should cause interrupt
"00"	4
"01"	8
"10"	12
"11"	14

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASIrqSts **01FF80F9**	(Not Used)	(Not Used)	(Not Used)	RxFifo Overrun (Read only)	TxFifoFull (Read only)	TxFifoEmpty (Read only)	RxFifoFull (Read only)	RxFifoEmpty (Read only)	Rst. Value XXh Read Value 05h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
SASIrqSts **01FF80F8**	RxTimeout Elapsed (Read only)	RxTimeout Enb	TxBufEmpty Or TxFifoThresh (Read only)	TxShfEmpty (Read only)	RxBufFull Or RxFifoThresh (Read only)	TxBufIrqEnb	TxShfIrqEnb	RxBufIrqEnb	Rst. Value X0XXX000b Read Value 30h

Register Description: SerSASIrqCtrl Register

Bit 15-13:   Not Used

Bit 12:   RxFifoOverrun(Receive FIFO data loss)—Read only bit.

This bit goes high when the receive FIFO is completely full, but an extra byte has just been received by the RxShiftReg. This will not cause an interrupt, but the RxFifoThresh should already have done so.

Bit 11:   TxFifoFull(Transmit FIFO full)—Read only bit.

This bit goes high when the transmit FIFO is completely full. Data cannot be written to the transmit FIFO after this point. This will not cause an interrupt. This is helpful when trying to figure out if more bytes can be written to the transmit FIFO.

Bit 10:   TxFifoEmpty(Transmit FIFO empty)—Read only bit.

This bit goes high when the transmit FIFO is completely empty. This will not cause an interrupt, but the TxFifoThresh should have already done so.

Bit 9:   RxFifoFull(Receive FIFO full)—Read only bit.

This bit goes high when the receive FIFO is completely full. New received data received from the RxShiftReg will be thrown away. This will not cause an interrupt, but the RxFifoThresh should have already done so.

Bit 8:   RxFifoEmpty(Receive FIFO empty)—Read only bit.

This bit goes high when the receive FIFO is completely empty. This will not cause an interrupt. This can be helpful in trying to figure out if more data can be read from the receive FIFO.

Bit 7:   RxTimoutElapsed (Receive FIFO Timeout occurred)—Read only bit.

The RxTimoutElapsed shows the status of the Timeout function of the receive FIFO. When the RxTimoutElapsed is set, the Timer has timed out which means that the FIFO has not received any new data for too long of a time, but the FIFO is not full. When this bit is cleared, the receive FIFO is running properly. This will cause an interrupt in conjunction with RxTimeoutEnb

Bit 6:   RxTimeoutEnb (Receive Timeout Interrupt Enable)—Read/Write bit.

When the RxTimerEnb is cleared (default), the RxTimeoutIRQ signal will be disabled. When the RxTimerEnb bit is set, the RxTimeoutIRQ will be enabled.

Bit 5:   TxBufEmpty/TxFifoThresh (Transmit Data Buffer Register Empty or Transmit FIFO Threshhold met)—Read only bit.

The TxBufEmpty bit shows the status of the transmit buffer register when TxFIFO is disabled. The transmit buffer register is empty when this bit is set. The transmit buffer register is when this bit is cleared. When TxFIFO is enabled, the transmit FIFO has met its threshhold limit when this bit is set, and is not yet emptied to its threshhold when this bit is cleared. This bit will cause an interrupt in conjunction with TxBufIRQEnb.

Bit 4:   TxShfEmpty (Transmit Shift Data Register Empty)—Read only bit.

The TxShfEmpty bit shows the status of the transmit shift register. The transmit shift register is empty when this bit is set. The transmit shift register is full when this bit is cleared.

Bit 3:   RxBufFull/RxFifoThresh (Receive Buffer Register Full or Receive FIFO Threshhold met)—Read only bit.

The RxBufFull bit show the status of the receive buffer register (when RxFIFO is disabled). The receive buffer register is empty when this bit is cleared. The receive buffer register is when this bit is set. When RxFIFO is enabled, the receive FIFO has met its threshhold when this bit is set, and is not yet filled to its threshhold limit when this bit is cleared. This bit will cause an interrupt in conjunction with RxBufIRQEnb.

Bit 2:   TxBufIrqEnb (Transmit Buffer IRQ Enable)—Read/Write bit.

When the TxBufIrqEnb bit is cleared (default), the TxBufIRQ signal is disabled. The TxBufIRQ will be activated, when the TxBufIrqEnb control bit and the TxBufEmpty status bit are set.

Bit 1:   TxShfIrqEnb (Transmit Shift Empty IRQ Enable)—Read/Write bit.

When the TxShfIrqEnb bit is cleared (default), the TxShfIRQ signal will be disabled. The TxShfIRQ will be activated, when the TxShfIrqEnb control bit and the TxShfEmpty status bit are set.

Bit 0:   RxBufIrq (Receive Buffer IRQ Enable)—Read/Write bit.

When the RxBufIrqEnb bit is cleared (default), the RxBufIRQ signal will be disabled. The RxBufIRQ will be activated, when the RxBufIrqEnb control bit and the RxBufFull status bit are set.

For monitoring the TxBuffer, TxShift, and RxBuffer Register status, the SASIF has four status bits. They are TxBufEmpty, TxShfEmpty, RxBufFull, and RxTimerElapsed status bits. Following are the conditions when the status bits are set and cleared:

	TxBufEmpty	TxShfEmpty	RxBufFull[1]
Set when	1. the system resets, or 2. the last bit of the TxShift Register is sent to the SASTXD pin	1. the system resets, or 2. the TxBufEmpty status bit is set and the last bit of the TxShift Register is sent to the SASTXD pin.	1. the stop bit is received by the SASRXD pin.
Cleared when	1. the CPU writes to the SASData Register, and 2. the TxShfEmpty status bit is cleared.	1. the CPU writes to the SASData Register, and 2. the TxBufEmpty status bit is set.	1. the system reset, or 2. the CPU reads the SASData Register.

	RxTimerElapsed
Set when	When all of these are true for the time it would take to receive 4 bytes into the Rx buffer. The RxFIFO is not full, No start bit has been received, the SASRxData Register has not been read and written back to.
Cleared when	1. the system resets 2. the CPU writes to the SASRxData Register

For keeping track of the transmit and receive activities, the SASIF has four separate interrupts -- the TxBufIRQ, TxShfIRQ, RxBufIRQ, and RxTimeoutIRQ. Any of these interrupt activate will turn on the IRQSASIF. Following are the conditions when they are set and cleared:

	TxBufIRQ	TxShfIRQ	RxBufIRQ[1]
Activated when	1. the TxBufEmpty status bit is set, and 2. the TxBufIrqEnb control bit is set.	1. the TxShEmpty status bit is set, and 2. the TxShfIrqEnb control bit is set, and	1. the RxBufFull status bit is set, and 2. the RxBufIrqEnb control bit is set.
Deactivated when	1. the TxBufEmpty status bit is cleared, or 2. the TxBufIrqEnb control bit is cleared.	1. the TxShEmpty status bit is cleared, or 2. the TxShfIrqEnb control bit is cleared.	1. the RxBufFull status bit is cleared, or 2. the RxBufIrqEnb control bit is cleared.

	RxTimeoutIRQ
Activated when	1. the RxTimerElapsed status bit is set, and 2. the RxTimerEnb control bit is set.
Deactivated when	1. the RxTimerElapsed status bit is cleared, or 2. the RxTimerEnb control bit is cleared.

Note: The RxBufFull status bit and the RxBufIRQ is valid in the AsyncMode only. During the SyncMode, data receive to the SASIF is synchronous with the transmit data, therefore, the RxBufFull status bit and the RxBufIRQ signal is not used in the SyncMode.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASDiv **01FF80F5**	SASRXD Early 1 - Early	(Not Used)	SASSCLK Divisor Upper Bits (13-8)						Rst. Value 0x000000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASDiv **01FF80F4**	SASSCLK Divisor Lower Bits (7-0)								Rst. Value 00h Read Value 00h

Register Description: This register controls bits of the SASSCLK Divisor.

Bit 15: SASRXDEarly (Receive Data Early)—Read/Write Bit.

> If this bit is set to one (default = 0), the input data will be sampled 1/2 shift clock early from what is illustrated in Figure 15-2. This is only for the sync mode.

Bit 14: Not used

Bit 13-8: DivHi (Divisor High Register)—Read/Write bits.

> The DivHi Register is the higher bits (bit 13-8) of the SASSCLKDivisor Register, which defines the SASSCLK frequency.

Bit 7-0: DivLo (Divisor Low Register)—Read/Write bits.

> The DivLo Register is the lower bits (bit 7-0) of the SASSCLKDivisor Register, which defines the SASSCLK frequency. Writing the DivLo register will cause the SASIF logic to load the DivHi and DivLo to the internal counter. Firmware can transmit data with the right speed immediately after the SASSCLK setup.

The SASSCLK Divisor Register defines the SASSCLK frequency, which depends on the AUXCLK frequency (f_{TstClk}). The SASSCLKDivisor Register has a fourteen bit divisor region. The SASSCLK frequency (f_{SClk}) and Divisor relationship is:

For the sync mode,
$$f_{SClk} = INT(\frac{f_{AUXCLK}}{2 * Divisor + 2})$$

For the async mode,
$$f_{SClk} = INT(\frac{f_{AUXCLK}}{8 * Divisor + 8})$$

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASTxFIFOn				SASIF Tx FIFO Byte (2n + 1)					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASTxFIFOn				SASIF Tx FIFO Byte (2n)					Rst. Value 00h Read Value 00h

Register Description: SASTxFIFO Register

SASTxFIFO0 – Byte 0, 1	0x01FF8600-01
SASTxFIFO1 – Byte 2, 3	0x01FF8602-03
SASTxFIFO2 – Byte 4, 5	0x01FF8604-05
SASTxFIFO3 – Byte 6, 7	0x01FF8606-07
SASTxFIFO4 – Byte 8, 9	0x01FF8608-09
SASTxFIFO5 – Byte A, B	0x01FF860A-0B
SASTxFIFO6 – Byte C, D	0x01FF860C-0D
SASTxFIFO7 – Byte E, F	0x01FF860E-0F

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SASRxFIFOn				SASIF Rx FIFO Byte (2n + 1)					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SASRxFIFOn				SASIF Rx FIFO Byte (2n)					Rst. Value 00h Read Value 00h

Register Description: SASRxFIFO Register (Read Only Register)

SASRxFIFO0 – Byte 0, 1	0x01FF8610-01
SASRxFIFO1 – Byte 2, 3	0x01FF8612-03
SASRxFIFO2 – Byte 4, 5	0x01FF8614-05
SASRxFIFO3 – Byte 6, 7	0x01FF8616-07
SASRxFIFO4 – Byte 8, 9	0x01FF8618-09
SASRxFIFO5 – Byte A, B	0x01FF861A-0B
SASRxFIFO6 – Byte C, D	0x01FF861C-0D
SASRxFIFO7 – Byte E, F	0x01FF861E-0F

15.3 SASIF Timing

15.3.1 SASSCLK Timing

When SASSCLK is on (Figure 15-2), the duty cycle of SASSCLK is always 50%.
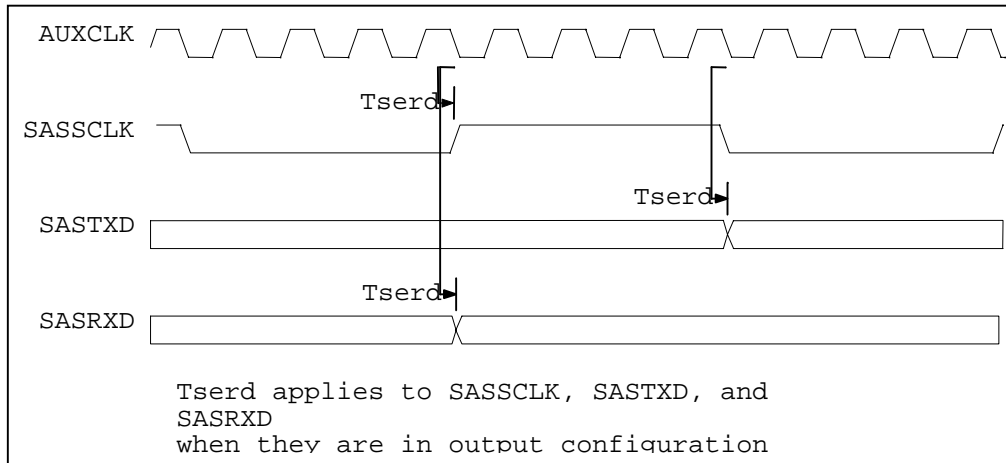
Figure 15-2. SASSCLK Timing Diagram

In Figure 15-2, T_{SERd} is the AUXCLK to SASSCLK, SASRXD and SASTXD delay. TSERd applies to SASSCLK, SASTXD, and SASRXD when they are in output configuration.

The SASSCLK polarity is controlled by the SAS Command Register SASSCLKPol Bit. When the SASSCLKPol bit is cleared (default), the SASTXD signal transition is on the SASSCLK falling edge. External logic can latch the SASTXD signal on the SASSCLK rising edge. When the SASSCLKPol bit is set, the SASTXD signal transition is on the rising edge of the SASSCLK. External logic can latch the SASTXD signal on the SASSCLK falling edge.

When configured in the SyncMode, the SASSCLK will be active for eight cycles for each write to the empty SASData Register. Continuous SASSCLK can be achieved by keeping the transmit buffer register full. When there is no data on the SASTXD pin, the SASSCLK will stay in idle state, which is logic zero as the SASSCLKPol bit is clear and logic one as the SASSCLKPol bit is set. When configured in the AsyncMode, the SASSCLK is a free running signal at the SASSCLKDivisor pre-defined clock rate.

15.3.2 Synchronous Mode Timing

When the SASIF is configured to the SyncMode, the SASIF generates the SASSCLK signal to synchronize data transmit and receive. Therefore, only the transmitter related status bit and IRQ signals are used, and the RxBufFull status bit and the RxBufIRQ signal are not used.

15.3.2.1 FIFOs Disabled

The TxBufEmpty and TxShfEmpty status bits are both set on system reset. When the FIFOs are disabled and both the TxBuffer and TxShift Register are empty, writing to the SASData Register will clear the TxShfEmpty status bit, and will fill the TxShift Register with the TxBuffer Register content.

When the TxBuffer is empty and the TxShift Register is non-empty, writing to the TxBuffer Register will clear the TxBufEmpty status bit, and the TxShift Register is unaffected.

When the TxBuffer is full and the TxShift Register is empty, the TxBuffer Register value will be loaded into the TxShift Register. The TxBufEmpty status bit will be set, and the TxShfEmpty flag is unaffected.

When both the TxBuffer Register and TxShift register are empty, the TxShfEmpty status bit will be set when the last bit of the data is shifted to the SASTXD pin.

15.3.2.2 FIFOs Enabled

When FIFOs are enabled and the transmit FIFO is completely empty and TxShift Register is empty, writing to the SASData Register will clear the TxShfEmpty status bit, keep the TxBufEmpty bit set and will fill the TxShift Register with the contents of the first FIFO location written to.

When the transmit FIFO is completely empty and the TxShift Register is non-empty, writing to the transmit FIFO (accomplished by writing SASData) will clear the TxBufEmpty status bit and the TxShift Register is unaffected (it will stay filled and the TxShfEnpty will stay cleared).

When the transmit FIFO is not empty and the TxShift Register is empty, the value of the oldest valid FIFO location will be loaded into the TxShift Register. The TxBufEmpty status bit will be set only if the last valid FIFO location was just transferred to the TxShift Register. The TxShfEmpty flag is unaffected.

When the transmit FIFO is completely empty and TxShift register are empty, the TxShfEmpty status bit will be set when the last bit of the data is shifted to the SASTXD pin.

15.3.2.3 Transmiting and Sampling

The SASRXD signal is sampled on the falling edge of the SASSCLK when the SASSCLKPol bit is cleared. SASRXD signal is sampled on the rising edge of the SASSCLK when the SASSCLKPol bit is set. If the SASRXDEarly bit is set the SASRXD signal will be sampled 1/2 SASSCLK early.

When the L2MSB control bit is cleared, the MSB to the LSB of the TxShift Register will be shifted to the SASTXD pin, and the SASRXD signal will be shifted into the MSB of the RxShift Register. If the L2MSB bit is set, the LSB to the MSB of the TxShift Register will be shifted to the SASTXD pin, and the SASRXD signal will be shift into the LSB of the RxShift Register. The DataLen bit is not used in the SyncMode, eight data bits will be transmitted or received regardless of setting.
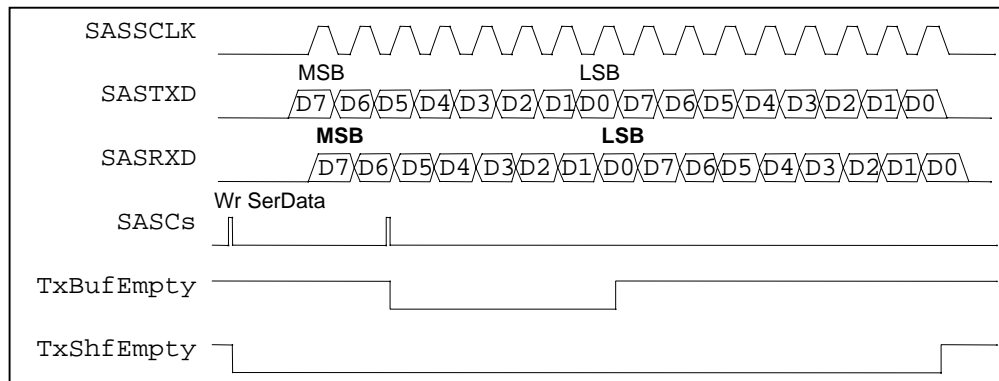
Figure 15-3. Synchronous Mode Timing

15.3.3 Asynchronous Mode Transmitter Timing

Once AsyncMode is set, the transmitter logic will set SASTXD to logic one and will stay in the transmitter idle state. After the ARM writes to the SASData Register, SASTXD will output the Start bit (logic 0) on the next SASSCLK falling edge. Starting from LSB of the TxShift Register (when L2MSB is set), a total of 7 or 8 bits of data (depending on the DataLen bit) will be sent. The bit next to the last data bit is the Stop bit. Similarly to the SyncMode transmit timing, the TxBufEmpty, TxShfEmpty status bits, and the TxBufIRQ, TxShfIRQ has the same timing sequence. See Figure 15-4.
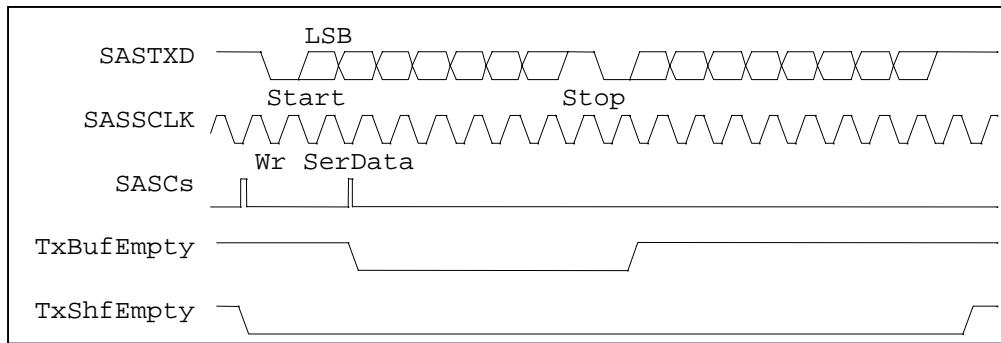
Figure 15-4. Asynchronous Transmitter Timing

The bit capture timing above is identical whether FIFO mode is enabled or disabled. At the end of each transmitted byte, a new byte will be loaded from the TxBuffer Register when FIFO mode is disabled, or from the oldest valid location with the FIFO when FIFO mode is enabled. Following are the timing sequence for the interrupt control and byte loading control for both FIFO enabled and FIFO disabled modes:
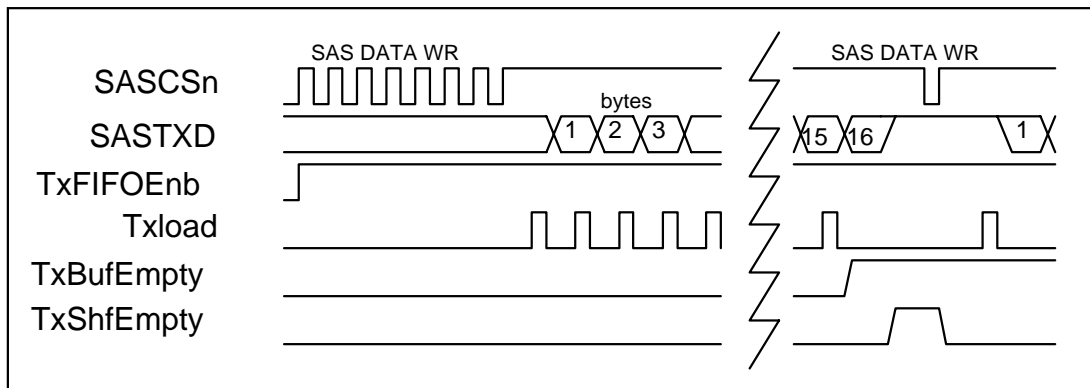
Figure 15-5. Asynchronous Transmitter byte Timing—FIFO Mode Enabled

15.3.3.1   Asynchronous Mode Receiver Timing

Once AsyncMode is set, the receiver logic will look for a falling edge in the SASRXD signal and will stay in the receiver idle state. While SASRXD stays high, the receiver idles. When the Start bit comes, the SASIF EdgeDetector is triggered by the falling edge. Until the next internal 8X baud rate clock arrives, the beginning of this character is defined. Each bit has 8 8X baud rate clock pulses in it. Data is sampled on the fourth Baud8X clock from the receiving bit boundary. See Figure 15-6.

If the sampling value of the Start bit is a logic 1, that start bit is regarded as a glitch or an invalid character, no RxBufIRQ will be generated.
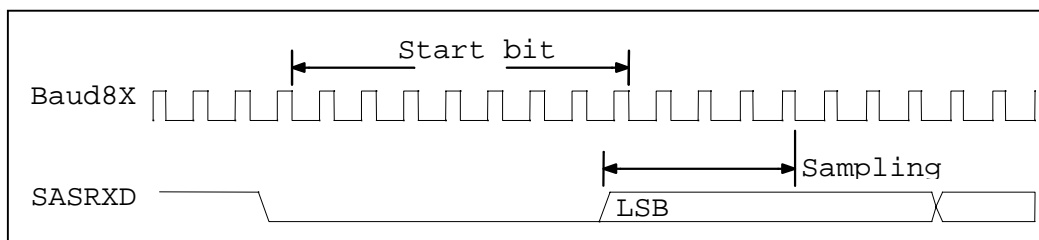
Figure 15-6. Asynchronous Receiver Bit Timing

After receiving the Stop bit, when FIFO mode is disabled, the SERIRQ and the RxBufFull will both rise to a logic 1. RxBufFull and the SERIRQ will be cleared by reading of the SASData Register.
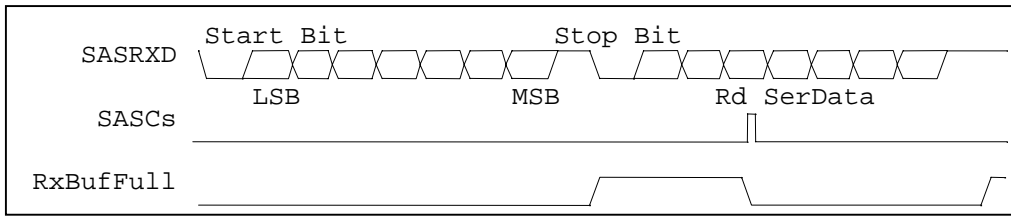
Figure 15-7. Asynchronous Receiver Byte Timing – FIFO mode disabled

When FIFO-mode is enabled, after receiving the stop bit, the SERIRQ and the RxBufFull will change to a logic 1 only if the receive FIFO becomes full as a result of adding that last byte. Both the interrupt and RxBufFull bit will be cleared when the SASData Register is read.
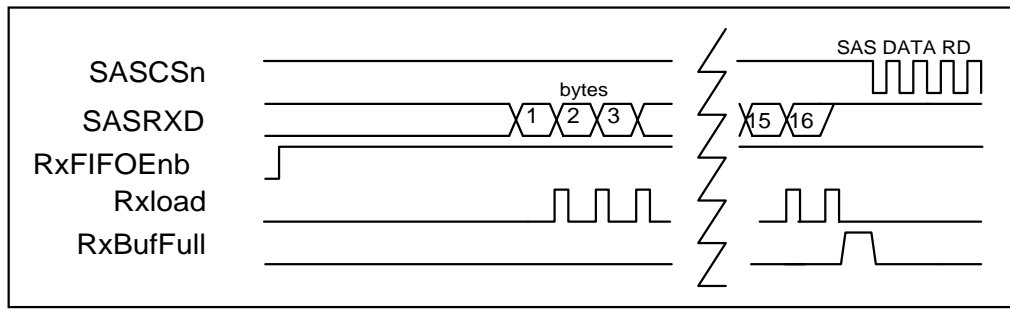
Figure 15-8. Asynchronous Receiver Byte Timing – FIFO mode enabled

The last feature of the FIFO mode is the timeout interrupt. In normal cases, the receive FIFO only causes an interrupt when it is completely full. However, in the case where less than 16 bytes were received and no more bytes are being sent to the SASIF, the timeout function tells the ARM that no data has been received within the required amount of time, so a RxTimeoutIRQ occurs. The time required to cause this interrupt is based on the baud rate and is 4 times the ammount of time that it would take to send one byte of data into the RxShift Register. So if no byte is transferred from the RxShift Register to the receive FIFO, nor is any bit shifted onto the RxShift Register in that time, then the RxTimeoutIRQ will occur. Here is an example of that:
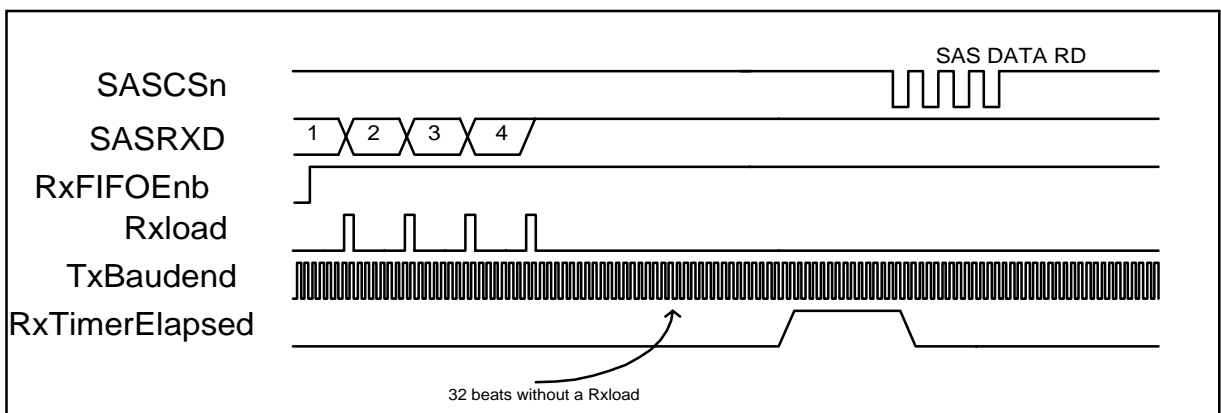
Figure 15-9. Asynchronous Receiver FIFO Timeout—FIFO Mode Enabled

15.4 Firmware Operation

15.4.1 SASIF SyncMode Operation

SyncMode Setup

1. Set the SASSCLKDivisor Register for proper transmission rate.
2. Set the L2MSB bit for proper shifting order.
3. Set the SASSCLKPol bit for proper SASSCLK polarity.
4. Set the transmit FIFO or receiver FIFO to be enabled if lower system overhead or faster transfer rate is required. By default, the two FIFO's are disabled.
5. Clear the TxBufIrqEnb bit and/or the TxShfIrqEnb bit if continuous transmitting is not required (lower CPU interrupt overhead and lower transfer rate), or set the TxBufIrqEnb bit and the TxShfIrqEnb bit if continuous transmission is needed (higher CPU interrupt overhead and higher transfer rate).
6. Set the SASMode bit to select SyncMode operations.

> *Note*: Steps 1 through 5 can be done in a setup-table load.

Data Transmit and Receive Example – FIFO mode disabled

Case when the TxBufIRQ and the TxShfIRQ are enabled:

1. The ARM writes first data to the SASData Register, and sets the TxBufIrqEnb and the TxShfIrqEnb bits.
2. The SASIF loads the TxBuffer Register (SASData Register) to the TxShift Register and clears the TxShfEmpty status bit.
3. The SASIF transmits, and receives bits 7 through bit 0 (if the L2MSB bit is cleared) or bit 0 through bit 7 (if the L2MSB bit is set).
4. By the end of the step 3 data shifting, the ARM gets the TxBufIRQ, which indicates that the TxBuffer is ready for next transmit and the RxBuffer is full.
5. The ARM reads the SASData Register for the RxBuffer Register data, and clears the TxBufIRQ by writing to the TxBuffer Register with the next data. If no more data, the ARM clears the TxBufIrqEnb bit. If only transmit is desired, the ARM can ignore the data in the RxBuffer Register.
6. If the TxBufEmpty status bit is cleared at the end of the data shifting, the SASIF will set the TxBufEmpty bit and goto step 2. However, if the TxBufEmpty bit is set, the SASIF will set the TxShfEmpty status bit and generates the TxShfIRQ.
7. The ARM gets the TxShfIRQ, which indicates the end of all data shifting. If no data is wished to be sent or received, the ARM can then clear the TxShfIrqEnb bit.

Data Transmit and Receive Example – FIFO mode (both FIFO's enabled)

Case when the TxBufIRQ and the TxShfIRQ are enabled:

> *Note*: When using FIFO mode in synchonous mode, both FIFO's must be enabled. This ensures that the RxFIFO will contain the same number of bytes of data by the time the TxBufIRQ occurs that were originally written to the TxFIFO. Otherwise, since only Tx interrupts are used, it will be impossible to grab the received data if no received FIFO is used, and data will be lost if no TxFIFO is used when an RxFIFO is used.

1. The ARM writes first data to the SASData Register up to sixteen times in a row, filling up the TxFIFO as much as desired, and sets the TxBufIrqEnb and the TxShfIrqEnb bits.

2. The SASIF loads the oldest unused byte from the Tx FIFO to the TxShift Register and clears the TxShfEmpty status bit. If this is the last byte held within the FIFO, then a TxBufIRQ and TxBufEmpty status bit will be set, which indicates that the Tx FIFO is ready for next data to be loaded into it.
3. The SASIF transmits, and receives bits 7 through bit 0 (if the L2MSB bit is cleared) or bit 0 through bit 7 (if the L2MSB bit is set).
4. By the end of the step 3 data shifting, if the Tx FIFO is empty then the ARM will receive the TxBufIRQ and TxBuffEmpty status bit will be set,
5. If the TxBufEmpty is cleared at the end of the data shifting (meaning there was leftover data in the TxFIFO), the SASIF will set go back to step 2. However, if the TxBufEmpty bit is set, the SASIF will set the TxShfEmpty status bit and generate the TxShfIRQ.
6. The ARM gets the TxShfIRQ, which indicates the end of all data shifting. The ARM reads the SASData Register as many times as it wrote to the SASData Register when filling up the Tx FIFO. After this, the next batch of data can be written to the Tx FIFO by writing up to sixteen times to SASData. This clears the TxBufEmpty bit and the TxShfEmpty bit. If no more data needs to be sent, the ARM clears the TxBufIrqEnb and TxShfIrqEnb bit. If only transmit is desired, the ARM can ignore the data in the Rx FIFO.

15.4.2 SASIF AsyncMode Operations

AsyncMode Setup

1. Set the SASSCLKDivisor Register for proper transmission rate.
2. Set the L2MSB bit for proper shifting order.
3. Set the SASSCLKPol bit for proper SASSCLK polarity.
4. Set the DataLen bit for proper data length.
5. Choose whether to use the transmit fifo or the receive fifo. Either, both, or neither can be used.
6. Clear the TxBufIrqEnb bit, the TxShfIrqEnb, and the RxBufEnb bit for lower CPU interrupt overhead and lower transfer rate, or set the TxBufIrqEnb bit, the TxShfIrqEnb, and the RxBufEnb bit for higher CPU interrupt overhead and higher transfer rate.
7. Set the SASMode bit to select AsyncMode operations.

> *Note*: Steps 1 through 7 can be done in a setup-table load.

Asynchronous Transmitter – FIFO mode disabled

Case when the TxBufIRQ and the TxShfIRQ are enabled:

1. The CPU writes data to the SASData Register (TxBuffer Register). This data is written at that time to the TxBuffer Register.
2. This data will automatically be transferred to the TxShift Register.
3. The SASIF automatically transmits the start bit.
4. The SASIF automatically clears the TxShfEmpty status bit in the SerCmd Register or the TxBufEmpty bit will be cleared if the TxShfEmpty was already cleared.
5. The SASIF automatically sends bits 0 through bit 6 (if 7 bit data is selected) or bit 7 (if 8 bit data is selected).
6. The SASIF transmits the stop bit, sets the TxBufEmpty bit only if the TxBuffer Register contains data (which will soon be transferred to the TxShift Register), or sets the TxBufEmpty and the TxShfEmpty bit if the TxBuffer Register is empty. If the TxBufEmpty status bit is set and the TxBufEnb is set, the the ARM will receive a TxBufIRQ. If the TxShfEmpty status bit is set and the TxShfEnb is set, the the ARM will receive a TxShfIRQ.
7. Upon receiving these interrupts, the TxShfEnb or TxBufEnb can be cleared if no more data is wished to be sent, or a new byte of data can be written to the SASData Register. In the latter case, the TxShfIRQ will be cleared and steps 2 through 6 will be repeated.

Asynchronous Transmitter – FIFO mode disabled

Case when the TxBufIRQ and the TxShfIRQ are enabled:

1. The CPU writes up to 16 bytes in a row of data to the SASData Register (TxBuffer Register). This data will be written to the most current empty locations within the transmit fifo.
2. Upon first writing to the transmit fifo, the data will automatically be transferred to the TxShift Register if the TxShift Register is currently empty.
3. The SASIF automatically transmits the start bit.
4. The SASIF automatically clears the TxShfEmpty status bit in the SerCmd Register.
5. SASIF automatically sends bits 0 through bit 6 (if 7 bit data is selected) or bit 7 (if 8 bit data is selected).
6. The SASIF transmits the stop bit, sets the TxBufEmpty bit only if the Tx FIFO is out of data (or is transfering its last byte to the TxShift Register), or sets the TxBufEmpty and the TxShfEmpty bit if the Tx FIFO is empty and has no data to give to the TxShift Register. If the TxBufEmpty status bit is set and the TxBufEnb is set, the the ARM will receive a TxBufIRQ. If the TxShfEmpty status bit is set and the TxShfEnb is set, the the ARM will receive a TxShfIRQ.
7. Upon receiving these interrupts, the TxShfEnb or TxBufEnb can be cleared if no more data is wished to be sent, or steps 1 through 6 can be repeated.

Asynchronous Receiver – FIFO mode disabled

1. The receiver is in idle state (SASRXD high) when the SASIF receiver is waiting for falling a edge of the SASRXD (start bit)
2. The start bit activates the SASIF receiver.
3. After receiving 7 or 8 data bits, SASIF receiver looks for stop bit.
4. After receiving the stop bit, the SASIF receiver will set the RxBufFull bit, and will generate a RxBufIRQ if the RxBufEnb interrupt enable bit is set.
5. This serially received data is automatically transferred to the RxBuffer Register.
6. An ARM Read of the SASData Register will retreive the data from the Rx Buffer Register and will clear the RxBufFull status bit and the RxBufIRQ.

Asynchronous Receiver – FIFO mode enabled

1. The receiver is in idle state (SASRXD high) when the SASIF receiver is waiting for falling a edge of the SASRXD (start bit)
2. The start bit activates the SASIF receiver.
3. After receiving 7 or 8 data bits, SASIF receiver looks for stop bit.
4. After receiving the stop bit, the SASIF receiver will set the RxBufFull bit, and will generate a RxBufIRQ if the RxBufEnb interrupt enable bit is set.
5. This serially received data is automatically transferred to the current open location in the Rx FIFO. If the addition of this data makes the Rx FIFO full, then the RxBufIRQ will be activated (if the RxBufEnb is set). When an interrupt is received, 16 bytes have been stored in the Rx FIFO.
6. An ARM Read of the Rx FIFO Register (SASData Register) will clear the RxBufFull status bit and the RxBufIRQ, however it is advised that 16 consecutives reads take place at this time to fully empty out the Rx FIFO.
7. In the case that less than 16 bytes reside in the Rx FIFO but no more will be sent, if the RxTimerEnb is set, then after the Rx Timer times out, then a RxTimeoutIRQ will be sent to the ARM.

> *Note*: Each mention in the above descritions of RxTimeoutIRQ, RxBufIRQ, TxShfEmpty, and TxBufEmpty are actually all received by the ARM as a single interrupt: SERIRQ. In order to figure out which interrupt was received, the SASIrqStatus Register must be read, and then the appropriate action be taken.

16. USB INTERFACE

16.1 Function Description

The USBIF interface block interfaces to the PC, IPB bus and the DMA block. It bridges print and scan data between the MFC2000 and the PC. It is compliant with USB protocol revision 1.1 and supports device remote-wakeup feature. There are six endpoints in the MFC2000, two control endpoints (including endpoint 0), two BULK IN endpoints and two BULK OUT endpoints. The MaxPacketSize for both the control endpoints are 8 byte and the MaxPacketSize for the four bulk transfer pipes are 64 byte. Endpoint 0 supports the Get_Descriptor and USB Vendor/Class commands. The USBIF contains in-silicon USB device controller core, four uni-directional FIFOs and two bi-directional buffers. Description of USB device controller core can be found in UDC specification document.

16.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Fifo1 (R/W) $01FF8581				EP1 Bulkin Half Word 1					Reset. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP1Fifo1 (R/W) $01FF8580				EP1 Bulkin Half Word 1					Reset. Value 00h

Bits 15-0:   Endpoint 1 FIFO Quad Half word 1

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Fifo2 (R/W) $01FF8583				EP1 Bulkin FIFO 2					Reset. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP1Fifo2 (R/W) $01FF8582				EP1 Bulkin Half Word 2					Reset. Value 00h Read Value 00h

Bits 15-0:   Endpoint 1 FIFO Quad Half word 2

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Fifo3 (R/W) $01FF8585				EP1 Bulkin Half Word 3					Reset. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP1Fifo3 (R/W) $01FF8584				EP1 Bulkin Half Word 3					Reset. Value 00h Read Value 00h

Bits 15-0:   Endpoint 1 FIFO Quad Half word 3

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Fifo4 (R/W) $01FF8587				EP1 Bulkin Half Word 4					Reset. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
USBEP1Fifo4 (R/W) $01FF8586				EP1 Bulkin Half Word 4					Reset. Value 00h Read Value 00h

Bits 15-0:   Endpoint 1 FIFO Quad Half word 4

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Hold (R/W) $01FF8589				EP1 Bulkin Holding Register					Reset. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
USBEP1Hold (R/W) $01FF8588				EP1 Bulkin Holding Register					Reset. Value 00h Read Value 00h

Bits 15-0:   Endpoint 1 FIFO Quad Holding Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Ctrl (R/W) $01FF858B	Fifo Enabled (Read Only)	Data Request (Read Only)	Fifo Ready	Fifo DMA Threshold			Fifo Output Pointer		Rst. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
USBEP1Ctrl (R/W) $01FF858A	unused	Holding Reg Full	Byte Present	Fifo Data Quantity			Fifo Input Pointer		Rst. Value x0000000b Read Value 00h

Bit 15:   This bit indicates that the FIFO is active and capable of generating DMA requests. This bit generally follows the setting of the LCL_ENB input signal except the FIFO Enabled register bit will remain asserted if the LCL_ENB input signal is set to false while the FIFO is executing a DMA transfer. In this case, the register bit will remain set for the duration of the DMA transfer and then become cleared.

Bit 14:   This bit is similar to the DMA_REQ signal except that it is not blocked when DMA_TC0 is set. It indicates that the programmed threshold has been met or exceeded and that the FIFO requires data transfers either by enabling hardware DMA through DMA_TC0 or by executing software DMA cycles.

Bit 13:   This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.

Bit 12-10:   This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 1 to 4. A value of zero causes no data transfers to occur. Values greater than 4 are treated as 4.

Bit 9-8:   This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.

Bit 7:   Not used

Bit 6:   This bit indicates that there is a full word in the holding register.

Bit 5:   This bit indicates that there is a single byte in the holding register.

Bit 4-2:   This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.

Bit 1-0:   This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Data (R/W) $01FF858D				EP1 Bulkin Dataport Register					Reset Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP1Data (R/W) $01FF858C				EP1 Bulkin Dataport Register					Reset Value 00h Read Value 00h

Bits 15-0:   Endpoint 1 FIFO Dataport Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP1Tran (W) $01FF858F				EP1 Transition Register					Reset. Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP1Tran (W) $01FF858E				EP1 Transition Register					Reset. Value xxh

Bit 15-0:   Not used

Writing any value to this register causes a push or pop of data through the FIFO data port and the FIFO Quad Halfword 1-4 registers. The FIFO operation depends on the direction of data flow. If system bus (CPU or DMA) is writing (filling) the FIFO, data is transferred from the FIFO data port register to the FIFO Quad Halfword 1 register, simultaneously with data transfer from Quad Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3, and Quad Halfword 3 to Quad Halfword 4, with the original data of Quad Halfword 4 being lost. If the system bus is reading (emptying) the FIFO, data is transferred from the FIFO Quad Halfword 1 register to the FIFO data port register, simultaneously with data transfer from Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the contents of Quad Halfword 4 becoming indeterminate.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Fifo1 (R/W) $01FF85A1				EP2 Bulkout Half Word 1					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Fifo1 (R/W) $01FF85A0				EP2 Bulkout Half Word 1					Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 2 FIFO Quad Half word 1

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Fifo2 (R/W) $01FF85A3				EP2 Bulkout Half Word 2					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Fifo2 (R/W) $01FF85A2				EP2 Bulkout Half Word 2					Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 2 FIFO Quad Half word 2

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Fifo3 (R/W) $01FF85A5				EP2 Bulkout Half Word 3					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Fifo3 (R/W) $01FF85A4				EP2 Bulkout Half Word 3					Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 2 FIFO Quad Half word 3

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Fifo4 (R/W) $01FF85A7				EP2 Bulkout Half Word 4					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Fifo4 (R/W) $01FF85A6				EP2 Bulkout Half Word 4					Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 2 FIFO Quad Half word 4

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Hold (R/W) $01FF85A9				EP2 Bulkout Holding Register					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Hold (R/W) $01FF85A8				EP2 Bulkout Holding Register					Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 2 FIFO Holding Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Ctrl (R/W) $01FF85AB	Fifo Enabled (Read Only)	Data Request (Read Only)	Fifo Ready	Fifo DMA Threshold			Fifo Output Pointer		Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Ctrl (R/W) $01FF85AA	unused	Holding Reg Full	Byte Present	Fifo Data Quantity			Fifo Input Pointer		Rst. Value x0000000b Read Value 00h

Bit 15: This bit indicates that the FIFO is active and capable of generating DMA requests. This bit generally follows the setting of the LCL_ENB input signal except the FIFO Enabled register bit will remain asserted if the LCL_ENB input signal is set to false while the FIFO is executing a DMA transfer. In this case, the register bit will remain set for the duration of the DMA transfer and then become cleared.

Bit 14: This bit is similar to the DMA_REQ signal except that it is not blocked when DMA_TC0 is set. It indicates that the programmed threshold has been met or exceeded and that the FIFO requires data transfers either by enabling hardware DMA through DMA_TC0 or by executing software DMA cycles.

Bit 13: This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.

Bit 12-10: This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 1 to 4. A value of zero causes no data transfers to occur. Values greater than 4 are treated as 4.

Bit 9-8: This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.

Bit 7: Not used

Bit 6: This bit indicates that there is a full word in the holding register.

Bit 5: This bit indicates that there is a single byte in the holding register.

Bit 4-2: This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.

Bit 1-0: This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Data (R/W) $01FF85AD				EP2 Bulkout Dataport Register					Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Data (R/W) $01FF85AC				EP2 Bulkout Dataport Register					Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 2 FIFO Data Port Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP2Tran (W) $01FF85AF				Endpoint 2 Tran					Rst. Value xxh
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP2Tran (W) $01FF85AE				Endpoint 2 Tran					Rst. Value xxh

Bit 15-0:   Not used   Writing any value to this register causes a push or pop of data through the FIFO data port and the FIFO Quad Halfword 1-4 registers. The FIFO operation depends on the direction of data flow. If system bus (CPU or DMA) is writing (filling) the FIFO, data is transferred from the FIFO data port register to the FIFO Quad Halfword 1 register, simultaneously with data transfer from Quad Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3, and Quad Halfword 3 to Quad Halfword 4, with the original data of Quad Halfword 4 being lost. If the system bus is reading (emptying) the FIFO, data is transferred from the FIFO Quad Halfword 1 register to the FIFO data port register, simultaneously with data transfer from Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the contents of Quad Halfword 4 becoming indeterminate.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Fifo1 (R/W) $01FF85B1	EP3 bulk-in Half Word 1								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Fifo1 (R/W) $01FF85B0	EP3 bulk-in Half Word 1								Rst. Value 00h Read Value 00h

Bits 15-0: Endpoint 3 FIFO Quad Half word 1

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Fifo2 (R/W) $01FF85B3	EP3 Bulk-in Half Word 2								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Fifo2 (R/W) $01FF85B2	EP3 Bulk-in Half Word 2								Rst. Value 00h Read Value 00h

Bits 15-0: Endpoint 3 FIFO Quad Half word 2

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Fifo3 (R/W) $01FF85B5	EP3 Bulk-in Half Word 3								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Fifo3 (R/W) $01FF85B4	EP3 Bulk-in Half Word 3								Rst. Value 00h Read Value 00h

Bits 15-0: Endpoint 3 FIFO Quad Half word 3

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Fifo4 (R/W) $01FF85B7	EP3 Bulk-in Half Word 4								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Fifo4 (R/W) $01FF85B6	EP3 Bulk-in Half Word 4								Rst. Value 00h Read Value 00h

Bits 15-0:   IsocIn FIFO Quad Half word 4

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Hold (R/W) $01FF85B9	EP3 bulk-in Holding Register								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Hold (R/W) $01FF85B8	EP3 bulk-in Holding Register								Rst. Value 00h Read Value 00h

Bits 15-0:   Endpoint 3 FIFO Holding Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Ctrl (R/W) $01FF85BB	Fifo Enabled (Read Only)	Data Request (Read Only)	Fifo Ready	Fifo DMA Threshold			Fifo Output Pointer		Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Ctrl (R/W) $01FF85BA	unused	Holding Reg Full	Byte Present	Fifo Data Quantity			Fifo Input Pointer		Rst. Value x0000000b Read Value 00h

Bit 15:   This bit indicates that the FIFO is active and capable of generating DMA requests. This bit generally follows the setting of the LCL_ENB input signal except the FIFO Enabled register bit will remain asserted if the LCL_ENB input signal is set to false while the FIFO is executing a DMA transfer. In this case, the register bit will remain set for the duration of the DMA transfer and then become cleared.

Bit 14:   This bit is similar to the DMA_REQ signal except that it is not blocked when DMA_TC0 is set. It indicates that the programmed threshold has been met or exceeded and that the FIFO requires data transfers either by enabling hardware DMA through DMA_TC0 or by executing software DMA cycles.

Bit 13:	This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.
Bit 12-10:	This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 1 to 4. A value of zero causes no data transfers to occur. Values greater than 4 are treated as 4.
Bit 9-8:	This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.
Bit 7:	Not used
Bit 6:	This bit indicates that there is a full word in the holding register.
Bit 5:	This bit indicates that there is a single byte in the holding register.
Bit 4-2:	This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.
Bit 1-0:	This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Data (R/W) $01FF85BD	EP3 Dataport Register								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Data (R/W) $01FF85BC	EP3 Dataport Register								Rst. Value 00h Read Value 00h

Bits 15-0:	Endpoint 3 FIFO Data Port Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP3Tran (W) $01FF85BF	— Ep3 Transition Register								
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP3Tran (W) $01FF85BE	Ep3 Transition Register								

Bit 15-0:   Not used

Writing any value to this register causes a push or pop of data through the FIFO data port and the FIFO Quad Halfword 1-4 registers. The FIFO operation depends on the direction of data flow. If system bus (CPU or DMA) is writing (filling) the FIFO, data is transferred from the FIFO data port register to the FIFO Quad Halfword 1 register, simultaneously with data transfer from Quad Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3, and Quad Halfword 3 to Quad Halfword 4, with the original data of Quad Halfword 4 being lost. If the system bus is reading (emptying) the FIFO, data is transferred from the FIFO Quad Halfword 1 register to the FIFO data port register, simultaneously with data transfer from Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the contents of Quad Halfword 4 becoming indeterminate.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Fifo1 (R) $01FF85C1	EP4 Fifo Half Word 1								Reset. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Fifo1 (R) $01FF85C0	EP4 Fifo Half Word 1								Reset. Value 00h Read Value 00h

Bits 15-0:   Endpoint 4 FIFO Quad Half word 1

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Fifo2 (R) $01FF85C3				EP4 Fifo Half Word 2					Reset. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Fifo2 (R) $01FF85C2				EP4 Fifo Half Word 2					Reset. Value 00h Read Value 00h

Bits 15-0:	Endpoint 4 FIFO Quad Half word 2

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Fifo3 (R) $01FF85C5				EP4 Fifo Half Word 3					Reset Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Fifo3 (R) $01FF85C4				EP4 Fifo Half Word 3					Reset. Value 00h Read Value 00h

Bits 15-0:	Endpoint 4 FIFO Quad Half word 3

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Fifo4 (R) $01FF85C7				EP4 Fifo Half Word 4					Reset Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Fifo4 (R) $01FF85C6				EP4 Fifo Half Word 4					Reset Value 00h Read Value 00h

Bits 15-0:	Endpoint 4 FIFO Quad Half word 4

Conexant

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Hold (R) $01FF85C9	EP4 Fifo Holding Register								Reset Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Hold (R) $01FF85C8	EP4 Fifo Holding Register								Reset Value 00h Read Value 00h

Bits 15-0: Endpoint FIFO Holding Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Ctrl (R/W) $01FF85CB	Fifo Enabled (Read Only)	Data Request (Read Only)	Fifo Ready	Fifo DMA Threshold			Fifo Output Pointer		Reset Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Ctrl (R/W) $01FF85CA	unused	Holding Register Full	Byte Present	Fifo Data Quantity			Fifo Input Pointer		Reset Value 00h Read Value 00h

Bit 15: This bit indicates that the FIFO is active and capable of generating DMA requests. This bit generally follows the setting of the LCL_ENB input signal except the FIFO Enabled register bit will remain asserted if the LCL_ENB input signal is set to false while the FIFO is executing a DMA transfer. In this case, the register bit will remain set for the duration of the DMA transfer and then become cleared.

Bit 14: This bit is similar to the DMA_REQ signal except that it is not blocked when DMA_TC0 is set. It indicates that the programmed threshold has been met or exceeded and that the FIFO requires data transfers either by enabling hardware DMA through DMA_TC0 or by executing software DMA cycles.

Bit 13: This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.

Bit 12-10: This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 1 to 4. A value of zero causes no data transfers to occur. Values greater than 4 are treated as 4.

Bit 9-8: This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.

Bit 7: Not used

Bit 6: This bit indicates that there is a full word in the holding register.

Bit 5:   This bit indicates that there is a single byte in the holding register.

Bit 4-2:   This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.

Bit 1-0:   This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Data (R/W) $01FF85CD	EP4 Endpoint Dataport Register								Reset Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Data (R/W) $01FF85CC	EP4 Endpoint Dataport Register								Reset Value 00h Read Value 00h

Bits 15-0:   Endpoint 4 FIFO Data Port Register

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP4Tran (W) $01FF85CF	EP4 Transition Register								
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP4Tran (W) $01FF85CE	EP4 Transition Register								

Bit 15-0:   Not used   Writing any value to this register causes a push or pop of data through the FIFO data port and the FIFO Quad Halfword 1-4 registers. The FIFO operation depends on the direction of data flow. If system bus (CPU or DMA) is writing (filling) the FIFO, data is transferred from the FIFO data port register to the FIFO Quad Halfword 1 register, simultaneously with data transfer from Quad Halfword 1 to Quad Halfword 2, Quad Halfword 2 to Quad Halfword 3, and Quad Halfword 3 to Quad Halfword 4, with the original data of Quad Halfword 4 being lost. If the system bus is reading (emptying) the FIFO, data is transferred from the FIFO Quad Halfword 1 register to the FIFO data port register, simultaneously with data transfer from Quad Halfword 2 to Quad Halfword 1, Quad Halfword 3 to Quad Halfword 2, and Quad Halfword 4 to Quad Halfword 3, with the contents of Quad Halfword 4 becoming indeterminate.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0Buf2 (R/W) $01FF85D1	EP0 Buffer 2								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0Buf1 (R/W) $01FF85D0	EP0 Buffer 1								Rst. Value 00h

Bits 15-8:   2nd data byte of endpoint 0 buffer

Bits 7-0:   1st data byte of endpoint 0 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0Buf4 (R/W) $01FF85D3	EP0 Buffer 4								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0Buf3 (R/W) $01FF85D2	EP0 Buffer 3								Rst. Value 00h

Bits 15-8:   4th data byte of endpoint 0 buffer

Bits 7- 0:   3rd data byte of endpoint 0 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0Buf6 (R/W) $01FF85D5	EP0 Buffer 6								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0Buf 5 (R/W) $01FF85D4	EP0 Buffer 5								Rst. Value 00h

Bits 15-8:   6th data byte of endpoint 0 buffer

Bits 7- 0:   5th data byte of endpoint 0 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0Buf8 (R/W) $01FF85D7	EP0 Buffer 8								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0Buf7 (R/W) $01FF85D6	EP0 Buffer 7								Rst. Value 00h

Bits 15-8:   8th data byte of endpoint 0 buffer

Bits 7- 0:   7th data byte of endpoint 0 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5Buf2 (R/W) $01FF85D9	EP5 Buffer 2								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5Buf1 (R/W) $01FF85D8	EP5 Buffer 1								Rst. Value 00h

Bits 15-8: 2nd data byte of endpoint 5 buffer

Bits 7- 0: 1st data byte of endpoint 5 buffer

address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5Buf4 (R/W) $01FF85DB	EP5 Buffer 4								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5Buf3 (R/W) $01FF85DA	EP5 Buffer 3								Rst. Value 00h

Bits 15-8: 4th byte of the endpoint 5 buffer

Bits 7-0: 3rd byte of the endpoint 5 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5Buf6 (R/W) $01FF85DD	EP5 Buffer 6								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5Buf5 (R/W) $01FF85DC	EP5 Buffer 5								Rst. Value 00h

Bits 15-8: 6th byte of the endpoint 5 buffer

Bits 7-0: 5th byte of the endpoint 5 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5Buf8 (R/W) $01FF85DF	EP5 Buffer 8								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5Buf7 (R/W) $01FF85DE	EP5 Buffer 7								Rst. Value 00h

Bits 15-8: 8th byte of the endpoint 5 buffer

Bits 7-0: 7th byte of the endpoint 5 buffer

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5StDat2 (R) $01FF85E1	EP5 Setup Packet Data 2nd byte								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5StDat1 (R) $01FF85E0	EP5 Setup Packet Data 1st byte								Rst. Value 00h Read Value 00h

Bits 15-8: 2nd data byte of the setup packet for EP5

Bits 7:0: 1st data byte of the setup packet for EP5

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5StDat4 (R) $01FF85E3	EP5 Setup Packet Data 4th byte								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5StDat3 (R) $01FF85E2	EP5 Setup Packet Data 3rd byte								Rst. Value 00h Read Value 00h

Bits 15-8:   4th data byte of the setup packet for EP5

Bits 7:0:   3rd data byte of the setup packet for EP5

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5StDat6 (R) $01FF85E5	EP5 Setup Packet Data 6th byte								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5StDat5 (R) $01FF85E4	EP5 Setup Packet Data 5th byte								Rst. Value 00h Read Value 00h

Bits 15-8:   6th data byte of the setup packet for EP5

Bits 7:0:   5th data byte of the setup packet for EP5

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP5StDat8 (R) $01FF85E7	EP5 Setup Packet Data 8th byte								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP5StDat7 (R) $01FF85E6	EP5 Setup Packet Data 7th byte								Rst. Value 00h Read Value 00h

Bits 15-8:   8th data byte of the setup packet for EP5

Bits 7:0:   7th data byte of the setup packet for EP5

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBIRQ (R) $01FF85E9	Ep4 Interrupt	Ep2 Interrupt	Suspend status	EP5 Write Ack	EP5 Read Ack	EP0 Write Ack	EP0 Read Ack	USB Reset Interrupt	Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBIRQ (R) $01FF85E8	Suspend Interrupt	DMA Interrupt	EP5 Write Interrupt	EP5 Read Interrupt	EP5 Setup Interrupt	EP0 Write Interrupt	EP0 Read Interrupt	EP0 Setup Interrupt	Rst. Value 00h

Bit 15:	This bit indicates that an odd packet is received at endpoint 4.
Bit 14:	This bit indicates that an odd packet is received at endpoint 2.
Bit 13:	When it is "1", it indicates that the device is in suspend mode. If the PC does a global wake up, this bit will be reset to "0". The firmware can reset this bit to "0" by performing a device remote wakeup (setting the resume bit to be "1" in USBClrIRQ register).
Bit 12:	This bit indicates that if the control write on endpoint 5 is successful. "1" – Ack. "0" – Nack.
Bit 11:	This bit indicates that if the control read on endpoint 5 is successful. "1" – Ack. "0" – Nack.
Bit 10:	This bit indicates that if the control write on endpoint 0 is successful. "1" – Ack. "0" – Nack.
Bit 9:	This bit indicates that if the control read on endpoint 0 is successful. "1" – Ack. "0" – Nack.
Bit 8:	This bit indicates that a USB reset is detected on the bus.
Bit 7:	This bit indicates a suspend is detected on the bus.
Bit 6:	This bit indicates the DMA has reached its block size.
Bit 5:	This bit indicates that the endpoint 5 data buffer is full.
Bit 4:	This bit indicates that the data in endpoint 5 buffer has been send.
Bit 3:	This bit indicates that a set up packet has been send to endpoint 5 and the endpoint 5 setup data buffer is full.
Bit 2:	This bit indicates that the endpoint 0 data buffer is full.
Bit 1:	This bit indicates that the data in endpoint 0 buffer has been send.
Bit 0:	This bit indicates that a set up packet has been send to endpoint 5 and the endpoint 5 setup data buffer is full.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBClrIRQ (R/W) $01FF85EB	unused	Clear EP4 Interrupt	Clear EP2 Interrupt	Clear USB Reset Interrupt	EP5 Write Buffer Read	EP5 Read Buffer Written	EP0 Write Buffer Read	EP0 Read Buffer Written	Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBClrIRQ (R/W) $01FF85EA	Clear Suspend Interrupt	Resume Device	Clear EP5 Write Interrupt	Clear EP5 Read Interrupt	Clear EP5 Setup Interrupt	Clear EP0 Write Interrupt	Clear EP0 Read Interrupt	Clear EP0 Setup Interrupt	Rst. Value 00h

Bit 15: Unused

Bit 14: Set this bit to "1" to clear the endpoint 4 interrupt.

Bit 13: Set this bit to "1" to clear the endpoint 2 interrupt.

Bit 12: Set this bit to "1" to clear the usb reset interrupt.

Bit 11: Set this bit to "1" to indicate that the endpoint 5 buffer is empty.

Bit 10: Set this bit to "1" to indicate that the endpoint 5 buffer is full.

Bit 9: Set this bit to "1" to indicate that the endpoint 0 buffer is empty.

Bit 8 Set this bit to "1" to indicate that the endpoint 0 buffer is full.

Bit 7: Set this bit to "1" to clear the suspend interrupt.

Bit 6: Set this bit to "1" to perform device remote wake-up.

Bit 5: Set this bit to "1" to clear endpoint 5 write interrupt.

Bit 4: Set this bit to "1" to clear endpoint 5 read interrupt.

Bit 3: Set this bit to "1" to clear endpoint 5 setup interrupt.

Bit 2: Set this bit to "1" to clear endpoint 0 write interrupt.

Bit 1: Set this bit to "1" to clear endpoint 0 read interrupt.

Bit 0: Set this bit to "1" to clear endpoint 0 setup interrupt.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBSofReset (W) $01FF85ED	Unused								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBSofReset (W) $01FF85EC	Unused							Software Reset	Rst. Value 01h

Bit 15:1: Not used

Bit 0: Set this bit to "1" reset usb interface block.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBStall (R/W) $01FF85EF	Unused								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBStall(R/W) $01FF85EE	Unused		EP5 Stalled	EP4 Stalled	EP3 Stalled	EP2 Stalled	EP1 Stalled	EP0 Stalled	Rst. Value 01h

Bit 15:6:   Not used

Bit5:   Set this bit to "1" to indicate that the endpoint 5 is stalled.

Bit4:   Set this bit to "1" to indicate that the endpoint 4 is stalled.

Bit3:   Set this bit to "1" to indicate that the endpoint 3 is stalled.

Bit2:   Set this bit to "1" to indicate that the endpoint 2 is stalled.

Bit1:   Set this bit to "1" to indicate that the endpoint 1 is stalled.

Bit0:   Set this bit to "1" to indicate that the endpoint 0 is stalled.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0StDat2 (R) $01FF85F1	EP0 Setup Packet Data 2nd byte								Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0StDat1 (R) $01FF85F0	EP0 Setup Packet Data 1st byte								Rst. Value 00h

Bits 15-8:   2nd data byte of the setup packet for EP0

Bits 7:0:   1st data byte of the setup packet for EP0

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0StDat4 (R) $01FF85F3	EP0 Setup Packet Data 4th byte								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0StDat3 (R) $01FF85F2	EP0 Setup Packet Data 3rd byte								Rst. Value 00h

Bits 15-8:   4th data byte of the setup packet for EP0

Bits 7:0:   3rd data byte of the setup packet for EP0

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0StDat6 (R) $01FF85F5	EP0 Setup Packet Data 6th byte								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0StDat5 (R) $01FF85F4	EP0 Setup Packet Data 5th byte								Rst. Value 00h

Bits 15-8:   6th data byte of the setup packet for EP0

Bits 7:0:   5th data byte of the setup packet for EP0

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
USBEP0StDat8 (R) $01FF85F7	EP0 Setup Packet Data 8th byte								Rst. Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
USBEP0StDat7 (R) $01FF85F6	EP0 Setup Packet Data 7th byte								Rst. Value 00h

Bits 15-8:   8th data byte of the setup packet for EP0

Bits 7:0:   7th data byte of the setup packet for EP0

16.3 Firmware Operation

16.3.1 Setup Transfer

During the setup transfer, a setup interrupt will be generated to the ARM. ARM can determine it is a setup interrupte by reading the interrupt register and clear the interrupt by setting the clear endpoint 0 or 5 setup interrupt bit of the USBClrIrq register. The 8 bytes of setup packet data can be read from the endpoint 0 or 5 setup buffer.

16.3.1.1 USB Standard Commands

Since the UDC core decodes and acts upon the USB standard commands except Get_Descriptor, the setup packet data of all standard commands (except Get_Descriptor) will not be stored in the setup buffers neither the setup interrupt will be generated during a starndard command transfer.

16.3.1.2 Get_Descriptor Command

The Get_Descriptor command is supported by endpoint 0. The ARM needs to decode the setup packet and load the descriptor data into the endpoint 0 buffer then set the ep0 read buffer written bit.

16.3.2 Control Read Transfer

During a control read transfer, an interrupt will be generated along with read status bit after the data in the ep0/ep5 buffer are send to the host. ARM can determine the type of interrupt by reading the interrupt register and clear the interrupt by setting the corresboing bit in the clear interrupt register. ARM needs to check the ep0/ep5 read status bit to find out if the control read is successful or not.

16.3.3 Control Write Transfer

During a control write transfer, the data from the PC will be written to an internal buffer first. After the control write transfer finished, an interrupt along with write status bit will be generated. ARM can determine the interrupt by reading the interrupt register and clear the interrupt by writing to the corresbonding bit in the clear_interrupt register. Then ARM can read data from the endpoint 0 or endpoint 5 data buffers. After all the data have been read , ARM will set the endpoint 0 or endpoint 5 write_buffer_read bit in the endpoint status register.

16.3.4 BULK_IN Transfer

The MaxPacketSize for both bulk in pipes are 64 bytes. DMA channel 0 is assigned to USB and it has four logic channels. Endpoint 1 and 3, which are bulk in endpoints, are assigned to logic channel 0 and 2. The procedures of programming the registers are:

1. Enable DMA channel 0 by writing 1 to bit[0] of DMA 0 configuration register.

2. Enable logic channel 0 or 2 and set logic channel 0 or 2 to be read mode by setting the corresbonding bits in DMA 0 configuration register.

3. Set memory address in DMAUSB0CntLo and DMAUSB0CntHi or DMAUSB2CntLo and DMAUSB2CntHi.

4. Define block size in DMAUSB0BlkSiz or DMAUSB2BlkSiz.

5. If more than one memory block is needed, repeat step 3 and 4 to set desired memory blocks.

6. Load 4 half words into endpoint 1 or 3 internal fifo by writing to endpoint 1 or 3 data port register and transition register OR by writing to the endpoint 1 or 3 halfword fifos.

7. Set endpoint 1 or 3 control register.

If the PC has read all the data in the memory blocks successfully, an interrupt will be generated. ARM can read the USBIRQ register to determine that it is DMA channel 0 interrupt. By reading DMA 0 configuration register can determine it is a logic channel 0 or 2 interrupt. Then ARM can repeat step 3 and 4 to set more memory blocks.

1.1.516.3.5 BULK_OUT Transfer

The MaxPacketSize for both bulk in pipes are 64 bytes. DMA channel 0 is assigned to USB and it has four logic channels. Endpoint 2 and 4, which are bulk out endpoints, are assigned to logic channel 1 and 3. The procedures of programming the registers are:

1. Enable DMA channel 0 by writing 1 to bit[0] of DMA 0 configuration register.

2. Enable logic channel 1 or 3 and set logic channel 1 or 3 to be write mode by setting the corresbonding bits in DMA 0 configuration register.

3. Set memory address in DMAUSB1CntLo and DMAUSB1CntHi or DMAUSB3CntLo and DMAUSB3CntHi.

4. Define block size in DMAUSB1BlkSiz or DMAUSB3BlkSiz.

5. If more than one memory block is needed, repeat step 3 and 4 to set desired memory blocks.

6. Set endpoint 2 or 4 control register.

If the PC has write to all the memory blocks successfully, an interrupt will be generated. ARM can read the USBIRQ register to determine that it is DMA channel 0 interrupt. By reading DMA 0 configuration register can determine it is a logic channel 1 or 3 interrupt. Then ARM can repeat step 3 and 4 to set more memory blocks.

If the PC send an odd byte count packet in the middle of a bulkout transaction, an interrupt will be generated to indicate that an single byte has been send to the memory location.

1.1.616.3.6 USB Suspend and Device Remote Wake-up

The MFC2000 supports device remote wake-up. If the UDC core detected the USB bus being idle for 3 ms, it will generated a suspend signal and an interrupt will be generated. ARM can determine that it is a suspend interrupt by reading the usb interrupt register then clear the interrupt by setting the clear_suspend_int bit in the clear interrupt register. The suspend status bit will remain high until the PC performs a global wake up or the ARM performs a device remote wake up. The ARM performs a device remote wake up by setting the resume_device bit in the endpoint status register to be "1".

1.1.716.3.7 USB Reset

An interrupt will be generated if a USB reset is detected on the USB bus. ARM can determine that it is a usb_reset_interrupt by reading the usb interrupt register then clear the interrupt by setting the clear_usb_reset_int bit in the clear interrupt register.

1.1.816.3.8 Software Reset

The ARM can reset the UDC core and related logic by setting the bit 0 in USBSofReset register.

17. Bi-directional Parallel Peripheral Interface

The Parallel Peripheral Interface(PPI) controller incorporates the IEEE-1284 Compatibility/Nibble/ Byte/ ECP protocols to offer flexible high-speed parallel data transfer. Furthermore, the PPI fully supports all variants of these modes, including device ID requests and run-length encoded data compression. All protocols are signal-compatible with classic parallel interfaces. The default protocol for the PPI is the Compatibility mode. The PPI contains specific hardware to support automatic handshaking during host to peripheral,or forward, data transfers in compatible and ECP modes, and run-length detection. The PPI also supports automatic handshaking during peripheral to host, or reverse, data transfers in the Nibble, Byte and ECP modes. The hardware handshaking can also be completely disabled to allow software to directly control the peripheral port interface signals and to support new protocols as they arise in the future.

17.1 Operational Modes

The PPI supports three hardware handshaking modes for host to peripheral, or forward, data transfers. Each mode can be enabled or disabled by software. One mode supports forward data transfers during compatibility mode, and the other two modes support forward data transfers during ECP mode. Only one of the three modes can be enabled at a time, or all modes can be disabled. When disabled, software must assume full responsibility for handshaking, and use the Parallel Port Interface Register and the Parallel Port Data Register to read and control the logic levels on all parallel port pins.

The hardware handshaking modes for peripheral to host, or reverse, data transfers are provided for Nibble, Byte and ECP modes explained.

17.1.1 Compatibility Mode

The interface is always initialized to the compatibility Mode, a conventional, unidirectional host-to peripheral interface. From the compatibility Mode the host may transmit data to peripheral or direct the interface to a mode capable of transmitting data to the host. Compatibility mode hardware handshaking is enabled by setting the MODE field to 1 in the Parallel Port Control Register. When compatibility mode is selected, the PPI interacts with host using STROBEn, BUSY and ACKn signals.

17.1.2 P1284 Negotiation

The negotiation phase is performed under firmware control. During the negotiation, the firmware will respond to the host stimulus by setting ACKn low, FAULTn high, SLCTOUT(XFlag) high, PE high as defined in the IEEE 1284 Specification. After the host responds to the prior signal setting, the firmware will respond with a PE low and a FAULTn low if data to the host is available. The SLCTOUT is set to it's appropriate value corresponding to the extensibility feature requested by the host. (Refer to the IEEE 1284 Specification for a more complete definition of this phase).

17.1.3 Nibble Mode

Provides an asynchronous, reverse (peripheral-to-host) channel, under control of the host. Data bytes are transmitted as two sequential, four bit nibbles using four peripheral-to-host status lines. In Nibble Mode(MODE=4), one byte of status data can be put to the PPI data register by CPU or DMA operation. The PPI Logic will separate the one byte data into two nibbles and send them in sequence. The signals used in the Nibble mode are AUTOFDn and ACKn for handshake and FAULTn, SLCTOUT, PE and BUSY for data transfer.

17.1.4 Byte Mode

Provides an asynchronous, byte-wide reverse peripheral-to-host channel using eight data lines of the interface for data and the control/status lines for handshaking. The PPI hardware can control the handshake of the reverse data transfer. The CPU must provide the status to send the next byte or to terminate the protocol. The MODE=5 for the Byte mode which enables a peripheral to send an entire byte of data to the PC in one data transfer cycle using the 8 data lines, rather than the two cycles using the Nibble mode.The handshake signals are AUTOFDn, ACKn and STROBEn.

17.1.5 ECP Forward

Extended Capabilities Port (ECP) Mode provides an asynchronous, byte-wide, bi-directional channel. The data can be received by DMA mode or CPU mode as described in the Compatibility Mode section. The ECP protocol provides the Data and command cycles types in both forward and reverse directions. The MODE = 2 and 3 for the forward mode with RLE . When MODE is set to 2, ECP mode hardware handshaking is enabled during forward data transfers, without RLE support. When mode 2 is programmed, the PPI responds to a high to low transition on STROBEn (HostClk), and automatically sets and clears the BUSY bit in the Parallel Port Interface Register and the BUSY to handshake with the host.. MODE may be reprogrammed at any time, but if an ECP cycle is currently in progress, it completes as normal.

17.1.6 ECP Reverse

After a successful Forward to Reverse phase switch, the ECP changes into the reverse phase in MODE = 6. When the 1284 interface is idle, this is the desired mode of operation. This mode will allow the MFC2000 ASIC to inform the host of incoming fax data. The ECP Reverse mode can operate under *CPU interrupt driven control* or automated DMA control. In CPU control, the software writes the outgoing byte into the output buffer and then sets ACKn low to send the data. After the host completes the data transfer, the ACKn is automatically set high and the CPU is free to respond to any PPI interrupts that have been enabled or to initiate an additional data transfer if needed. The handshake signals are INITn, PE, ACKn, AUTOFDn and BUSY.

17.1.7 The Non-standard Dribble Mode

The dribble mode was used for the slower PC before P1284 is standardized. Dribble mode is similar to ECP in that the channel can be reversed without having to go through a negotiation phase to turn the channel around. It sends two bits at a time on the nPeriphRequest and XFlag signals rather than using the PIO data bus. In this respect, it is similar to nibble mode's data transfer phase in that it uses HostBusy and PtrClk to transfer each portion of the data byte. The dribble mode can only operate under CPU interrupt-driven control (no DMA mode).

> *Note*: This is not an IEEE-1284 mode but is useful to support certain product interfaces created prior to the IEEE-1284 specification.

17.2 Additional Features

17.2.1 Filter

A digital filter is provided for incoming host control and data signals. SLCTINn, STROBEn, AUTOFDn, and INITn are provided with digital filter circuits which are collectively controlled by software. IF DFIL =0, then no digital filtering is selected.

Note that synchronization plus digital filtering adds up to four CLK periods of delay before a level change at one of the host inputs appears in the Parallel Port Control Register, and five periods of delay before an output responds to an input (e.g., before BUSY responds to STROBEn).

Likewise, synchronization and digital filtering of STROBEn affect the point at which ippid[7-0] and AUTOFDn are latched into the parallel Port Data Register. Without a DFIL=3, ippid[7-0] and AUTOFDn are sampled an the fourth rising edge of CLK after STROBEn is sampled. With digital filtering, ippid7-0 and AUTOFDn are sampled on the fifth rising edge of CLK after STROBEn is sampled.

17.3 Functional Description

The PPI contains an interface controller, which consists of: a data receive and transmit latch, run-length encoding decompression logic, input conditioning logic, and edge detector logic. The RLE decompression is accomplished through a state machine and additional control logic. There is input conditioning logic to filter the incoming control signals. The peripheral interface block supports four IEEE 1284 communication modes: compatibility, nibble, byte, and enhanced capabilities port (ECP) mode. The interface with the DMA controller requires storing data into 8 bit bytes before transferring to the four half word FIFO for burst operation. There will be a counter within the FIFO that keeps track of the bytes transferred to the FIFO. There is a control bit allowing FIFO flush to be done. If firmware detects the end of transfer to the PPI, it can flush the FIFO by forcing the flush bit. The timing of the F/W flush is based on a fixed time from the last DMA access and the FIFO not empty status and there will be a busy status bit set by F/W to stop further transmission from the host. The DMA or the interrupt operations are enabled under the control of the F/W. The PPI has fifteen sources of interrupt that can each be individually enabled or disabled and they are all ORed together to generate one PPI IRQ interrupt for the CPU.
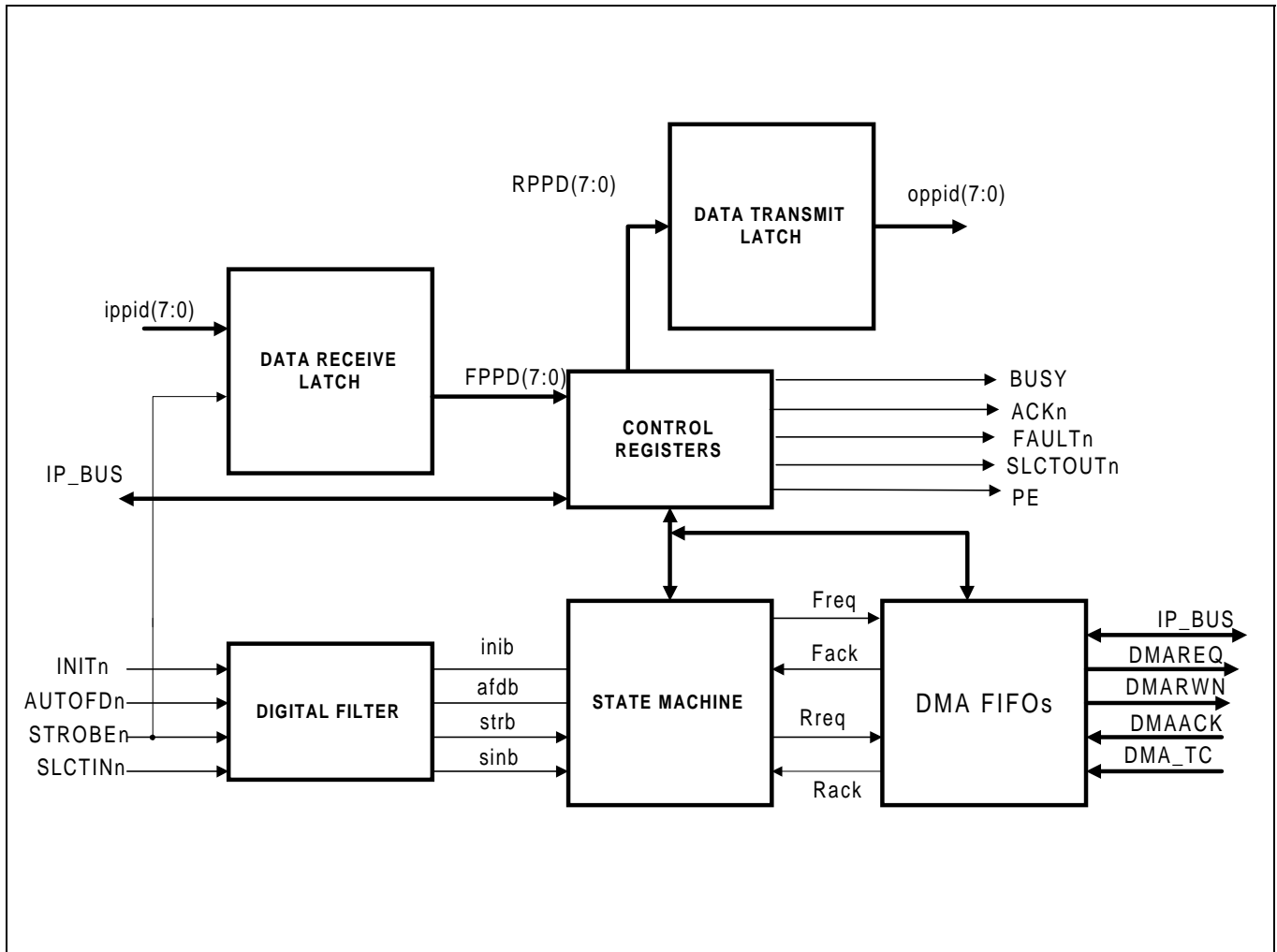

Figure 17-1. Parallel Port Interface Controller Block Diagram

17.4 Register Description

Parallel Port Control Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Control Reg. *(PIOCtrl)* **$01FF8201**	(Not Used)	(Not Used)	TRIZN	PPIRSTN	MODE	MODE	RFULL (R)	FFULL (R)	Rst. Value xx000000b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Parallel Port Control Reg. *(PIOCtrl)* **$01FF8200**	RLD (R)	ABRT	PDOE (R)	ERRC	MODE	MODE	DFIL	DFIL	Rst. Value 00h Read Value 00h

DFIL   Digital Filter   This 2 bit field controls the number of clocks of digital filtering to be applied on the four host control signal inputs, SLCTINn, STROBEn, AUTOFDn, and INITn. From 0 to 3 clocks of digital filtering may be selected.

MODE   Mode   This 4 bit field(bits 11,10,3,2) selects and enables a hardware handshaking mode for forward data transfers.

MODE	Direction	Function
0		Disable
1	Fwd	Compatibility mode
2	Fwd	ECP-fwd without RLE mode
3	Fwd	ECP-fwd with RLE mode
4	Rev	Nibble mode
5	Rev	Byte mode
6	Rev	ECP-rev mode
7	Rev	Dribble mode
8-15		Reserved

        Error Cycles   This bit is used to execute an error cycle when in compatibility mode (mode 1). When set, ERRC immediately forces the BUSY pin high. If ERRC is set when a compatibility mode handshake sequence is in process, the BUSY remains high beyond the end of the cycle. ERRC does not affect an ACKn pulse that is already active, but does prevent an ACKn pulse if it is about to be generated. While ERRC is set, software may set or clear SEL, PERR, and NFLT in the Parallel Port Interface Register. When ERRC is cleared, the PPI generated an ACKn pulse and tries to lower BUSY to conclude the error cycle. If, however, FBSY is set, BUSY remains high until FBSY is cleared. When MODE is set to any value except 1, setting ERRC has no effect. Setting MODE to 1 when ERRC is already set causes the handshake controller to immediately begin an error cycle.

PDOE	Parallel Data Output Enable	This bit performs two functions. It controls the state of the PD bus three-state output drives, and it qualifies the latching of data from the PD bus into the Parallel Port Data Register. When set, PDOE enables the PD bus output pin drivers, and prevents data from being latched into the Parallel Port Register.
		Setting ABRT affects the operation of PDOE. IF ABRT is set, SLCTINn (1284ACTIVE) must remain high to allow PDOE to be set or remain set. If ABRT is set and SLCTINn (1284Active) goes low, PDOE is cleared, and setting PDOE has no effect.
ABRT	Abort	This causes the PPI to use SLCTINn (1284Active) to detect when the host suddenly aborts a reverse transfer and returns to compatibility mode. If ABRT is set, a low level on SLCTINn (1284 Active) causes PDOE to be cleared and the PD bus output drivers to be three-stated. In fact, if ABRT is set and SLCTINn (1284Active) is low, setting PDOE has no effect. This protection logic, as all internal logic, operates from a synchronized and optionally digitally filtered SLCTINn (1284Active) that is latched into the parallel Port Interface Register.
RLD	Run Length Decompression	This is a read-only status bit which indicates when run-length decompression is taking place during ECP forward data transfers. RLD is set when a run-length count is received and loaded into the internal counter, and cleared when the last read of the Parallel Port Data Register takes place. This bit can only be set when ECP with RLE (mode 3) is enabled. If MODE is reprogrammed during a decompression, decompression continues and RLD remains set until the operation is completed. RLD is cleared when RST is issued.
FFULL	Forward Data Register Full	This is a read-only status bit that indicates when parallel port data from the host is latched in the Parallel Port Data Register. FFULL is cleared when the Parallel Port Data Register is read. When handshaking and DMA is enabled, FFULL sets and clears as data is latched and read during forward data transfers. FFULL is also cleared when RST is issued.
RFULL	Reverse Data Register Full	This is a read-only status bits indicates when parallel port data to the host is latched in the Parallel Port Data Register. RFULL is cleared when the Parallel Port Data Register is sent. When handshaking and DMA is enabled, RFULL sets and clears as data is latched and sent during reverse data transfers. RFULL is also cleared when RST is issued.
PPIRSTN	Parallel port software reset	Active low resets all the state machines and is programmable using this register bit. During the normal operation this bit needs to be set to allow the state machine to start.
TRIZN	Tri-state output enable	Active low to enable tri-state input during the forward(PC to printer) transfers.

Parallel Port Interface Register

The Parallel Port Interface Register is a read/write register that contains eleven bits to control the parallel port interface signals. Four read-only bits are used to read the logic level of host input pins, two read-only bits are used to read the logic level on the BUSY and ACKn printer output pins, and five read/write bits control the logic levels on the printer output pins. When hardware handshaking is enabled(mode 1,2,3) BUSY and ACKn are controlled by the PPI state machine. When hardware handshaking is disabled (mode 0) and the PPI state machine is idle BUSY and ACKn may be controlled by the software using the FBSY and FACK bits.

> *Note: NFLT,SEL,PERR and FBSY are arranged as register bits 0, 1, 2, and 3, respectively, to correspond to their use as parallel data lines during nibble mode reverse data transfers.*

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Interface Reg. *(PIOIF)* **$01FF8203**	(Not Used)	FRDA	SPER (R)	SSEL (R)	SNFL (R)	NINI (R)	NAFD (R)	NSTR (R)	Rst. Value x0000000b Read Value 07h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Parallel Port Interface Reg. *(PIOIF)* **$01FF8202**	NSIN (R)	NACK (R)	BUSY (R)	FACK	FBSY	PERR	SEL	NFLT	Rst. Value 08h Read Value 08h

NFLT	FAULTn	This bit controls the logic level driven on FAULTn output pin. Setting NFLT drives a high level and clearing SEL drives a low level.
SEL	SLCTOUT	This controls the logic level driven on the SLCTOUT output pin. Setting SEL drives a high level and clearing SEL drives a low level.
PERR	Paper Error	This bit controls the logic level driven on the PE output pin. Setting PERR drives a high level and clearing PERR drives a low level.
FBSY	Force Busy	This bit force a high level to be driven on the BUSY output pin. Normally this is done when hardware handshaking is disabled(mode 0), and the PPI state machine is idle. The FBSY bit is OR'ed with the BUSY output from the PPI state machine before driving the BUSY output pin. IF FBSY is set, then the BUSY output pin is forced high, and the BUSY bit is read high. This bit is set on reset.
FACK	Force Ack	This bit forces a low level to be driven on the ACKn output pin. Normally this is done when hardware handshaking is disabled (mode 0), and the PPI state machine is idle. The FACK bit is NOR'ed with the ACKn output from the PPI state machine before driving the ACKn output pin. If FACK is set, then the ACKn output pin is forced low, and the NACK bit is to be read low.
BUSY		This status bit indicates the level driven on the BUSY output pin. This bit is the FBSY bit OR'ed with BUSY output from the PPI state machine. Since FBSY is set on reset, this bit will appear set on reset. This is a read-only status bit.

NACK   ACKn   This status bit indicates the level driven on the ACKn output pin. This bit is the FACK bit NOR'ed with the ACKn output from the PPI state machine. Since FACK is cleared on reset, this bit will appear set on reset. This is read-only status bit.

NSIN   SLCTINn   This status bit indicates the level read on the SLCTINn* input pin after synchronization and optional digital filtering. This is a read-only status bit.

NSTR   Strobe*   This status bit indicates the level read on the STROBE* input pin after synchronization and optional digital filtering. This is a read-only status bit.

NAFD   AUTOFDn   This status bit indicates the level read on the AUTOFDn input pin after synchronization and optional digital filtering. This is a read-only status bit.

NINI   INITn   This status bit indicates the level read on the INITn input after synchronization and optional digital filtering. This is a read-only status bit.

SNFL   FAULTn Status   This status bit indicates the level read on the FAULTn output pin. This is a read-only status bit.

SSEL   SLCTOUT Status   This status bit indicated the level read on the SLCTOUT output pin. This is a read-only status bit.

SPER   Paper Error Status   This status bit indicated the level read on the PE output pin. This is a read-only status bit.

FRDA   Force Reverse Data Available   This bit controls the RDA status during Nibble Mode and Byte Mode when DMA is not used.

Parallel Port Data Register

The parallel Port Data Register is a 9-bit read/write register that is used to control the parallel port data bus. Reading this register provides the latched logic levels at PD7-0 and AUTOFDn pin. Data is latched on every high to low transition of STROBEn (HostClk), when PDOE is clear in the Parallel Port Control Register. If PDOE is set, the Parallel Port Data Register is frozen, and unaffected by the transitions on STROBEn. Reading this register clears the FULL bit, and clears the PDMA request. This register should not be read while the PDMA channel is enabled.

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Data Reg. *(PIOData)* **$01FF8205**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	NCMD	Rst. Value xxxxxxx0b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Parallel Port Data Reg. *(PIOData)* **$01FF8204**	DATA(7)	DATA(6)	DATA(5)	DATA(4)	DATA(3)	DATA(2)	DATA(1)	DATA(0)	Rst. Value 00h Read Value 00h

DATA   Data   This field is an 8-bit read/write field. When used, DATA provides the latched logic levels on PD7-0 when STROBEn (HostCLk) last transmitted from high to low with PDOE clear. When written, the DATA value defines the logic levels to be driven by the QP1700 when PD7-0 is enabled by the PDOE bit. The most significant bit of the DATA field corresponds to PD7 and the least significant bit to PD0.

NCMD   Command   When read, this bit provides the logic level of the AUTOFDn pin when STROBEn (hostClk) transitioned from high to low with PDOE clear. If set, AUTOFDn was latched high. If low, AUTOFDn was latched low. This is a read-only data bit. Writing NCMD has no effect.

Parallel Port ACK Pulse Width Register

The Parallel Port ACK Pulse Width Register is an 8-bit read/write register that defines the pulse width of ACKn during compatibility mode (MODE = 1), and setup timing relative to ACKn during reverse transfer modes (MODE=4,4,6). The Parallel Port ACKn Pulse Width Register can be written and rewritten to program different timing values as transitions are made to new transfer modes.

An ACKn pulse width can be programmed to be 0 to 255 IHSCLK periods wide. At 20MHZ, this allows software to set pulse widths anywhere in the range of 0 to 12.75us. If ACKW is set to zero, no ACKn pulse is generated at the end of compatibility mode cycles.

Setup timing relative to ACKn during reverse transfer modes can be programmed to be 1 to 256 IHSCLK periods. During nibble or byte modes(MODE=4,5), ACKW defines the setup timing from nibble or byte data (event 8 or 15) and the falling edge of ACKn (event 9), or from peripheral status (event 13) to the rising edge of ACKn (event 11). During ECP reverse transfers (MODE = 6), ACKW defines the setup timing from reverse data (event 42) to the falling edge of ACKn (event 43).

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port ACK Pulse Width Reg. *(PIOAckPW)* **$01FF8207**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Parallel Port ACK Pulse Width Reg. *(PIOAckPW)* **$01FF8206**	ACKW(7)	ACKW(6)	ACKW(5)	ACKW(4)	ACKW(3)	ACKW(2)	ACKW(1)	ACKW(0)	Rst. Value 00h Read Value 00h

ACKW   ACKn Pulse Width   This 8 bit field defines the pulse width of ACKn during compatibility mode transfers (MODE = 1), and the setup relationship relative to ACKn during reverse transfer modes (MODE=4,5,6). A pulse width from 0 to 255 IHSCLK periods can be programmed.

Parallel Port Reverse Data Status Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Reverse Data Status Reg. *(PIORevDataSTS)* **$01FF8209 (R)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Parallel Port Reverse Data Status Reg. *(PIORevDataSTS)* **$01FF8208 (R)**	RVDAT(7)	RVDAT(6)	RVDAT(5)	RVDAT(4)	RVDAT(3)	RVDAT(2)	RVDAT(1)	RVDAT(0)	Rst. Value 00h Read Value 00h

REVDATA   Reverse Data   This 8-bit field is the status of the reverse data that was written into the Parallel Port Data Register oppid[7:0]. This normally is needed only for diagnostic purposes. This is a read-only status register.

Parallel Port Data Bus Status Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Data Bus Status Reg. *(PIODataBusSTS)* **$01FF820B (R)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Parallel Port Data Bus Status Reg. *(PIODataBusSTS)* **$01FF820A (R)**	PDBUS(7)	PDBUS(6)	PDBUS(5)	PDBUS(4)	PDBUS(3)	PDBUS(2)	PDBUS(1)	PDBUS(0)	Rst. Value 00h Read Value 00h

PPDBUS Data Bus Status   This 8 bit field is the status of the parallel port data bus ippid[7:0]. This normally is needed only for diagnostic purposes. This is a read-only status register.

Parallel Port Host Timeout Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Host Timeout Reg. *(PIOHostTimeOut)* **$01FF820D**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Parallel Port Host Timeout Reg. *(PIOHostTimeOut)* **$01FF820C**	HTIM(7)	HTIM(6)	HTIM(5)	HTIM(4)	HTIM(3)	HTIM(2)	HTIM(1)	HTIM(0)	Rst. Value 00h Read Value 00h

HTIM Host Timeout   This 8 bit field controls the timer for the host timeout error interrupt HTE. The timer counts from the system timer which is usually programmed to count every 8 msec. HTTM is usually programmed with a value of 0x7d (125) so that a host timeout interrupt occurs after 1 sec. If the host timeout error interrupt HTE is not desired, then interrupt enable should be false.

Interrupt Status Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Interrupt Status *(PIOIRQSTS)* **$01FF820F**	(Not Used)	DMARIRQ	DMAFIRQ	HTE	RDX	IVT	CRX	DRX	Rst. Value x0000000b Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Parallel Port Interrupt Status *(PIOIRQSTS)* **$01FF820E**	INIL	INIH	AFDL	AFDH	STRL	STRH	SINL	SINH	Rst. Value 00h Read Value 00h

SINH	SLCTINn high edge interrupt.
SINL	SLCTINn low edge interrupt.
STRH	STROBEn high edge interrupt.
STRL	STROBEn low edge interrupt.
AFDH	AUTOFDn high edge interrupt.
AFDL	AUTOFDn low edge interrupt.
INIH	INITn high edge interrupt.
INIL	INITn low edge interrupt.
DRX	data received interrupt.
CRX	command received interrupt.
IVT	invalid transition interrupt.
RDX	rev data sent interrupt.
THE	host timeout interrupt.
FDMAIRQ	Forward transfer DMA end of block interrupt
RDMAIRQ	Reverse transfer DMA end of block interrupt

Clearing any interrupt requires writing a 1 to the corresponding bit in the interrupt status register.

Interrupt Mask Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port Interrupt Mask. *(PIOIRQMask)* **$01FF8211**	(Not Used)	DMARIRQ MASK	DMAFIRQ MASK	HTE MASK	RDX MASK	IVT MASK	CRX MASK	DRX MASK	Rst. Value x0000000b Read Value 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Parallel Port Interrupt Mask *(PIOIRQMask)* **$01FF8210**	INIL MASK	INIH MASK	AFDL MASK	AFDH MASK	STRL MASK	STRH MASK	SINL MASK	SINH MASK	Rst. Value 00h Read Value 00h

The mask register will be used to enable the interrupts when writing one to each corresponding bit and disable the interrupts when writing zero to the corresponding bit.

FIFO Interface Register

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Parallel Port FIFO Interface *PIOFIFOIF* **01FF8213**	(Not Used)	(Not Used)	REVFIFO_LCLCLEARING	REVFIFO_LCLFLUSHING	REVFIFO_LCLREQ (RD)	FWDFIFO_LCLCLEARING (RD)	FWDFIFO_LCLFLUSHING (RD)	FWDFIFO_LCLREQ (RD)	Rst. Value xx000000b Read Value 00h
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
Parallel Port FIFO Interface *PIOFIFOIF* **01FF8212**	REVFIFO_LCLCLR	FWDFIFO_LCLCLR	FWDFIFO_TC0EN	FWDFIFO_ENABLE	FWDFIFO_FLUSH	REVFIFO_TC0EN	REVFIFO_ENABLE	REVFIFO_FLUSH	Rst. Value 00h Read Value 00h

REVFIFO_FLUSH — causes the reverse FIFO to execute a flush of its contents.

REVFIFO_ENABLE — enables the reverse FIFO operation. When false, any pending DMA requests are first completed before the reverse FIFO becomes disabled.

REVFIFO_TC0EN — When low it enables the reverse FIFO DMA operation.

FWDFIFO_FLUSH — causes the forward FIFO to execute a flush of its contents.

FWDFIFO_ENABLE — enables the forward FIFO operation. When false, any pending DMA requests are first completed before the forward FIFO becomes disabled.

FWDFIFO_TC0ENA — When low it enables the forward FIFO DMA operation.

FWDFIFO_LCLCLR — FWD FIFO local asynchronous clear except waits for any pending DMA requests to finish before reset.

REVFIFO_LCLCLR — REV FIFO local asynchronous clear except waits for any pending DMA requests to finish before reset.

FWDFIFO_LCLREQ — FWD FIFO local request is similar to dmareq but ignores dmaEq0 input. Used for firmware DMA as a status signal.

FWDFIFO_LCLFLUSHING — high when lclFlush is pulsed high and stays high until the FIFO is flushed.

FWDFIFO_LCLCLEARING — high when lclClr is pulsed and stays high until the clear can be done(no dmaReq).

REVFIFO_LCLREQ — REV FIFO local request is similar to dmareq but ignores dmaEq0 input. Used for firmware DMA as a status signal.

REVFIFO_LCLFLUSHING — high when lclFlush is pulsed high and stays high until the FIFO is flushed.

REVFIFO_LCLCLEARING — high when lclClr is pulsed and stays high until the clear can be done (no dmaReq).

DMA FIFO Registers

There will be two 4 half word FIFOs used for forward and reverse DMA transfers. The FIFO locations will be accessed from CPU as well as DMA. The address locations for the forward DMA FIFO will be 01FF8220 to 01FF822F. The address locations for the reverse FIFO will be 01FF8230 to 01FF823F.

Output data FIFO halfword[3:0] for the data to the external memory (the forward direction) (for DMA channel 11):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Parallel Port Output Data FIFO *(PIOOutFIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Parallel Port Output Data FIFO *(PIOOutFIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

- Address assignment for Output FIFO halfword[3:0]

The Input FIFO halfword	The register name	Address
Output FIFO halfword 3	*PIOOutFIFO3*	01FF8227-26
Output FIFO halfword 2	*PIOOutFIFO2*	01FF8225-24
Output FIFO halfword 1	*PIOOutFIFO1*	01FF8223-22
Output FIFO halfword 0	*PIOOutFIFO0*	01FF8221-20

Output data FIFO Holding Register for the data to the external memory (the forward direction) (for DMA channel 11):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Parallel Port Output Data FIFO Holding Register *(PIOOutHold)* **$01FF8229**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value xxh Read Value xxh
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Parallel Port Output Data FIFO Holding Register *(PIOOutHold)* **$01FF8228**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

Output data FIFO Control Register for the data to the external memory (the forward direction) (for DMA channel 11):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Parallel Port Output Data FIFO Control Register *(PIOOutFIFOCtrl)* **$01FF822B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Parallel Port Output Data FIFO Control Register *(PIOOutFIFOCtrl)* **$01FF822A**	(Not Used)	Holding Register Full	Byte Present	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0000000b Read Value 00h

Bit 15:	This bit indicates that the FIFO is active and capable of generating DMA requests. This bit generally follows the setting of the LCL_ENB input signal except the FIFO Enabled register bit will remain asserted if the LCL_ENB input signal is set to false while the FIFO is executing a DMA transfer. In this case, the register bit will remain set for the duration of the DMA transfer and then become cleared.
Bit 14:	This bit is similar to the DMA_REQ signal except that it is not blocked when DMA_TC0 is set. It indicates that the programmed threshold has been met or exceeded and that the FIFO requires data transfers either by enabling hardware DMA through DMA_TC0 or by executing software DMA cycles.
Bit 13:	This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.
Bit 12-10:	This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 1 to 4. A value of zero causes no data transfers to occur. Values greater than 4 are treated as 4.
Bit 9-8:	This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.
Bit 7:	Not used
Bit 6:	This bit indicates that there is a full word in the holding register.
Bit 5:	This bit indicates that there is a single byte in the holding register.
Bit 4-2:	This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.
Bit 1-0:	This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

> **Note:** *It is strongly recommended that the FIFO be disabled before firmware writes to the FIFO Control register. The FIFO Control register contains data that is essential to the FIFO's operation. If this data is changed while the FIFO is operating, unpredictable operation will result.*

Input data FIFO halfword[3:0] for the data from the external memory (the reverse direction) (for DMA channel 12):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Parallel Port Input Data FIFO *(PIOInFIFOx)*	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Parallel Port Input Data FIFO *(PIOInFIFOx)*	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

- Address assignment for Input FIFO halfword[3:0]

The Input FIFO halfword	The register name	Address
Input FIFO halfword 3	*PIOInFIFO3*	01FF8237-36
Input FIFO halfword 2	*PIOInFIFO2*	01FF8235-34
Input FIFO halfword 1	*PIOInFIFO1*	01FF8233-32
Input FIFO halfword 0	*PIOInFIFO0*	01FF8231-30

Input data FIFO Holding Register for the data from the external memory (the reverse direction) (for DMA channel 12):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Parallel Port Input Data FIFO Holding Register *(PIOInHold)* **$01FF8239**	data bit 15	data bit 14	data bit 13	data bit 12	data bit 11	data bit 10	data bit 9	data bit 8	Rst Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Parallel Port Input Data FIFO Holding Register *(PIOInHold)* **$01FF8238**	data bit 7	data bit 6	data bit 5	data bit 4	data bit 3	data bit 2	data bit 1	data bit 0	Rst Value 00h Read Value 00h

Input data FIFO Control Register for the data from the external memory (the Reverse direction) (for DMA channel 12):

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Parallel Port Input Data FIFO Control Register *(PIOInFIFOCtrl)* **$01FF823B**	FIFO Enabled (R)	Data Request (R)	FIFO Ready (R)	FIFO DMA Threshold			FIFO Output Pointer		Rst Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Parallel Port Input Data FIFO Control Register *(PIOInFIFOCtrl)* **$01FF823A**	(Not Used)	Holding Register Full	Byte Present	FIFO Data Quantity			FIFO Input Pointer		Rst Value x0000000b Read Value 00h

Bit 15: This bit indicates that the FIFO is active and capable of generating DMA requests. This bit generally follows the setting of the LCL_ENB input signal except the FIFO Enabled register bit will remain asserted if the LCL_ENB input signal is set to false while the FIFO is executing a DMA transfer. In this case, the register bit will remain set for the duration of the DMA transfer and then become cleared.

Bit 14: This bit is similar to the DMA_REQ signal except that it is not blocked when DMA_TC0 is set. It indicates that the programmed threshold has been met or exceeded and that the FIFO requires data transfers either by enabling hardware DMA through DMA_TC0 or by executing software DMA cycles.

Bit 13: This bit indicates that the FIFO can accept data transfers to/from the local logic. This bit is low when the data direction is into the local logic and the FIFO is empty or when the data direction is out of the local logic and the FIFO is full.

Bit 12-10: This bit field indicates at which point the FIFO should issue a DMA request. It is compared with the FIFO Data Quantity bit field to determine when this should occur. Legal values are from 1 to 4. A value of zero causes no data transfers to occur. Values greater than 4 are treated as 4.

Bit 9-8: This bit field indicates which byte of the FIFO quad halfword structure is the next to be used as output data.

Bit 7: Not used

Bit 6: This bit indicates that there is a full word in the holding register.

Bit 5: This bit indicates that there is a single byte in the holding register.

Bit 4-2: This bit field indicates the number of quad halfwords that contain data. Valid values are from 0 to 4.

Bit 1-0: This bit field indicates which byte of the FIFO quad halfword structure is the next to receive input data.

Note: It is strongly recommended that the FIFO be disabled before firmware writes to the FIFO Control register. The FIFO Control register contains data that is essential to the FIFO's operation. If this data is changed while the FIFO is operating, unpredictable operation will result.

17.5 Timing

17.5.1 Compatibility Timing

When this mode of handshaking is enabled, the PPI automatically generates BUSY upon reception of the leading edge of STROBEn from the host, and latches the logic levels to the ippid bus and AUTOFDn in the Parallel Port Data Register. The PPI then waits for the STROBEn to de-assert and the Parallel Port Data Register to be read. After the later of these events occurs, the PPI asserts ACKn for the duration specified in the Parallel Port ACK Pulse Width Register and then de-asserts ACKn and BUSY to conclude the data transfer.

When data is latched into the Parallel Port Data Register, the PPI controller generates two events: an internal parallel port DMA request, and a processor interrupt request. Software may alternate between DMA and interrupt based data transfers. The DMA request remain active until the data is read by either the DMA channel, or the processor in response to the interrupt request.

If the Forward DMA FIFO has been enabled, the Forward DMA FIFO generates fdmareq after reading the Parallel Port Data Register and writing the data to its internal half word quad registers. When the fdmack is generated the DMA channel reads the Forward FIFO internal Registers, ACKn is pulsed, BUSY is deserted, and the fdmack is cleared.

17.5.1.1 Compatibility Mode Timing Diagram

1. Host writes data to the data bus
2. If obusy is not active, the host will assert the istroben low.
3. Peripheral will assert obusy high to indicate data transfer cycle.
4. Host will set the istroben and at the rising edge, the peripheral will latch the data .
5. Peripheral asserts oackn low to acknowledge the transfer and drops the obusy.
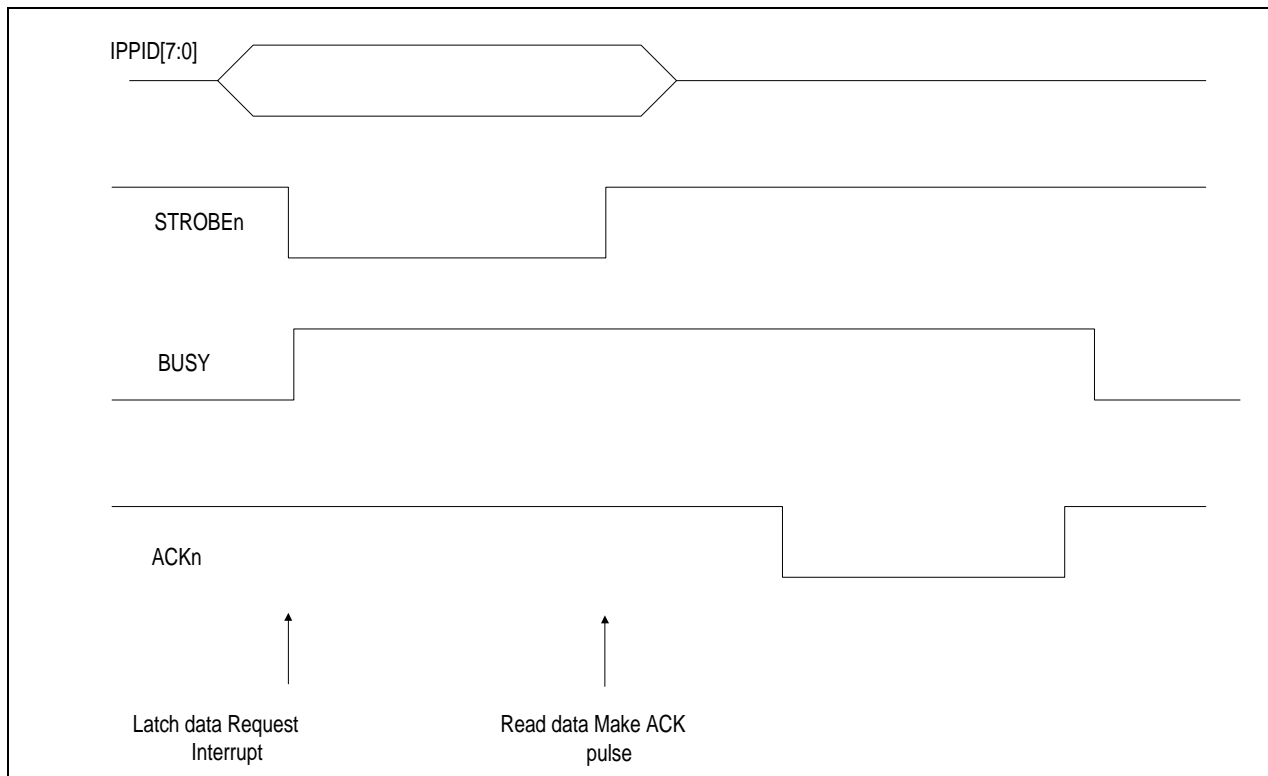
Figure 17-2. Compatibility Mode Timing Diagram

17.5.2 Nibble Mode Timing

The Nibble mode is the most common way to get reverse channel from printer or peripheral. This mode is usually combined with the compatibility mode or a proprietary forward channel mode to create a complete bi-direction channel.

The Nibble Mode Phase transitions:

1. Host signal ability to take data by asserting AUTOFDn low
2. Peripheral responses by placing first nibble on status lines(FAULTn,SLCTOUT,PE and BUSY)
3. Peripheral signals valid nibble by asserting ACKn low
4. Host sets AUTOFDn high to indicate that it has received the nibble and is not ready for another nibble.
5. Peripheral set ACKn high for the second nibble.
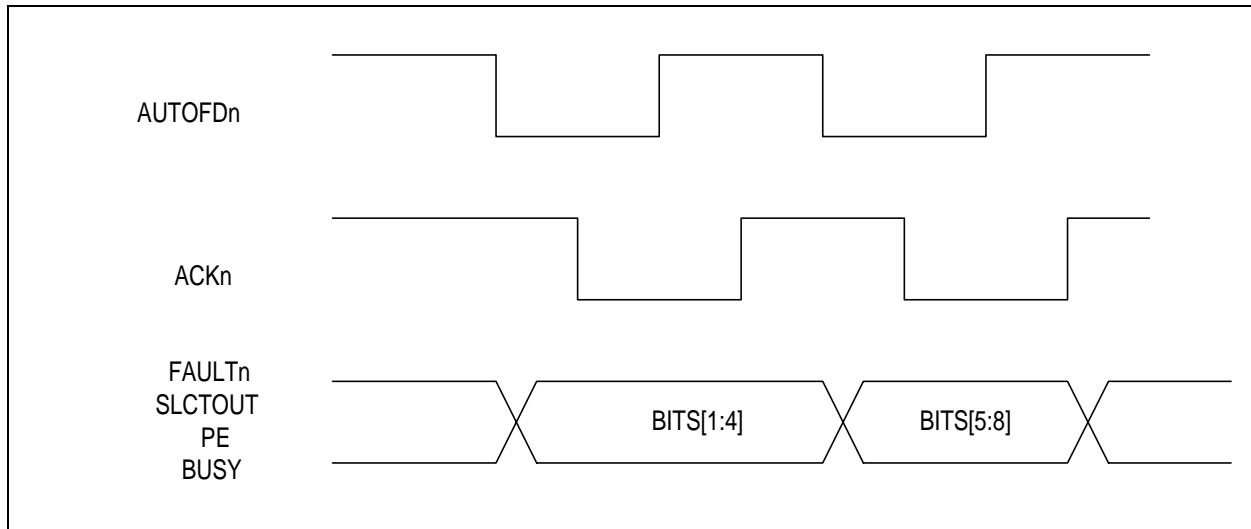
Figure 17-3. Nibble Mode Data Transfer Cycle

17.5.3 Byte Mode Timing

Byte mode is a reverse channel data transfer used to provide data rates into the PC approaching that of compatibility mode.

The byte mode signal handshaking is as follows:

1. Host signals ability to take data by asserting AUTOFDn low
2. Peripheral responds by placing first byte on data lines
3. Peripheral signals valid byte by asserting ACKn low
4. Host sets AUTOFDn high to indicate it has received data
5. Peripheral sets ACKn high to acknowledge host
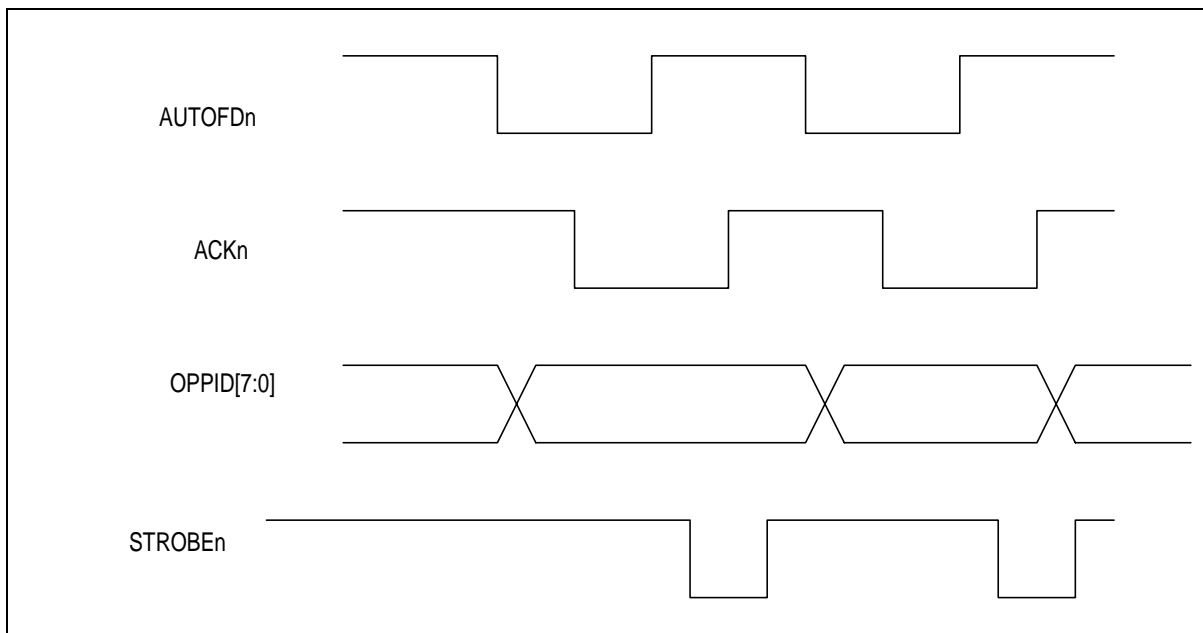6. Host pulses STROBEn as an acknowledge to the peripheral.

Figure 17-4. BYTE Mode Data Transfer Cycle

17.5.4 ECP Mode Timing

Two modes of hardware handshaking are offered for ECP forward transfers. ECP mode hardware handshaking is enabled by setting the MODE field in the Parallel Port Control Register to 2. The ECP mode hardware handshaking with RLE is enabled by setting MODE to 3. When either mode is enabled, the PPI automatically responds to STROBEn (HostClk) by latching the logic levels on the PD bus and AUTOFDn in the Parallel Data Register. When the Parallel Port Data Register is read, the PPI drives BUSY high,waits for STROBEn to high, and then drives BUSY (periphAck) low to conclude the cycle. The BUSY signal is controlled by the PPI automatically if the data is received using DMA mode. Note that since no ACKn pulse is generated, the pulse width duration specified in the Parallel Port ACK Pulse Width Register is not used in any manner. As in the compatibility handshake mode, data is latched at the leading edge of STROBEn. This then causes a DMA request and posting of DRX (data received) interrupt event. DMA and interrupt operation is the same as explained in the compatibility mode. In mode 2, reception of both run-length counts and channel addresses causes a CRX interrupt event to be posted. Software is responsible for responding to channel addresses and performing data decompression. When MODE is set to 3, ECP mode hardware handshaking is enabled during forward data transfers, with Run Length Encoding(RLE) data compression support. If MODE is reprogrammed when decompression is taking place, that is, when RLD is set, the decompression continues unhindered to completion.

17.5.4.1 Forward ECP Transfer Handshake:

The host places data on the data lines and indicates a data cycle by setting AUTOFDn high.

1. Host asserts STROBEn low to indicate valid data.
2. Peripheral acknowledges host by setting BUSY high.
3. Host sets STROBEn high and the data is clocked in to the peripheral.
4. Peripheral sets BUSY low to indicate that it is ready for the next byte.
5. The cycle repeats, but this time it is a command cycle because AUTOFDn is low.
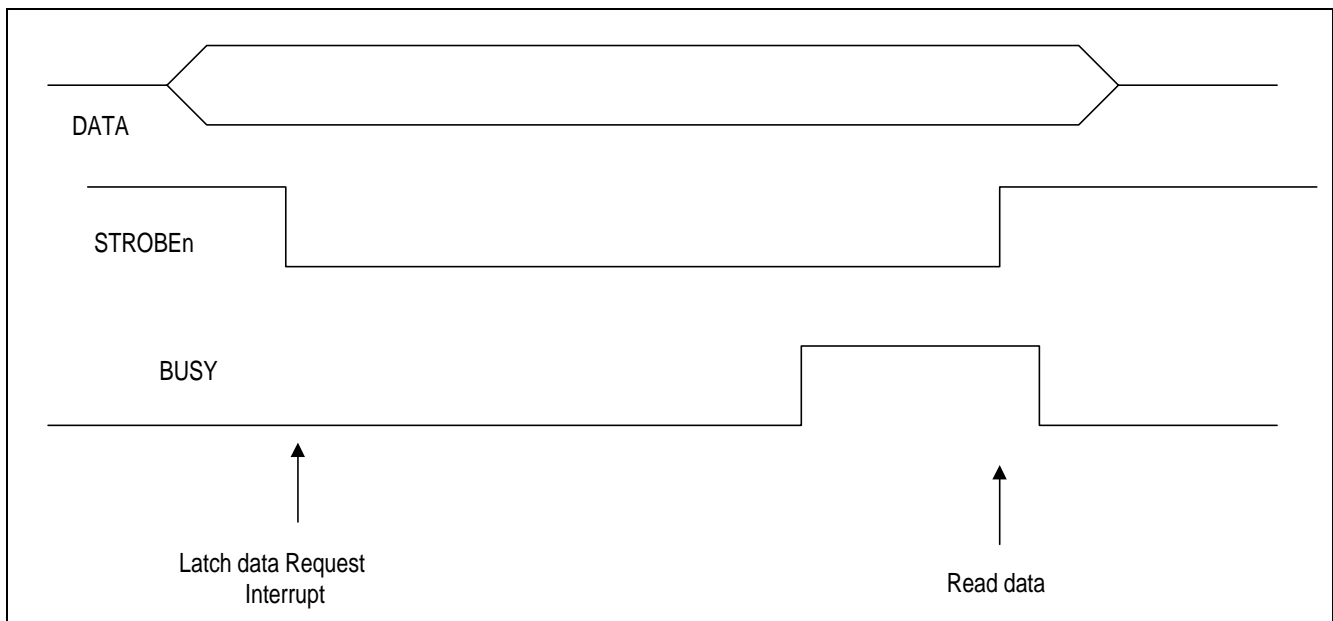
Figure 17-5. ECP Mode Timing Diagram

17.5.4.2   Reverse ECP Transfer Handshake

1.   The host requests a reverse channel transfer by setting INITn low.
2.   The peripheral signals that it is OK to proceed by setting PE low.
3.   The peripheral places data on the data lines and indicates a data cycle by setting BUSY high.
4.   The peripheral asserts ACKn low to indicate valid data.
5.   The host acknowledges by setting AUTOFDn high.
6.   The peripheral sets ACKn high to clock the data in to the host.
7.   The host sets AUTOFDn low to indicate that it is ready for the next byte.
8.   The cycle repeats, but this time it is a command cycle because ACKn is low.
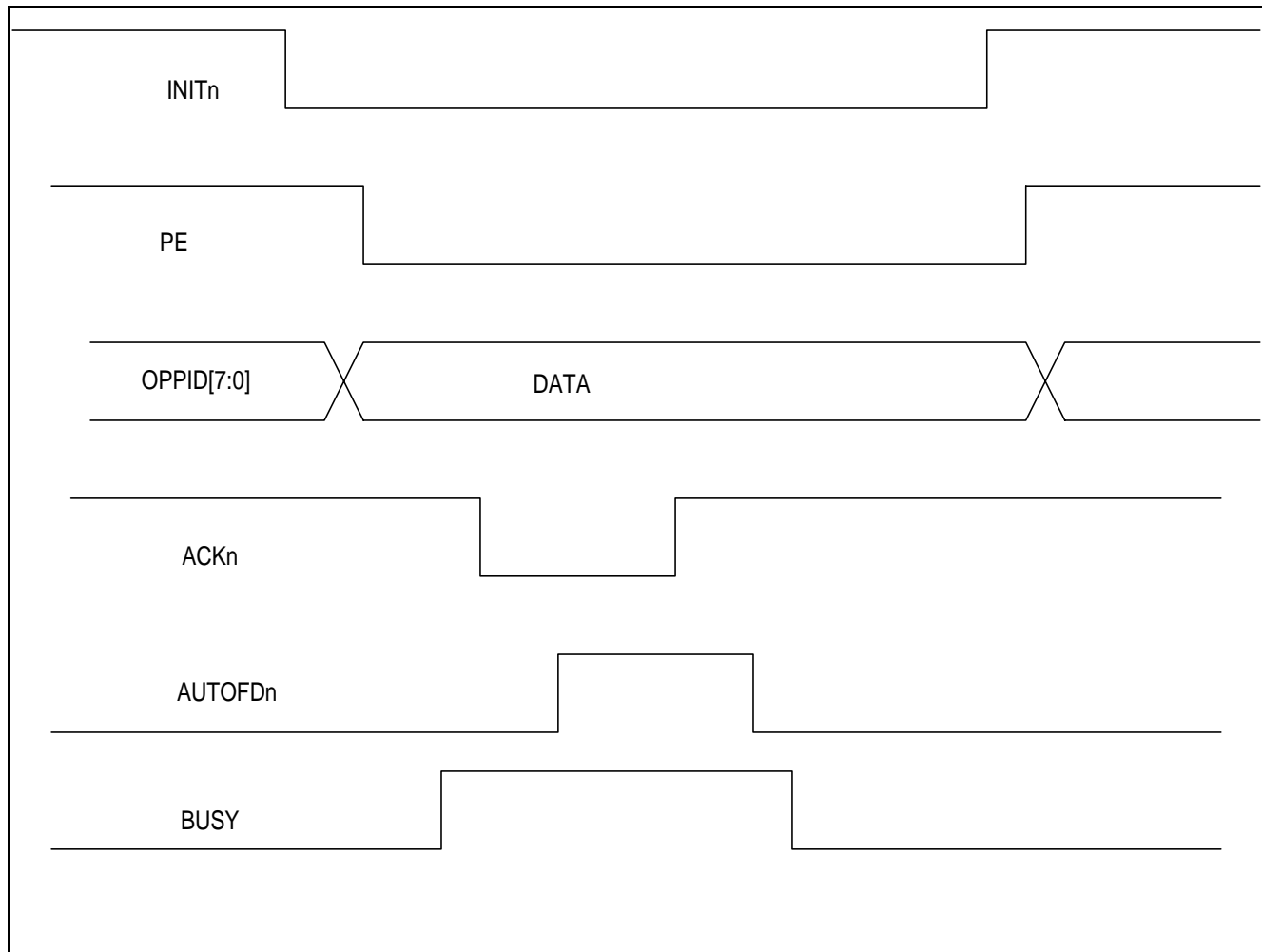
Figure 17-6. Reverse ECP Transfer Timing

17.5.5 Error Cycle Timing

Error cycles are performed by a printer to alert the host of a change in the operational status of the printer. Error cycles include such events as the user taking the printer off-line, the occurrence of a paper jam, or the printer running out of paper.

An error cycle consists of raising BUSY, and then changing the state of any one or more of the three status lines, SLCTOUT, PE and FAULTn to reflect the error condition. The ERRC bit in the control register prevents handshake logic from generating ACKn and BUSY in the compatibility mode.
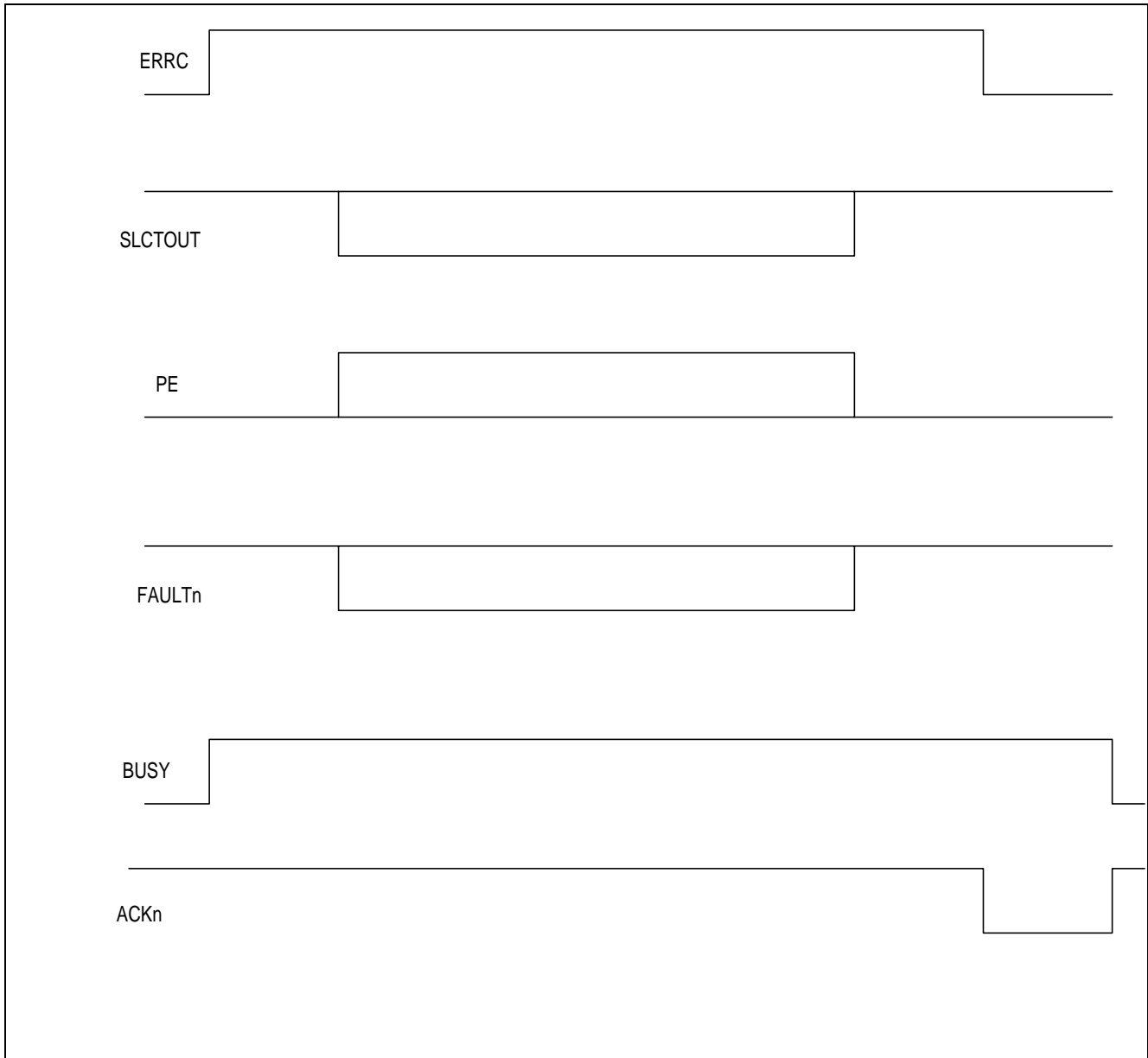
Figure 17-7. Error Cycle Timing Diagram

17.6 Firmware Operation

All the Parallel Port negotiations will be done in firmware.

17.6.1 Disabling All Hardware Handshaking

When hardware handshaking is disabled(MODE=0), software is then responsible for controlling the parallel port interface. Software can control all parallel port operations including negotiation and termination phases, as well as reverse channel transfers. This provides flexibility to adapt to existing protocols as they are revised and to new protocols as they are created.

Software can control the BUSY and ACKn interface pins with the FBSY (force BUSY) and FACK (force ACKn) bits in the Parallel Port Interface Register. The FBSY bit is OR'ed with the BUSY output from the PPI state machine before driving the BUSY interface pin, and the FACK bit is NOR'ed with the ACKn output from the PPI state machine before driving the ACKn interface pin. Normally the PPI state machine should be idle when using FBSY and FACK. Even with all hardware handshaking disabled, the Parallel port register continues to latch parallel port data on the leading edge of the STROBEn. Software can issue a RST in the Parallel Port Control Register to immediately force the PPI state machine to idle.

17.6.2 Software Interrupts

The following is a list of P1284 signal transition interrupts provided for ease of use:

SLCTINn	Rising and Falling
STROBEn	Rising and Falling (remains high until data is fetched from PPI)
INITn	Rising and Falling
AUTOFDn	Rising and Falling
DRX	Data Received
CRX	Command Received
IVT	Invalid Transition
RDX	Reverse Data Transmitted
HTE	Host Time-out Error

17.6.3 Interrupt Operation

There will be a bit corresponding to each interrupt in the interrupt enable register that enables the interrupt. When the event corresponding to an interrupt occurs, this bit needs to be set at the following rising edge of the SIU clock. The PPI Interface possesses 15 sources of interrupts. Each source can post an event in the interrupt status register and each event can be individually enabled or disabled in the interrupt mask register. The fifteen interrupts have been in Section 17. The first eight interrupt events signal level changes that occur at the host control signal input pins. Note that these events are detected after the host inputs are synchronized, digitally filtered, and recorded in the parallel Port Interface Register.

18. Real-Time Clock

18.1 Description

The Real-Time Clock (RTC) circuit is required to maintain current time information under MFC2000 primary power and battery backup conditions.

The RTC Circuit consists of six binary counters that track seconds, minutes, hours, days, months, and years elapsed since the reference year, 1992. Each counter can be read from the appropriate register described. The tracking range of the RTC is up to 32 years, and automatic compensation is made for leap years.

A 32768 Hz signal is required to operate RTC counters and logic. A 32768 Hz (watch) crystal should be used. The RTC block diagram is illustrated in Figure 18-1.
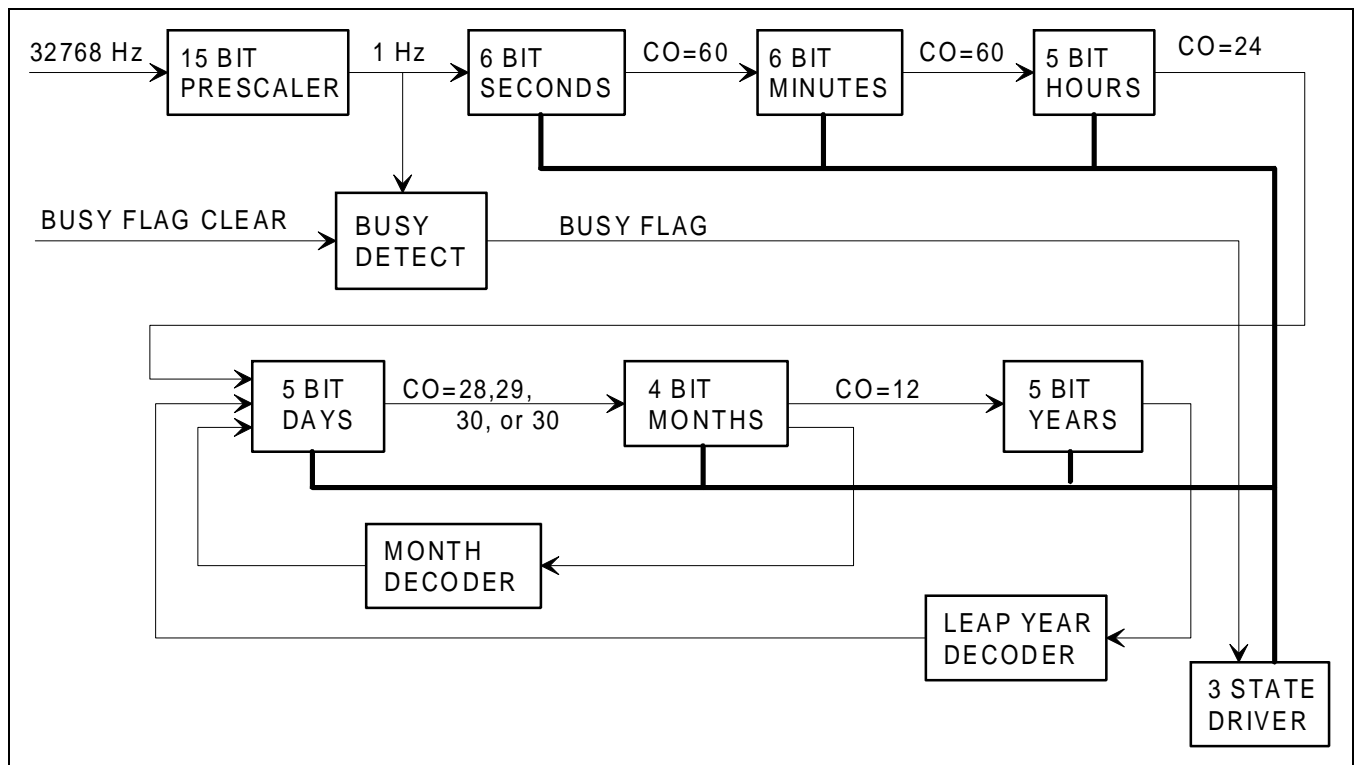

Figure 18-1. RTC Block Diagram

Conexant

18.2 Real-Time Clock (RTC) Registers

> **Note:** All RTC registers are cleared only by battery power-on reset or a write to the RTC control register.

As you can see, second and minute are in the same 16-bit register. Hour_Day, and Month_Year registers have the same arrangement. If CPU does a 16-bit write to the Hour_Day register, both hour and day will increment. If CPU only writes to the lower 8-bit or the upper 8-bit, only hour or only day will increment. The Month_Year register operates in the same manner. The Sec_Min register operates differently. If CPU does a 16-bit write or a lower 8-bit write to the Sec_Min register, year, month, day, hour, and minute will increment once. If CPU only writes to the upper 8-bit, only minute will increment.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Second and Minute *(Sec_Min)* **01FF8091**	Busy Flag (R)	(Not Used)	Read: Data Range 000000 - 111011 Write: increment minute, (data is a don't care)						Rst. Value 0x000000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Second and Minute *(Sec_Min)* **01FF8090**	Busy Flag (R)	(Not Used)	Read: Data Range 000000 - 111011 Write: increment all - year,month,day,hour and minute registers, (data is a don't care)						Rst. Value 0x000000b Read Value 00h

Increment all will increment the year, month, day, hour, and minute registers by one.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Hour and day *(Hour_Day)* **01FF8093**	Busy Flag (R)	(Not Used)	(Not Used)	Read: Data Range 00000 - 11110 Write: increment day (data is a don't care)					Rst. Value 0xx00000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Hour and Day *(Hour_Day)* **01FF8092**	Busy Flag (R)	(Not Used)	(Not Used)	Read: Data Range 00000 - 10111 Write: increment hour (data is a don't care)					Rst. Value 0xx00000b Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Month and Year *(Month_Year)* **01FF8095**	Busy Flag (R)	(Not Used)	(Not Used)	Read: Data Range 00000 - 11111 Write: increment year (data is a don't care)					Rst. Value 0xx00000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Month and Year *(Month_Year)* **01FF8094**	Busy Flag (R)	(Not Used)	(Not Used)	(Not Used)	Read: Data Range 0000 - 1011 Write: increment month, (data is a don't care)				Rst.Value 0xxx0000b Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Real Time Clock Ctrl *(RTC)* **01FF8097**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Real Time Clock Ctrl *(RTC)* **01FF8096**	Read: Clears the busy flag (data is undefined) Write: Resets the Real Time Clock (RTC), (data is a don't care)								Rst. Value xxh Read Value 00h

Read: clear BUSY flag, data is undefined. Write: reset the RTC, data is 'don't care'.

18.3 RTC Operations

18.3.1 Power-Up/Power-Down Operation

When prime power is applied to the MFPC and battery reset (BATRSTn) is generated, all counters in the RTC are reset to 0 and the RTC becomes accessible after a delay of from 125 to 250 ms, provided that the crystal clock is running. (Note: The RTC is also software resetable.)

When prime power drops below a level where the power-down input (PWRDWNn) becomes active, an internal lockout signal prevents read and write access of the RTC, and RTC operation switches to battery power. Inactivation of the PWRDWNn pin (when prime power is restored) removes the lockout condition, and the RTC is again accessible. (Note: activation of external reset, RESETn, does not affect RTC operation.)

18.3.2 Setting Time

The first step in setting the time is to reset the time counters by writing to the RTC Control register.

The user can then set the current year (number of years elapsed since the reference year, 1992) into the upper 8-bit of the month_year register. Each write to the year register increments the year counter by one.

The month is then set by writing to the lower 8-bit of the month_year register, and each write increments the month counter by one.

The same procedure should then be used to set the correct day, hour, and minute by writing to the corresponding registers. Writing to month_year, hour_day, or sec_min register resets the seconds counter to all zeros.

> **Note** that the RTC updates the second counter once per second independent of read or write operations to RTC registers.

18.3.3 Reading Time

Each 8-bit (upper or lower) of the time interval registers can be independently read. The upper most bit of each 8-bit section is a Busy Flag, which when set, indicates that a one second clock edge has occurred, and that the time is being updated in all registers. If the Busy Flag is set when reading any of the time registers, the user must issue a read to the RTC Control register to clear the Busy Flag, and then re-read all of the time registers until all Busy Flags are 0.

18.3.4 Crystal Oscillator

Specifications for the clock crystal, a crystal oscillator schematic and circuit board layout recommendations follow.

18.3.5 RTC Crystal Specifications

Table 18-1. RTC Crystal Specifications for 32.768 kHz

Characteristic	Value for 32.768 kHz
Nominal Frequency @25°C (F)	32.768 kHz ± 20 ppm
Turnover Temperature (To)	25°C ± 5°C
Temperature Characteristic (K)*	-0.038 ppm/°C2 Typ
Temperature Stability	-160 ppm, Typ @ -40°C -137 ppm, Typ @ +85°C
Quality Factor	100,000 Typ
Load Capacitance	12.5 pF ±2%
Series Resistance	30 KΩ Max
Motional Capacitance	3.5 pF Typ
Shunt Capacitance	1.7 pF Typ
Maximum Drive Level	1.0 µW
Aging (Δ F/f)	5.0 ppm, First Year
Operating Temperature	-10°C to +60°C
*(Δ F/f) = K(To-T)2 where T= point of temperature comparison	

Recommended RTC Crystal Interface Circuit

Typical Values

Fc	32.768 kHz
R1	10 MΩ
R2	0 KΩ
C1	22 pf
C2	22 pf
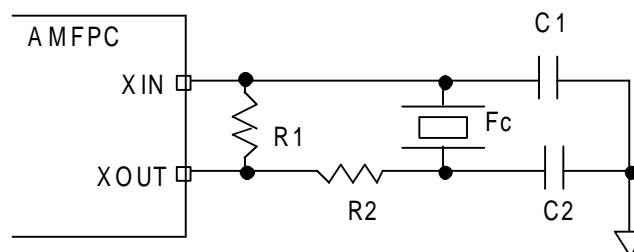
Note: Adjustment to the values of C1 and C2 may be required to obtain the correct operating frequency due to stray capacitance on the Developer's circuit board.

Conexant

Circuit Layout Criteria

For best performance the following design steps should be followed:

1. Keep stray capacitance and resistance to a minimum.
2. Run XIN and XOUT traces as far apart as possible.
3. Clear the ground plane from under the interface circuit.
4. Keep digital signals away from this interface.
5. This is a high impedance circuit; residual contamination (e.g., water soluble flux) of this circuit may cause unreliable operation.

This page is intentionally blank.

19. Synchronous Serial Interface (SsIF)

19.1 Introduction and Features

19.1.1 Introduction

There are two identical SSIFs in the MFC2000. The common signal and block names are used in the functional and timing descriptions without indicating which SSIF. Two SSIFs are distinguished as SSIF1 and SSIF2 only in the register description. The SSIF built into the MFC2000, allows system peripherals to communicate with the MFC2000. The SSIF provides separate signals for Data (SSTXD, SSRXD), Clock (SSCLK), and optional (SSREQ) and Data Acknowledge (SSACK). The SSIF may be configured to operate as either a master or slave interface. After reset, SSTXD is 0 and SSSTAT is 1. The following describes the operation of this synchronous only serial interface.

19.1.2 Features

- Full duplex, three wire system
- Master or slave operation (Bi-directional Clock)
- Programmable bit rates with maximum frequency = $AUXCLK/2$ and minimum = $AUXCLK/32$ with a 50% duty cycle.
- Programmable clock polarity and phase
- Internal interrupt for *Receive Register Full* (mask-able at the Interrupt Controller)
- Optional data request/acknowledge interface during slave mode
- 8 data bit synchronous serial-interface with programmable data shifting order
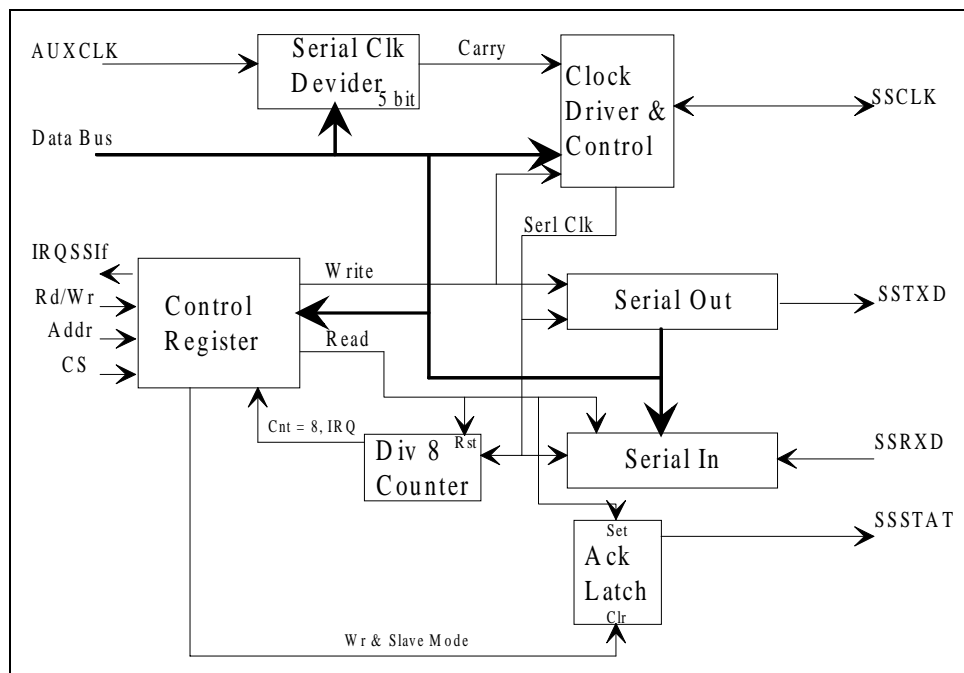
Figure 19-1. SSIF Block Diagram

19.2 Register Description

19.2.1 The 1st SSIF (SSIF1)

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SSData1 **01FF8103**	(Not Used)								Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SSData1 **01FF8102**	SSData1 (7-0)								Rst. Value xxh Read Value xxh

Register Description: SSIF1 Data Register

Note:

The SSData1 Register serves as Transmit Shift Register (TxBuffer1) during CPU write, or as Receive Shift Register (RxBuffer1) during CPU read. Reading the SSData1 Register will clear the SSInFull1 Interrupt bit. Writing to the SSData1 Register will reset the shift bit counter, overwrite the output register and initiate an 8 bit serial shift while in serial mode.

A bit called SSSTATOut1 is added into the SSCmd1 register. The SSSTATOut1 value will be output to the SSSTAT1/GPIO[8] pin as the GPIO[8] data when the SSSTAT1/GPIO[8] pin is programmed as GPIO[8] through the GPIOConfig register.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SSCmd1 **01FF8101**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	SSSTATOut 1	Rst. Value xxxxxxx0b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SSCmd1 **01FF8100**	(Not Used)	(Not Used)	(Not Used)	L2MSB1	SSCLKPol1	SSCLKPhas 1	(Not Used)	Mode11 0 - Master 1 - Slave	Rst. Value xxx000x0b Read Value 00h

Register Description: SSIF1 Command Register

Bit 8:		The SSSTATOut1 bit is used only when the SSSTAT1/GPIO[8] pin is programmed as GPIO[8] through the GPIOConfig register. The data value in the SSSTATOut1 bit is the GPIO[8] data value.
Bit 4:	L2MSB1, Read/Write bit.	This bit controls the shifting order of the transmit and receive registers. When the L2MSB1 bit is cleared (default), the transmit and receive shift register will shift from the MSB to the LSB. When the L2MSB1 bit is set, the transmit and receive register will shift from the LSB to MSB.

Bit 3:   SSCLKPol 1, Read/Write bit.   This bit controls the polarity of the SSCLK1 pin. The default value is zero. When the SSMode1 is configured at the IOMode1, the SSCLKPol1 bit will not affect the SSCLK1 pin value.

Bit 2:   SSCLKPhas1, Read/Write bit.   The default value is zero.

SSCLKPol1	SSCLKPhas1	Description (used for both master and slave modes)
0	0	Enable the external logic to clock the SSTXD1 pin on the rising edge of the SSCLK1 pin. The SSCLK1 pin's first transition will be in the middle of the first bit.
0	1	Enable the external logic to clock the SSTXD1 pin on the falling edge of the SSCLK1 pin. The SSCLK1 pin's first transition will occur at the beginning of the first bit.
1	0	Enable the external logic to clock the SSTXD1 pin on the falling edge of the SSCLK1 pin. The SSCLK1 pin's first transition will be in the middle of the first bit.
1	1	Enable the external logic to clock the SSTXD1 pin on the rising edge of the SSCLK1 pin. The SSCLK1 pin's first transition will occur at the beginning of the first bit.

Figure 19-2. SSCLK1 Diagram

Note: SSCLK1 is used to synchronize the movement of data both in and out of device through SSRXD1 and SSTXD1 pins. The master and slave devices are capable of exchanging a data byte of information during a sequence of eight clock pulses. Both master and slave devices must be operated in the same timing mode as controlled by SSCLKPol1 and SSCLKPhas1 signals.

Bit 0:   Mode11, Read/Write bit.   This bit controls the SSIF1 master/slave modes.

                                        0: Master (default)
                                        1: Slave

                                        When in the Master Mode, the SSCLK1 pin is configured as an output pin. When in the Slave Mode, SSCLK1 signal is an input to the SSIF1.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SSDiv1 **01FF8105**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SSDiv1 **01FF8104**	SSInFull1	(Not Used)	(Not Used)	(Not Used)	SSCLK1 Divisor				Rst. Value 0xxx0000b Read Value 00h

Register Description: This register controls the bits of the SSCLK1 Divisor.

Bit 7:   SSInFull1, Read bit.   This bit indicates the status of the input register.

0: Empty (default)
1: Full

This bit allows the Firmware to poll the status of the serial interface when the use of the interrupt is not desired and is cleared by reading the in SSIF1 Data Register. SSInFull1 is active on the falling edge of AUXCLK1 and is used by Interrupt controller where it masked.

Bit 6-4:   Not used

Bit 3-0:   DivReg (Divisor Register), Read/Write bits.

The Div Register is the SSCLK1 Divisor Register, which defines the SSCLK1 frequency. Writing the DivReg register will cause the SSIF1 logic to load the internal counter. Firmware can transmit data with the right speed immediately after the SSCLK1 setup.

The SSCLK1 Divisor Register defines the SSCLK1 frequency, which depends on the AUXCLK frequency. The SSCLK1 Divisor Register is a five bit register. The SSCLK1 frequency (f_{SCK}) and Divisor relationship is:

$$F_{SCK} = \frac{AUXCLK}{2 * (Divisor + 1)}$$

19.2.2 The 2nd SSIF (SSIF2)

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SSData2 **01FF810B**	(Not Used)								Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SSData2 **01FF810A**	SSData2 (7-0)								Rst. Value xxh Read Value xxh

Register Description: SSIF2 Data Register

> *Note:*
>
> *The SSData2 Register serves as Transmit Shift Register (TxBuffer2) during CPU write, or as Receive Shift Register (RxBuffer2) during CPU read. Reading the SSData2 Register will clear the SSInFull2 Interrupt bit. Writing to the SSData2 Register will reset the shift bit counter, overwrite the output register and initiate an 8 bit serial shift while in serial mode.*
>
> *A bit called SSSTATOut2 is added into the SSCmd2 register. The SSSTATOut2 value will be output to the SSSTAT2/AO[2]/GPO[15] pin as the GPO[15] data when the SSSTAT2/GPO[15] pin is programmed as GPO[15].*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SSCmd2 **01FF8109**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	SSSTATOut2	Rst. Value xxxxxxx0b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SSCmd2 **01FF8108**	(Not Used)	(Not Used)	(Not Used)	L2MSB2	SSCLKPol2	SSCLKPhas2	(Not Used)	Mode12 0 - Master 1 - Slave	Rst. Value xxx000x0b Read Value 00h

Register Description: SSIF2 Command Register

Bit 8:		The SSSTATOut2 bit is used only when the SSSTAT2/AO[2]/GPO[15] pin is programmed as GPO[15]. The data value in the SSSTATOut2 bit is the GPO[15] data value.
Bit 4:	L2MSB2, Read/Write bit.	This bit controls the shifting order of the transmit and receive registers. When the L2MSB2 bit is cleared (default), the transmit and receive shift register will shift from the MSB to the LSB. When the L2MSB2 bit is set, the transmit and receive register will shift from the LSB to MSB.
Bit 3:	SSCLKPol2, Read/Write bit.	This bit controls the polarity of the SSCLK2 pin. The default value is zero. When the SSMode2 is configured at the IOMode2, the SSCLKPol2 bit will not affect the SSCLK2 pin value.
Bit 2:	SSCLKPhas2, Read/Write bit.	The default value is zero.

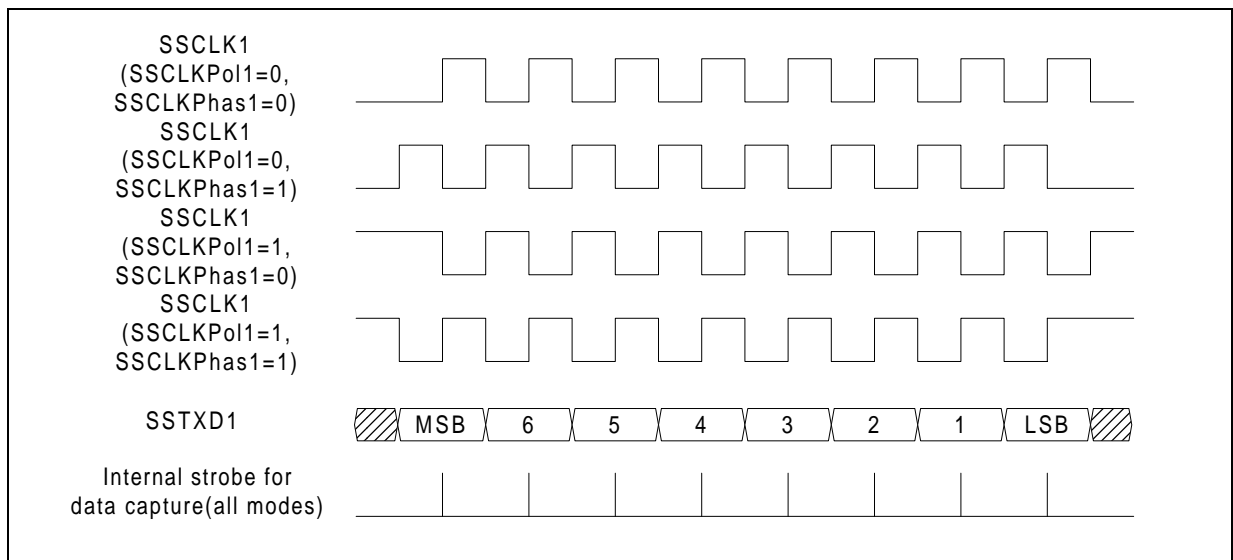| SSCLKPol2	SSCLKPhas2	Description (used for both master and slave modes)
0	0	Enable the external logic to clock the SSTXD2 pin on the rising edge of the SSCLK2 pin. The SSCLK2 pin's first transition will be in the middle of the first bit.
0	1	Enable the external logic to clock the SSTXD2 pin on the falling edge of the SSCLK2 pin. The SSCLK2 pin's first transition will occur at the beginning of the first bit.
1	0	Enable the external logic to clock the SSTXD2 pin on the falling edge of the SSCLK2 pin. The SSCLK2 pin's first transition will be in the middle of the first bit.
1	1	Enable the external logic to clock the SSTXD2 pin on the rising edge of the SSCLK2 pin. The SSCLK2 pin's first transition will occur at the beginning of the first bit.

Figure 19-3. SSCLK2 Diagram

Note: SSCLK2 is used to synchronize the movement of data both in and out of device through SSRXD2 and SSTXD2 pins. The master and slave devices are capable of exchanging a data byte of information during a sequence of eight clock pulses. Both master and slave devices must be operated in the same timing mode as controlled by SSCLKPol2 and SSCLKPhas2 signals.

Bit 0:   Mode12, Read/Write bit.   This bit controls the SSIF2 master/slave modes.

0: Master (default)
1: Slave

When in the Master Mode, the SSCLK2 pin is configured as an output pin. When in the Slave Mode, SSCLK2 signal is an input to the SSIF2.

Conexant

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
SSDiv2 **01FF810D**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
SSDiv2 **01FF810C**	SSInFull2	(Not Used)	(Not Used)	(Not Used)	SSCLK2 Divisor				Rst. Value 0xxx0000b Read Value 00h

Register Description: This register controls the bits of the SSCLK2 Divisor.

Bit 7:   SSInFull2, Read bit.   This bit indicates the status of the input register.

0: Empty (default)
1: Full

This bit allows the Firmware to poll the status of the serial interface when the use of the interrupt is not desired and is cleared by reading the in SSIF2 Data Register. SSInFull2 is active on the falling edge of AUXCLK and is used by Interrupt controller where it masked.

Bit 6-4:   Not used

Bit 3-0:   DivReg (Divisor Register), Read/Write bits.

The Div Register is the SSCLK2 Divisor Register, which defines the SSCLK frequency. Writing the DivReg register will cause the SSIF logic to load the internal counter. Firmware can transmit data with the right speed immediately after the SSCLK2 setup.

The SSCLK2 Divisor Register defines the SSCLK2 frequency, which depends on the AUXCLK frequency. The SSCLK2 Divisor Register is a 5-bit register. The SSCLK2 frequency (f_{SCK}) and Divisor relationship is:

TO BE ADDED

19.3 SSIF Timing

19.3.1 SSCLK Timing

When SSCLK is on (Figure 19-4), the duty cycle of SSCLK is always 50%. The SSCLK polarity is controlled by the SSIF Command Register SSCLKPol Bit. When the SSCLKPol bit and the phase control is cleared (default), the SSTXD signal transition is on the SSCLK falling edge. External logic can latch the SSTXD signal on the SSCLK rising edge. When the SSCLKPol bit is set, the SSTXD signal transition is on the rising edge of the SSCLK. External logic can latch the SSTXD signal on the SSCLK falling edge. For each write to the empty SSIF Data Register, the SSCLK will be active for eight cycles. When there is no data on the SSTXD pin, the SSCLK will stay in idle state, (logic zero when SSCLKPol bit is clear, and logic one when the SSCLKPol bit is set).

19.3.2 Request/Acknowledge Timing

Request/Acknowledge handshaking may be used while the SSIF is operating in the slave mode. While in this mode the peripheral (master) may initiate a data transfer by driving the SSACK signal low. The SSACK signal from the external peripheral is connected to any GPIO pin (used as GPI). Firmware will in turn write SSTXD data to the data register. This will cause the SSIF hardware to drive the SSREQ(SSSTAT) signal low. Once the SSREQ is low the peripheral may start the SSCLK to transfer the data. Upon completion of the 8 bit transfer, the peripheral must drive the SSACK high. The IFC will then read the serial data out of the input register which will in turn drive the SSREQ high and complete the transfer. If the IFC wishes to initiate a transfer in this mode it must write to the SSIF output register which will in turn set the SSREQ low. The peripheral must then provide 8 clocks on the SSCLK line to receive the data from the MFC2000's SSIF.
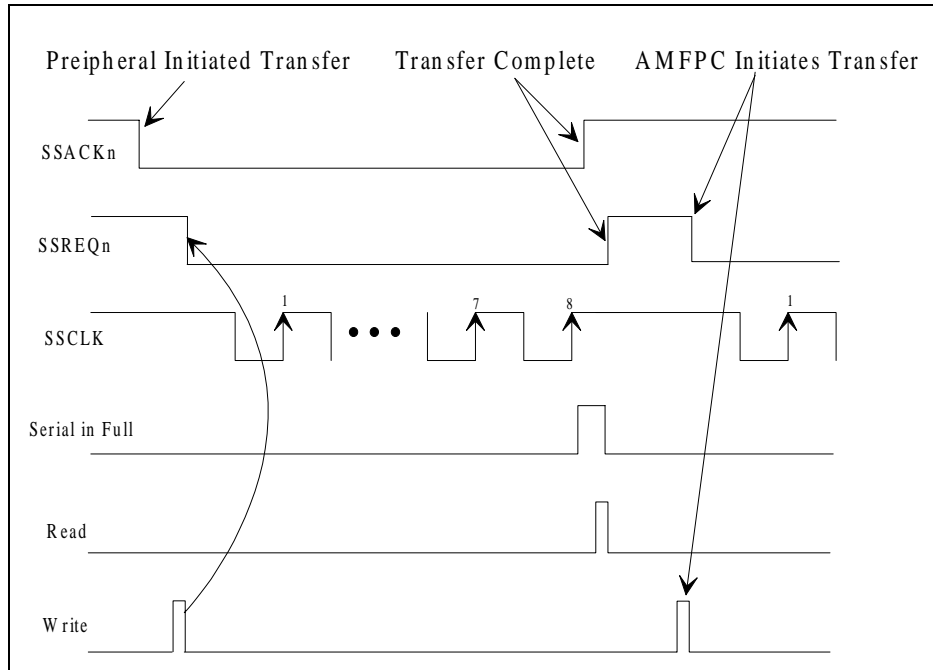
Figure 19-4. Timing Diagram

20. Programmable Tone Generators

20.1 Introduction

The AMFPC provides three programmable clock generator outputs. Two of the generators are to be used as tone generators and the third as a bell or ring driver.

20.2 Bell/Ringer Generator

The frequency of this bell/ringer generator is programmable, ranging from 500 Hz to 15.6 Hz. This frequency is derived from the AMFPC's internal 125µs period Clock, and it is calculated based on the following formula:

$$OutputFrequency = \frac{1}{125\mu s \times (\underline{BellPeriod} + 1) \times 16}$$, where <u>BellPeriod</u> is ranging from 0 to 31.

Each bell/ringer period is divided into 16 phases, as shown in the following figure. The phase number may be used by the firmware to determine when to turn off bell signal.
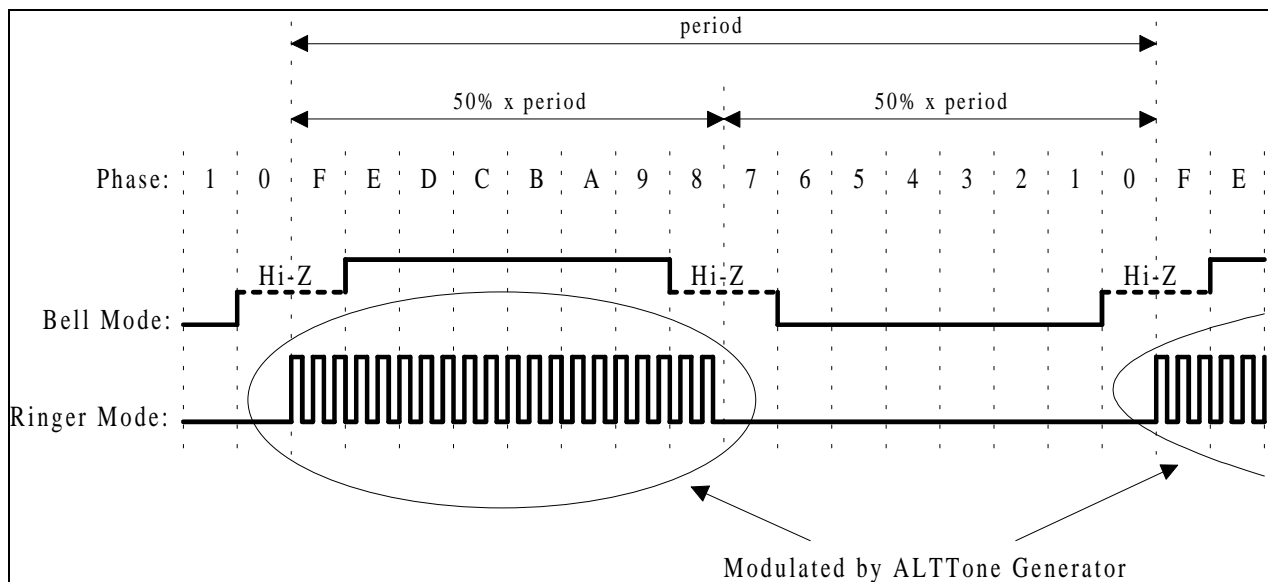
Figure 20-1. Bell/Ringer Timing

The mode of operation for this bell/ringer generator is selected by <u>RingerEnb</u> bit in BellControl register.

Figure 20-2 shows the block diagram of this bell/ringer generator.

Table 20-1 shows the relationship between BellPeriod value and output frequency.
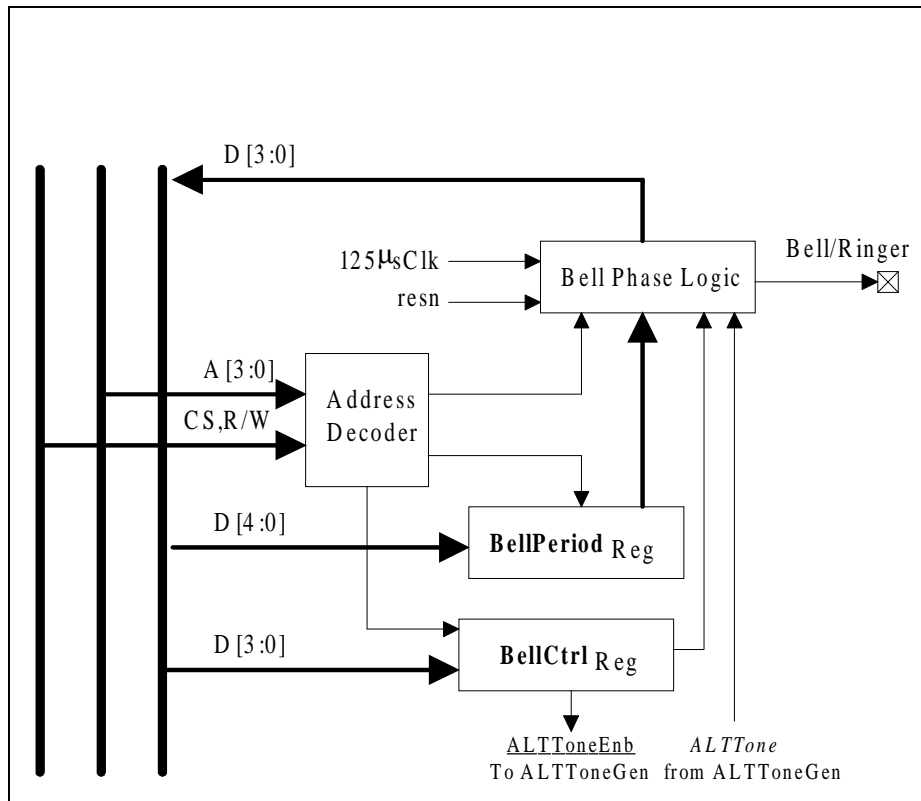
Figure 20-2. Bell/Ringer Block Diagram

Table 20-1. Bell/Ringer Setting

BellPeriod+1	Bell/Ringer Freq(Hz)		BellPeriod+1	Bell/Ringer Freq(Hz)
1	500		17	29.4
2	250		18	27.8
3	166.7		19	26.3
4	125		20	25
5	100		21	23.8
6	83.3		22	22.7
7	71.4		23	21.7
8	62.5		24	20.8
9	55.6		25	20
10	50		26	19.2
11	45.5		27	18.5
12	41.7		28	17.9
13	38.5		29	17.2
14	35.7		30	16.7
15	33.3		31	16.1
16	31.2		32	15.6

20.2.1 Bell Mode

Firmware operation

The firmware needs to control Bell/Ringer generator to issue the following waveform:
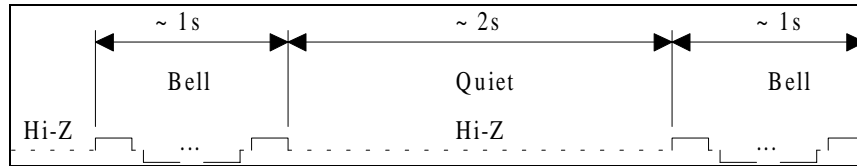
In order to setup for Bell mode, the following steps must be taken.

Initialize:

1. Write the desired frequency of bell in **BellPeriod** register (see Table 20-1)
2. Allow Hi-Z on output (BellDriveEnb = 0)
3. Disable ringer mode (RingerEnb = 0)

Start bell signal:

1. Enable Bell generator (BellEnb = 1)
2. (This setting will start the counter to run)

Create "Bell-Quiet-Bell" signal:

1. Once the bell signal has been generated, the firmware will read BellPeriod register. On every Phase A or 9, the off-hook signal is checked. If it is detected, then the bell signal is disabled by having BellEnb = 0. This will ensure the bell signal is terminated on the high to high-Z transition.
2. If after about 1s, the off-hook signal is not detected, then the firmware will disable the bell signal by having BellEnb = 0 when the phase is on 9. When BellEnb is reset to 0, then the bell signal will stop at the Hi-Z, and the phase will be in zero. After about 2 s of quiet, the bell signal can be activated again by setting BellEnb to 1.

20.2.2 Ringer Mode

The Ringer mode of operation requires the use of both the Bell/Ringer generator as well as the ALTTone generator. The fundamental frequency of this ringer signal is the same as the bell signal. The ringer signal works in conjunction with the ALTTone Generator to modulated it's output. The active window for the ring pulses (bell period high), will automatically shift so that the modulating waveform is not truncated at the beginning or end.

> **Note:** *The firmware needs to control Bell/Ringer generator to issue the following waveformn:*
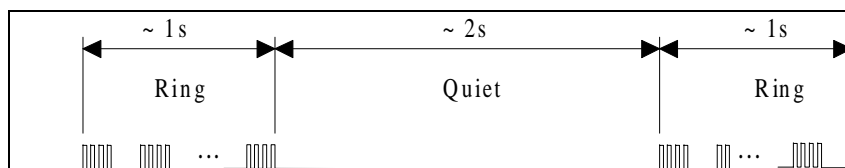
Figure 20-3. Bell/Ringer Generator Waveform

To setup for Ringer mode, the following steps must be taken:

Initialize:

1. Write the desired frequency of bell in BellPeriod register (see Table 20-1)
2. Enable output drive (BellDriveEnb = 1, will over-ride Hi Z control)
3. Enable ringer mode (RingerEnb = 1)
4. Write the desired frequency of tone1 in ALTToneGen register
5. If ALTTone signal is not desired on the ALTTone pin, then disable ALTTone signal to the pin (ALTToneEnb = 0).

> *Note*: *ALTToneEnb is only to enable the ALTTone signal to the ALTTone pin. If ALTToneEnb = 0, ALTTone signal will be blocked to ALTTone pin, but ALTTone signal is still available to Bell/Ringer generator for modulation purpose.*

Start ringer signal:

1. Enable Bell generator (BellEnb = 1)
2. (This setting will start the counter to run)

Create "Ring-Quiet-Ring" signal:

1. The firmware can generate the signal by switching the BellEnb setting.

20.2.3 Registers Description

The following is a description of the registers used to control the Bell/Ringer Function.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
BellPeriod **01FF80B7**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
BellPeriod **01FF80B6**	(Not Used)	(Not Used)	(Not Used)	Bell Period (0 - 31), bell period = (BellPeriod+1)*2ms					Rst. Value xxx00000b Read Value 00h

Register Description: Bell Period Register

Bit 15-5:   Not Used

Bit 4-0:   Writing to this register will set the Bell Period. Reading this register will return that value.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
BellPhase **01FF80B9**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
BellPhase **01FF80B8 (R)**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Bell Phase				Rst. Value xxxx0000b Read Value 00h

Register Description: Bell Phase Register

Bit 15-4:   Not Used

Bit 3-0:   Read only register returns current phase of the bell signal.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
BellCtrl **01FF80B5**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
BellCtrl **01FF80B4**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	ModToneEnb 1 = Enabled	RingerEnb 1=Ring Mode 0=Bell Mode	BellDrvEnb 1 = Enabled 0 = Allow Hi Z out	BellEnb 1 = Enabled	Rst. Value xxxx0000b Read Value 00h

Register Description: Bell Control Register

Bit 15-4:   Not Used

Bit 3:   Enables the ALTTone output to external pin but has no effect on the Ringer Mode. If this signal is set to 0 the ALTTone output will be at 0.

Bit 2:   Sets the generator into Bell Mode or Ringer Mode (Ringer Mode overrides bit 1 and does not allow the tristate condition.)

Bit 1:   In Bell Mode, it Controls weighter the output pin is always driven or tristated during bell phases 0, F, 8, and 7.

Bit 0:   Enables the Bell/Ringer counters to run. When this bit is set to 0, the counters are held at the start of phase "F".

> *Note:*   *After reset the Bell/Ringer pin will be tristated with the counters held at bell phase "0".*

20.3 Tone Generator

There are two tone generators which can be operated independently, namely ALTTone and Tone. ALTTone can be used alone or with the bell/ringer generator for modulation purpose. The duty cycle of the tone signal is 50%.

Frequency modulation on the tone signal can be controlled by the firmware. While the tone generator is running, the firmware may write a new value to the frequency register which will take effect on the start of the next tone cycle. The tone signal is guaranteed to have a smooth transition to the new frequency Figure 20-4 shows an example of increasing frequency with smooth transition on the tone signal.
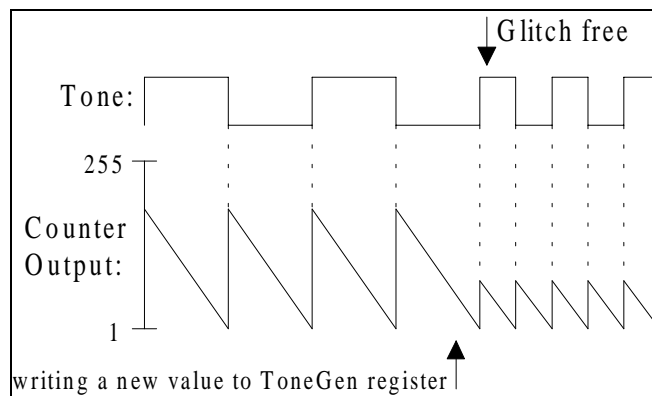
Figure 20-4. Tone Generator Frequency Change

20.3.1 Tone/AltTone

The signal generator called Tone, has the ability to generate a dual-tone output. The output frequency switch time and total cycle time are both programmable from 0 to 104.8mS with a step resolution of 102.4uS. Figure 1-5 is a block diagram of the Tone signal generator. Frequency Register 1 and 2 hold the value of the two alternating frequencies. Switch time specifies the time at which the output frequency stitches from Freq1 to Freq2. The total time register sets the total time of the repeating pattern. Figure 1-6 shows the timing diagram of when the output frequency counter switches between Freq1 and Freq2. Figure 1-7 depicts a waveform of a dual-tone output pattern showing what settings effect what portion of the output. Figure 1-8 is a block diagram of the AltTone frequency generator.
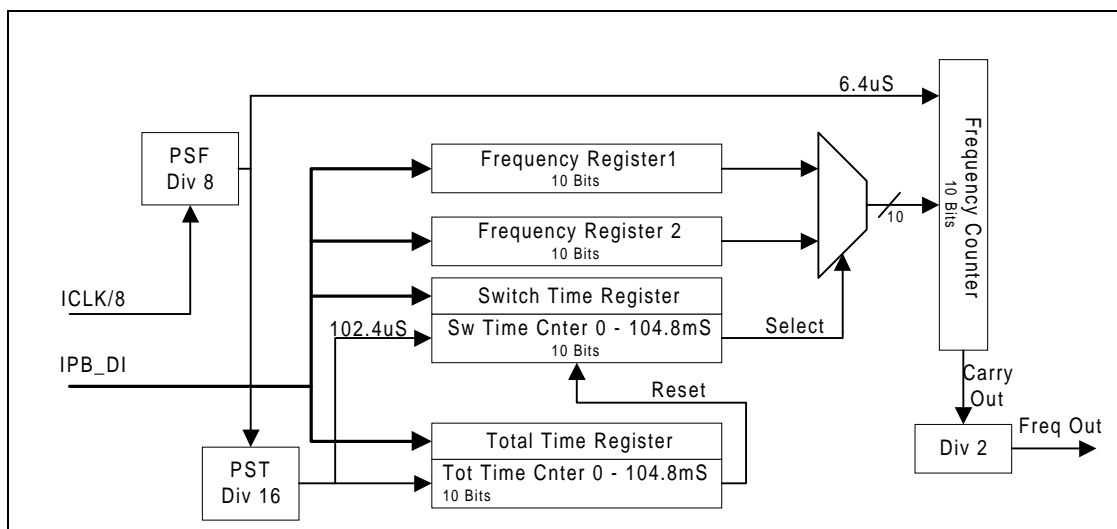
Figure 20-5. Dual-Tone Tone Generator Block Diagram
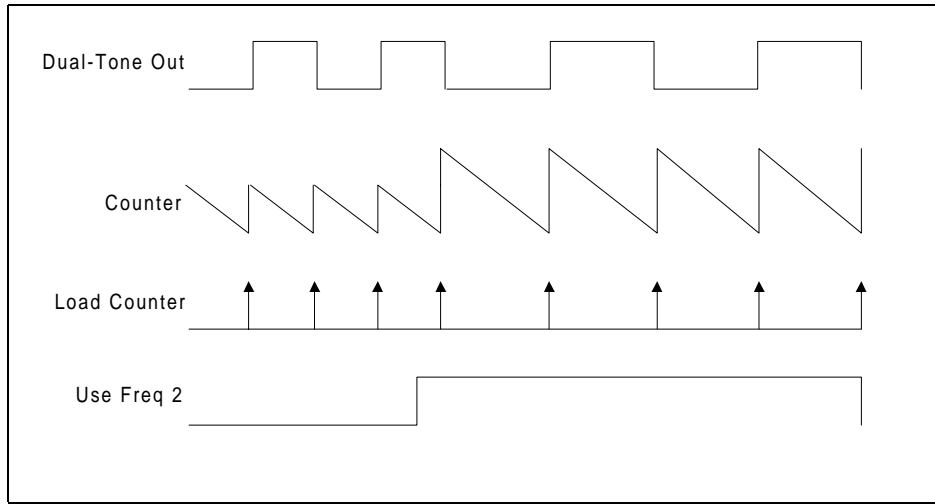
Figure 20-6. Dual-Tone Tone Generator Counter Operation
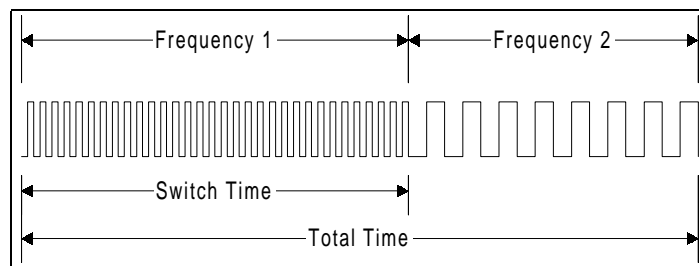
Figure 20-7. Dual-Tone Tone Generator output
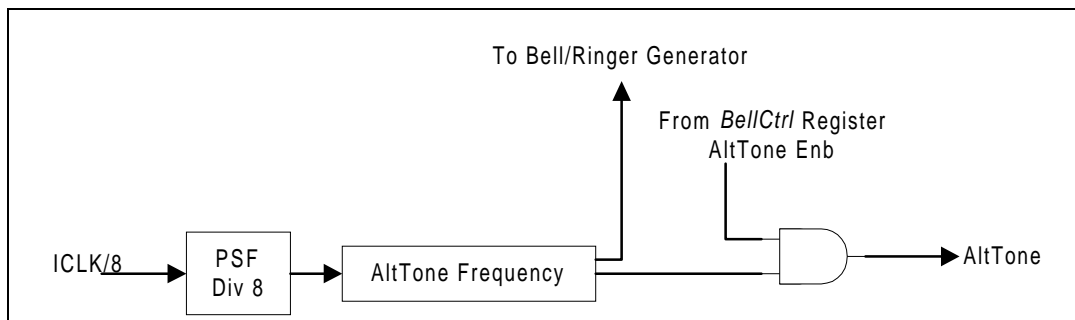
Figure 20-8. AltTone Generator Block Diagram

The frequency of the tone generators can be set according to the following formula:

$$ToneFrequency = \frac{AUXCLK}{(ToneGenValue)*8*8*2}$$

> *Note:*
>
> *ToneGenValue* is the value in **ALTToneGen** or **ToneGen** Freq1 and Freq2 registers.
>
> When *ToneGenValue* is zero, then the tone generator is disabled, and the output is low.

20.3.2 Registers Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ALTToneGen **01FF80B3**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	ALTTone Frequency Register		Rst. Value xxxxxx00b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ALTToneGen **01FF80B2**	ALTTone Frequency Register								Rst. Value 00h Read Value 00h

Register Description: ALTTone Generator Register. Read/Write register sets the frequency of ModTone.

Bit 9-0:   ALTTone frequency

> *Note:*   When ALT*ToneGen* = 0, then ALTTone is disabled and the ALTTone output to pin is 0.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ToneGenF1 01FF80B1	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Tone Frequency 1 Register		Rst. Value xxxxxx00b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ToneGenF1 01FF80B0	Tone Frequency 1 Register								Rst. Value 00h Read Value 00h

Register Description: Tone Generator Register. Read/Write register sets the frequency of Tone Frequency 1.

Bit 9-0:   Tone frequency 1

> *Note:*   When *Tone* = 0, then tone is disabled and the tone output to pin is 0.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ToneGenF2 **01FF80BB**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Tone Frequency 2 Register		Rst. Value xxxxxx00b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
ToneGenF2 **01FF80BA**	Tone Frequency 2 Register								Rst. Value 00h Read Value 00h

Register Description: Tone Generator Register. Read/Write register sets the frequency of Tone Frequency 2.

Bit 9-0:   Tone frequency 2

> ***Note:***   *When <u>Tone</u> = 0, then tone is disabled and the tone output to pin is 0.*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ToneGenSwitch **01FF80BD**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Tone Frequency Switch Time		Rst. Value xxxxxx00b Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
ToneGenSwitch **01FF80BC**	Tone Frequency Switch Time								Rst. Value 00h Read Value 00h

Register Description: Tone Generator Frequency Switch Register. Read/Write register sets the time of Frequency 1 duration.

Bit 9-0:   Tone Frequency Switch Time (0 to 104.8 mS in 102.4 uS steps)

If this register is set to 000h, the output frequency will = Freq2 for the Total Time. If the register value is >= the Total Time, the output frequency will = Freq1 for the Total Time.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ToneGenTotal **01FF80BF**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Tone Frequency Total Time		Rst. Value xxxxxx00b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ToneGenTotal **01FF80BE**	Tone Frequency Total Time								Rst. Value 00h Read Value 00h

Register Description: Tone Generator Total Time Register. Read/Write register sets the time of Frequency 1 plus Frequency 2 duration. When set to 000'h, The tone output will remain low or will be held low once the output switches to it's low state, and all counters will be held in reset (000'h). When this register is set to a non zero value, the counters will be released alowing generation of the tone signal.

The the end of the total time will not take effect until the output of the tone is low, preventing truncation of the ouput signal when high.

Bit 9-0:   Tone Frequency Total Time (0 to 104.8 mS in 102.4 uS steps)

21. Pwm Logic

21.1 Functional Description

There are five PWM channels, each channel having separate logic and control.

PWM channel 0 outputs from the GPIO[11]/CPCIN pin. There is a PWMCh0 select bit in the PWMCh0Duty Register. The PWMCh0 select bit goes to the GPIO block where the signal selection and the prioritization for the GPIO[11]/CPCIN/PWM0 pin is done.

PWM channel 1 and PWM channel 2 share pins with FCS[0]n and FCS[1]n. There is a PWMCh1 select bit in the PWMCh1Duty Register. There is a PWMCh2 select bit in the PWMCh2Duty Register.

PWM channel 3 shares the pin with GPIO[5]/CS[3]n. There is a PWMCh3 select bit in the PWMCh3Duty Register. The PWMCh3 select bit goes to the GPIO block where the signal selection and the prioritization for the GPIO[5]/CS[3]n/PWM3 pin is done.

PWM channel 4 shares the pin with GPIO[10]/P80_PB3. There is a PWMCh4 select bit in the PWMCh4Duty Register. The PWMCh4 select bit goes to the GPIO block. The GPIO logic does the signal selection and the prioritization for the GPIO[10]/P80_PB3/PWM4 pin.

The PWM function requirements are as follows.

- Signal frequency range → AUXCLK/256 or AUXCLK/(2*256)
- Duty cycle → high level: 1/256 to 256/256
- 8-bit counter for duty cycle programmability

21.2 Register Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
PWM Channel 0 Control *(PWMCh0Ctrl)* **01FF80C1h**	ALTTONE/ PWMCh0 Output Select 0=ALTTONE 1=PWMCh0	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	PWMCh0 Clock Divider	Rst. Value 0xxxxxx0b Read Value 0xxxxxx0b
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
PWM Channel 0 Control *(PWMCh0Ctrl)* **01FF80C0h**	Bit 7 of PWMCh0 Duty Cycle Control	Bit 6 of PWMCh0 Duty Cycle Control	Bit 5 of PWMCh0 Duty Cycle Control	Bit 4 of PWMCh0 Duty Cycle Control	Bit 3 of PWMCh0 Duty Cycle Control	Bit 2 of PWMCh0 Duty Cycle Control	Bit 1 of PWMCh0 Duty Cycle Control	Bit 0 of PWMCh0 Duty Cycle Control	Rst. Value 00h Read Value 00h

Bit 15   Controls which signal is output to the GPIO[11]/CPCIN/GPIO[11] pin (PWM channel 0 output signal, CPCIN input signal, or GPIO input/output signal).

Bit 8/GPIOConfig1 Register	Bit 15/PWMCh0Ctrl Register	GPIO[11]/CPCIN/PWM0 Pin
0	0	GPIO[11]
0	1	GPIO[11]
1	0	CPCIN
1	1	PWM0

Bit 8   PWMCh0 clock divider (n = 0-1)

$$\text{PWMCh0 clock} = \frac{AUXCLK}{(n+1)}$$

Bits 7-0   Duty cycle control (n = 0 - 255)

$$\text{PWMCh0 signal frequency} = \frac{PWMCh0\,clock}{256}$$

$$\text{PWMCh0 signal 'high' period} = \frac{n+1}{256} * \text{PWMCh0 signal cycle time}$$

$$\text{PWMCh0 signal 'low' period} = \frac{256-(n+1)}{256} * \text{PWMCh0 signal cycle time}$$

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
PWM Channel 1 Control *(PWMCh1Ctrl)* **01FF80C3h**	PWMCh1 Output Select	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	PWMCh1 Clock Divider	Rst. Value 0xxxxxx0b Read Value 0xxxxxx0b
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
PWM Channel 1 Control *(PWMCh1Ctrl)* **01FF80C2h**	Bit 7 of PWMCh1 Duty Cycle Control	Bit 6 of PWMCh1 Duty Cycle Control	Bit 5 of PWMCh1 Duty Cycle Control	Bit 4 of PWMCh1 Duty Cycle Control	Bit 3 of PWMCh1 Duty Cycle Control	Bit 2 of PWMCh1 Duty Cycle Control	Bit 1 of PWMCh1 Duty Cycle Control	Bit 0 of PWMCh1 Duty Cycle Control	Rst. Value 00h Read Value 00h

Bit 15   Controls which signal is output to the FCS[0]n pin (PWM channel 1 output signal or the other output signal).

Bit 8/FlashCtrl Register	Bit 15/PWMCh1Ctrl Register	FCS[0]n/PWM1 Pin
0	0	FCS[0]
0	1	FCS[0]
1	0	FCS[0] for NAND-type Flash Mem
1	1	PWM1

Bit 8   PWMCh1 clock divider (n = 0-1)

$$\text{PWMCh1 clock} = \frac{AUXCLK}{(n+1)}$$

Bits 7-0   Duty cycle control (n = 0 - 255)

$$\text{PWMCh1 signal frequency} = \frac{PWMCh1\,clock}{256}$$

$$\text{PWMCh1 signal 'high' period} = \frac{n+1}{256} * \text{PWMCh1 signal cycle time}$$

$$\text{PWMCh1 signal 'low' period} = \frac{256-(n+1)}{256} * \text{PWMCh1 signal cycle time}$$

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
PWM Channel 2 Control *(PWMCh2Ctrl)* **01FF80C5h**	PWMCh2 Output Select	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	PWMCh2 Clock Divider	Rst. Value 0xxxxxx0b Read Value 0xxxxxx0b
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
PWM Channel 2 Control *(PWMCh2Ctrl)* **01FF80C4h**	Bit 7 of PWMCh2 Duty Cycle Control	Bit 6 of PWMCh2 Duty Cycle Control	Bit 5 of PWMCh2 Duty Cycle Control	Bit 4 of PWMCh2 Duty Cycle Control	Bit 3 of PWMCh2 Duty Cycle Control	Bit 2 of PWMCh2 Duty Cycle Control	Bit 1 of PWMCh2 Duty Cycle Control	Bit 0 of PWMCh2 Duty Cycle Control	Rst. Value 00h Read Value 00h

Bit 15   Controls which signal is output to the FCS[1]n pin (PWM channel 2 output signal or the other output signal).

Bit 9/FlashCtrl Register	Bit 15/PWMCh2Ctrl Register	FCS[1]n/PWM2 Pin
0	0	FCS[1]
0	1	FCS[1]
1	0	FCS[1] for NAND-type Flash Mem
1	1	PWM2

Bit 8   PWMCh2 clock divider (n = 0-1)

$$\text{PWMCh2 clock} = \frac{AUXCLK}{(n+1)}$$

Bits 7-0   Duty cycle control (n = 0 - 255)

$$\text{PWMCh2 signal frequency} = \frac{PWMCh2clock}{256}$$

$$\text{PWMCh2 signal 'high' period} = \frac{n+1}{256} * \text{PWMCh2 signal cycle time}$$

$$\text{PWMCh2 signal 'low' period} = \frac{256-(n+1)}{256} * \text{PWMCh2 signal cycle time}$$

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
PWM Channel 3 Control *(PWMCh3Ctrl)* **01FF80C7h**	PWMCh3 Output Select	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	PWMCh3 Clock Divider	Rst. Value 0xxxxxx0b Read Value 0xxxxxx0b
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
PWM Channel 3 Control *(PWMCh3Ctrl)* **01FF80C6h**	Bit 7 of PWMCh3 Duty Cycle Control	Bit 6 of PWMCh3 Duty Cycle Control	Bit 5 of PWMCh3 Duty Cycle Control	Bit 4 of PWMCh3 Duty Cycle Control	Bit 3 of PWMCh3 Duty Cycle Control	Bit 2 of PWMCh3 Duty Cycle Control	Bit 1 of PWMCh3 Duty Cycle Control	Bit 0 of PWMCh3 Duty Cycle Control	Rst. Value 00h Read Value 00h

Bit 15   Controls which signal is output to the GPIO[5] pin (PWM channel 3 output signal or the other output signal).

Bit 4/GPIOConfig1 Register	Bit 15/PWMCh3Ctrl Register	GPIO[5]/CS[3]n/PWM3 Pin
0	0	GPIO[5]
0	1	GPIO[5]
1	0	CS[3]n
1	1	PWM3

Bit 8   PWMCh3 clock divider (n = 0-1)

$$\text{PWMCh3 clock} = \frac{AUXCLK}{(n+1)}$$

Bits 7-0   Duty cycle control (n = 0 - 255)

$$\text{PWMCh3 signal frequency} = \frac{PWMCh3clock}{256}$$

$$\text{PWMCh3 signal 'high' period} = \frac{n+1}{256} * \text{PWMCh3 signal cycle time}$$

$$\text{PWMCh3 signal 'low' period} = \frac{256-(n+1)}{256} * \text{PWMCh3 signal cycle time}$$

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
PWM Channel 4 Control *(PWMCh4Ctrl)* **01FF80C9h**	PWMCh4 Output Select	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	PWMCh4 Clock Divider	Rst. Value 0xxxxxx0b Read Value 0xxxxxx0b
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
PWM Channel 4 Control *(PWMCh4Ctrl)* **01FF80C8h**	Bit 7 of PWMCh4 Duty Cycle Control	Bit 6 of PWMCh4 Duty Cycle Control	Bit 5 of PWMCh4 Duty Cycle Control	Bit 4 of PWMCh4 Duty Cycle Control	Bit 3 of PWMCh4 Duty Cycle Control	Bit 2 of PWMCh4 Duty Cycle Control	Bit 1 of PWMCh4 Duty Cycle Control	Bit 0 of PWMCh4 Duty Cycle Control	Rst. Value 00h Read Value 00h

Bit 15 Controls which signal is output to the GPIO[10] pin (PWM channel 4 output signal or the other output signal).

Bit 7/GPIOConfig1 Register	Bit 15/PWMCh4Ctrl Register	GPIO[10]/P80_PB3/PWM4 Pin
0	0	GPIO[10]
0	1	GPIO[10]
1	0	P80_PB3
1	1	PWM4

Bit 8 PWMCh4 clock divider (n = 0-1)

$$\text{PWMCh4 clock} = \frac{AUXCLK}{(n+1)}$$

Bits 7-0 Duty cycle control (n = 0 - 255)

$$\text{PWMCh4 signal frequency} = \frac{PWMCh4\,clock}{256}$$

$$\text{PWMCh4 signal 'high' period} = \frac{n+1}{256} * \text{PWMCh4 signal cycle time}$$

$$\text{PWMCh4 signal 'low' period} = \frac{256-(n+1)}{256} * \text{PWMCh4 signal cycle time}$$

22. Calling Party Control (CPC)

In some countries, after the called party is on hook, a pulse is sent to the caller party by stopping the line current of the caller party for a short time to indicate that the called party is on hook, as seen in Figure 22-1. The duration of this pulse varies widely for different countries. The purpose of having Calling Party Control (CPC) circuit is to detect the *on-hook signal* by sampling CPC signal.
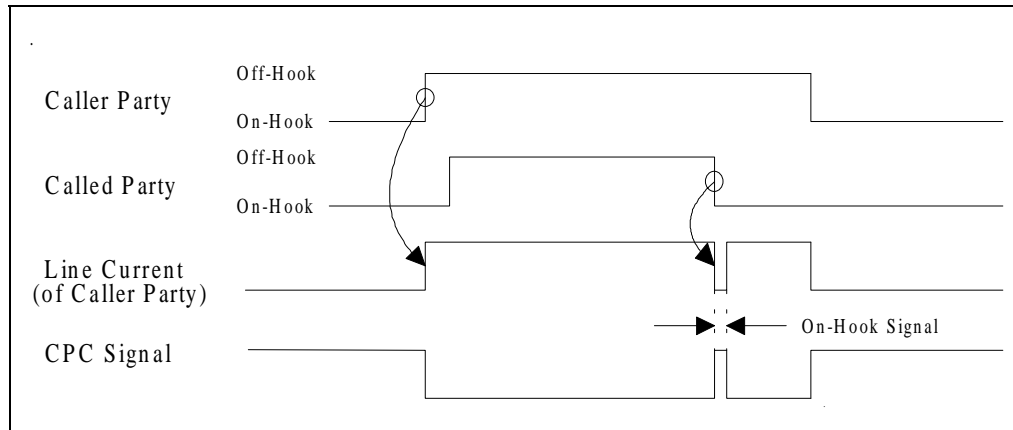
Figure 22-1: CPC Signal

Firmware Note*:*

In order to setup CPC circuit, the following initialization procedure must be done in the given order after *both parties are off hook:*

*1. Write the desired threshold in **CPCThreshold** register*
*2. Select the sampling period in **CPCStatCtrl** register (This will clear the CPCflag and reset CPC counter to zero)*

*Once CPC has been set up, the firmware can periodically check CPCflag by reading **CPCStatCtrl** register.*

Tthe following flowchart illustrates the CPC operation:
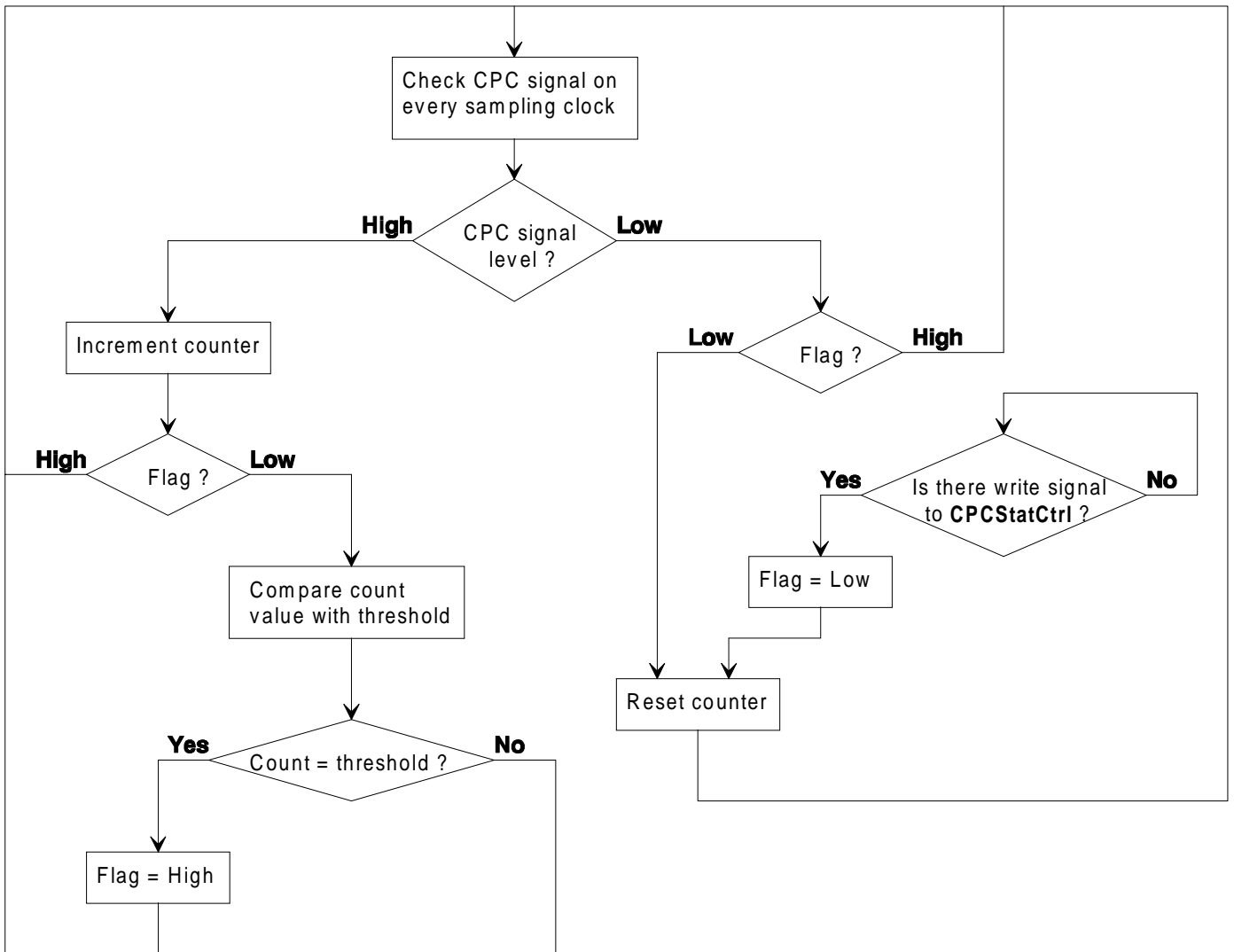
Figure 22-2. CPC Operation Flowchart

Figure 22-3 illustrates an example of the CPC operation with **CPCThreshold** containing a value of 4.
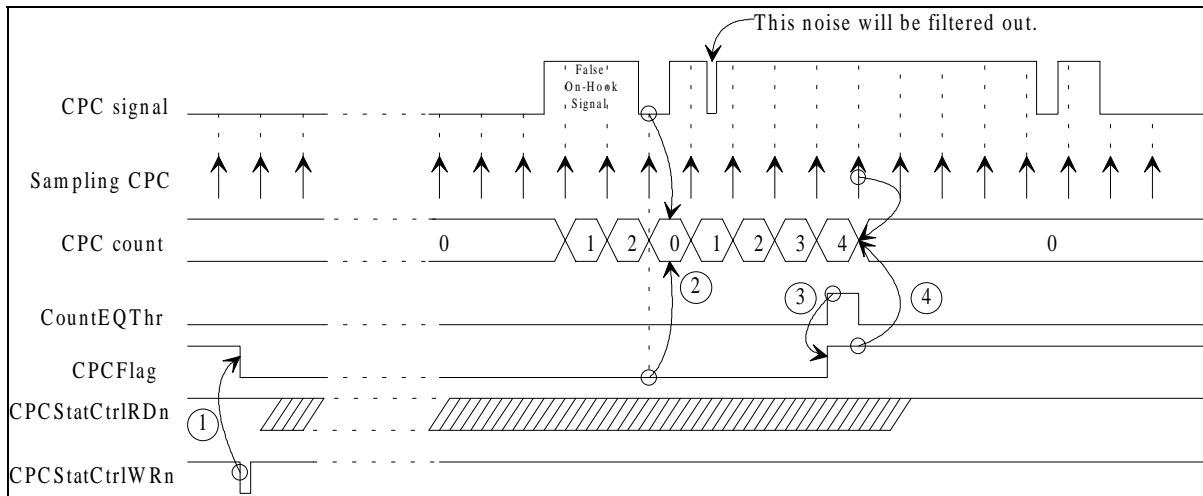
Figure 22-3: CPC Operation (with CPCThreshold = 4)

> *Note*:
>
> 1. *Writing to CPCStatCtrl register will clear CPC Flag.*
>
> 2. *Whenever both CPC signal and CPC Flag are low, the counter is reset in order to restart the counting sequence. The previous counter value is regarded as the result from a noise signal. This condition is only checked at every sampling time. In other words, if it happens between the sampling time, then the noise will be filtered out.*
>
> 3. *As soon as the counter value is equal to the threshold value, CPC Flag is raised.*
>
> 4. *Once CPC Flag has been raised, the counter will be clear on the next sampling clock.*

Figure 22-4 illustrates the CPC block diagram.
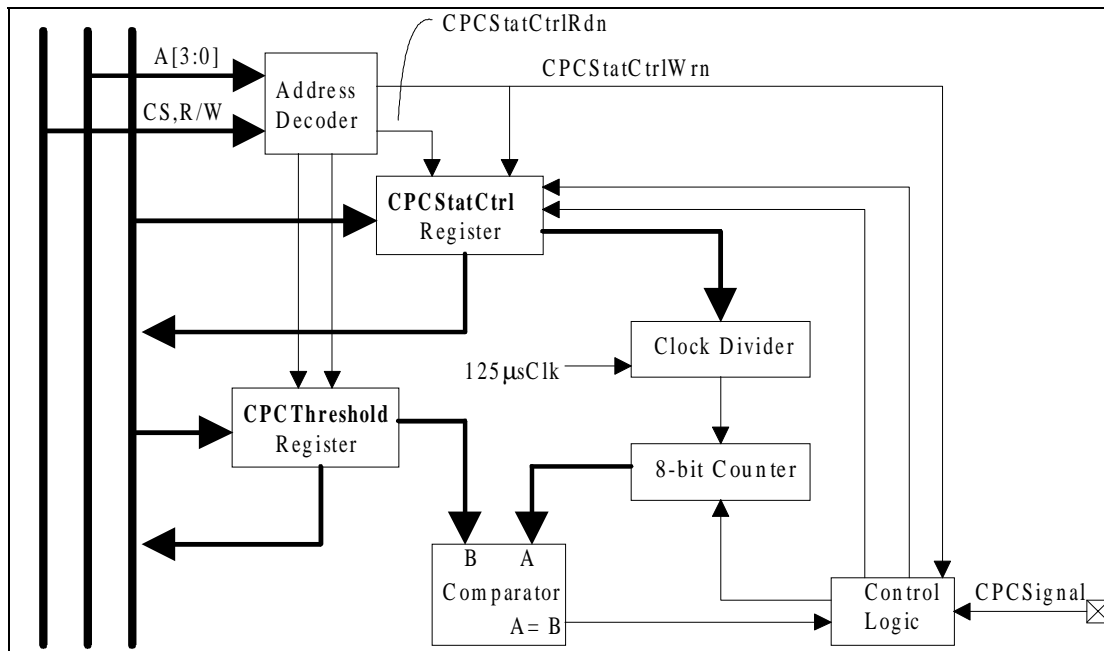
Figure 22-4: CPC Block Diagram

22.1 Registers Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
CPC Threshold (*CPCThreshold*) **01FF80A9**	(Not Used)								Rst Value xxh Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
CPC Threshold (*CPCThreshold*) **01FF80A8**	CPC Threshold								Rst Value FFh Read Value FFh

Bit 15–8   Not Used

Bit 7–0   CPC Threshold

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
CPC Status & Control (*CPCStatCtrl*) **01FF80AB**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst Value xxh Read Value 00h
Name/Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
CPC Status & Control (*CPCStatCtrl*) **01FF80AA**	CPC Flag (R)	CPC Signal (R)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Sampling Clock Select: 0: 125 μs 1: 500 μs 2: 2 ms 3: 8 ms		Rst Value xxxxxx11b Read Value 00000011b

Bit 15–8   Not Used

Bit 7   CPC Flag

Bit 6   CPC Signal

Bit 5–2   Not Used

Bit 1–0   Sampling Clock Selection

This page is intentionally blank

23. SSD_P80

23.1 Function Description

A 56K/33.6K data/fax modem core is integrated into the MFC2000 chip. It is called the P80 core. Furthermore, Conexant's SmartDAA technology is put into the MFC2000 chip. The P80 core can be connected to the phone line through either the modem IA/traditional DAA or the SmartDAA System Side Device (SSD) logic/the SmartDAA Line Side Device (LSD). The SSD logic is integrated into the MFC2000 chip. The SSD_P80 block includes the SSD logic, P80 core, and interface logic to the internal ARM bus system.

23.1.1 System Configurations

The SSD_P80 can be configured in the following ways.

• Configuration one: data acquisition is done only via SSD.
• Configuration two: data acquisition is done via SSD and voice acquisition is done via External IA.
• Configuration three: data acquisition is done only via External IA.
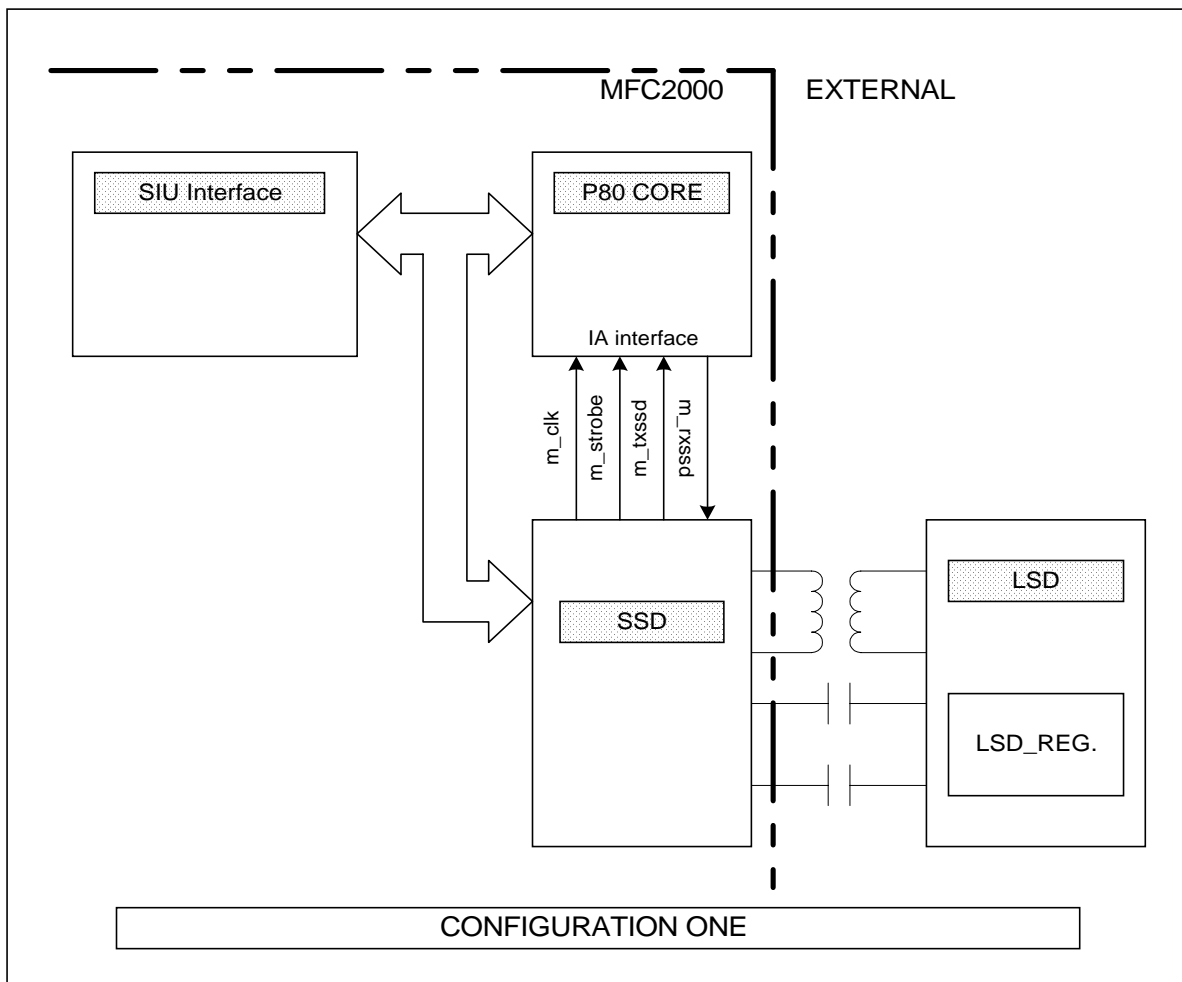
Figure 23-1. System Configuration One
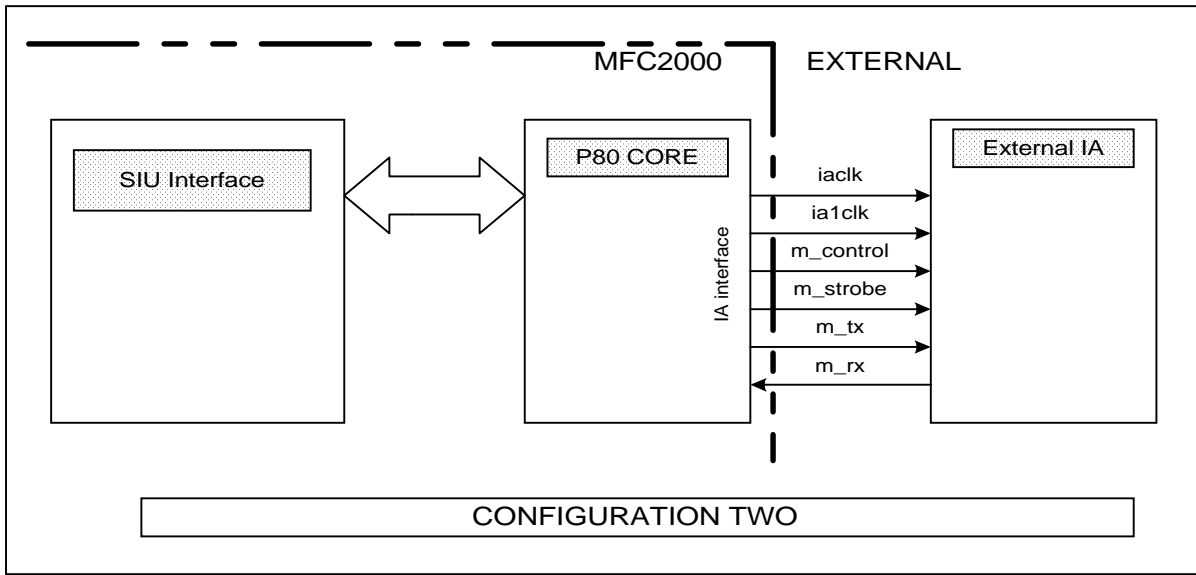
Figure 23-2. System Configuration Two
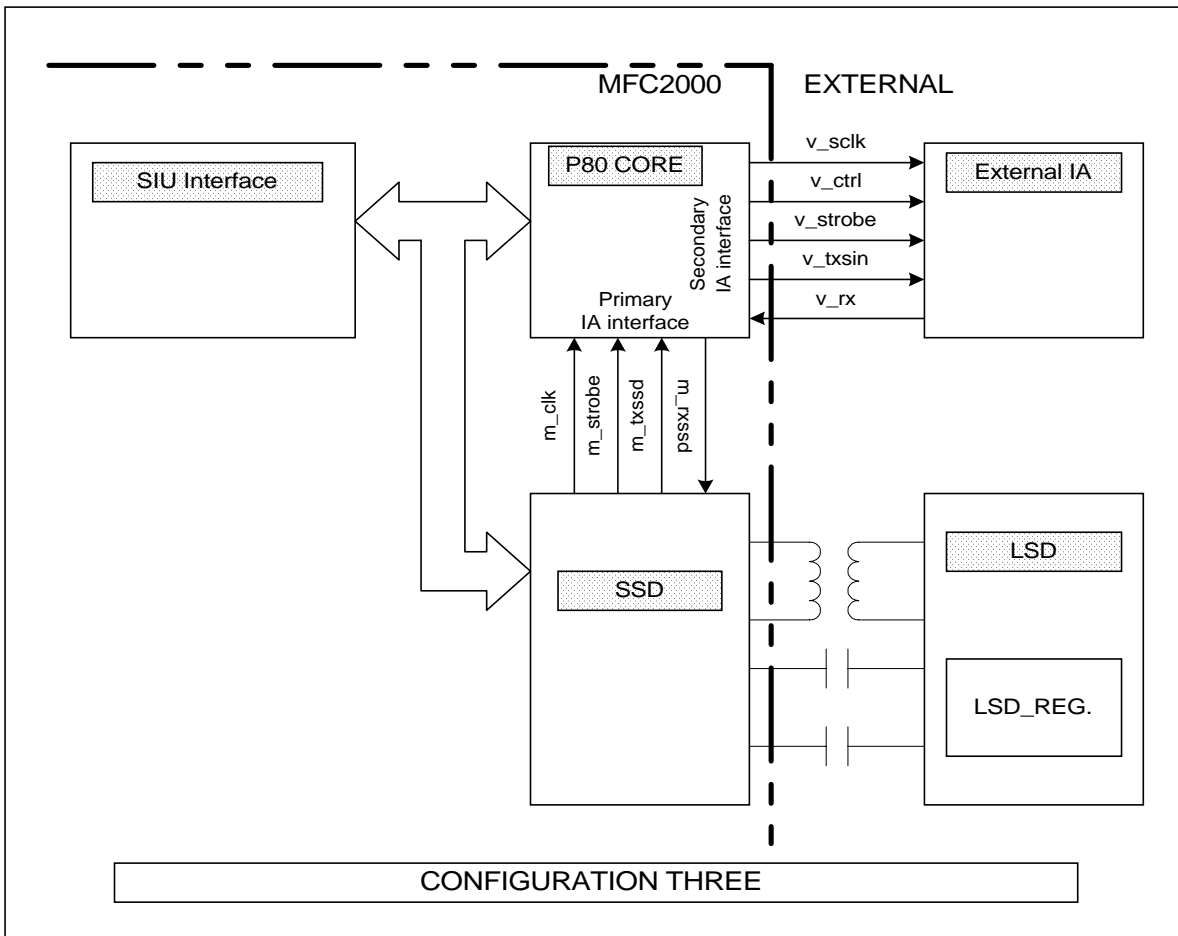
Figure 23-3. System Configuration Three

Conexant

23.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
P80CONTROL 0x01FF88E1	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	(not used)	Rst Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
P80CONTROL 0x01FF88E0	(not used)	SSD Ringwaken	SSD Raw_ringn	Ext Raw_ringn	Ring_Sel[1:0]		EXTIA_ MODE	IA_SEL	Rst Value xxxx0000b Read Value 00h

Bit 15:7   Not used

Bit 6   This is read only bit which indicates status of ring envelope from SSD.

Bit 5   This is a read only bit which indicates raw ring signals from SSD.

Bit 4   This is a read only bit which indicates raw ring signals from external DAA.

Bit 3:2   This control bits are used to select types of ring signals to be connected to P80_CORE to perform ring detection.

                                  "00" : No selection, logic 'high' will be driven to PB[3]_C of P80_CORE.
                                  "01" : SSD Raw_ringn is selected.
                                  "10" : SSD Ringwaken is selected.
                                  "11" : External DAA Raw_ringn is selected.

Bit 1   This bit controls muxing of output I/O pins. When asserted to high, selected P80 core pins will be directed to external pin.

Bit 0   This bit controls muxing of primary and secondary I/O pins in P80 core. When asserted to high, secondary I/O will be selected.

23.2.1 SSD Registers

> *Note*: *For detailed desciption on register contents, refer to SSD Spec.*

Table 23-1. SSD Registers

Register Name	MFC2K Address	Bit Positions	R/W	# Bits
SSD Register 00	00C30050	7:0	R/W	8
SSD Register 01	00C30052	7:0	R/W	8
SSD Register 02	00C30054	7:0	R/W	8
SSD Register 03	00C30056	7:0	R/W	8
SSD Register 04	00C30058	7:0	R/W	8
SSD Register 05	00C3005A	7:0	R/W	8
SSD Register 06	00C3005C	7:0	R/W	8
SSD Register 07	00C3005E	7:0	R/W	8
SSD Register 08	00C30060	7:0	R/W	8
SSD Register 09	00C30062	7:0	R/W	8
SSD Register 0A	00C30064	7:0	R/W	8
SSD Register 0B	00C30066	7:0	R/W	8
SSD Register 0C	00C30068	7:0	R/W	8
SSD Register 0D	00C3006A	7:0	R/W	8
SSD Register 0E	00C3006C	7:0	R/W	8
SSD Register 0F	00C3006E	7:0	R/W	8
SSD Register 10	00C30070	7:0	R/W	8
SSD Register 11	00C30072	7:0	R/W	8
SSD Register 12	00C30074	7:0	R/W	8
SSD Register 13	00C30076	7:0	R/W	8
SSD Register 14	00C30078	7:0	R/W	8
SSD Register 15	00C3007A	7:0	R/W	8
SSD Register 16	00C3007C	7:0	R/W	8
SSD Register 17	00C3007E	7:0	R/W	8
SSD Register 18	00C30080	7:0	R/W	8
SSD Register 19	00C30082	7:0	R/W	8
SSD Register 1A	00C30084	7:0	R/W	8
SSD Register 1B	00C30086	7:0	R/W	8
SSD Register 1C	00C3008E	7:0	R/W	8

23.2.2 P80 CORE Registers

> *Note*: For detailed description on register contents, refer to FM336 hardware spec.

Table 23-2. P80 CORE Registers

Register Name	MFC2K Address	Bit Positions	R/W	# Bits	
P80 Register 00	00C30000	7:0	R/W	8	
P80 Register 01	00C30002	7:0	R/W	8	
P80 Register 02	00C30004	7:0	R/W	8	
P80 Register 03	00C30006	7:0	R/W	8	
P80 Register 04	00C30008	7:0	R/W	8	
P80 Register 05	00C3000A	7:0	R/W	8	
P80 Register 06	00C3000C	7:0	R/W	8	
P80 Register 07	00C3000E	7:0	R/W	8	
P80 Register 08	00C30010	7:0	R/W	8	
P80 Register 09	00C30012	7:0	R/W	8	
P80 Register 0A	00C30014	7:0	R/W	8	
P80 Register 0B	00C30016	7:0	R/W	8	
P80 Register 0C	00C30018	7:0	R/W	8	
P80 Register 0D	00C3001A	7:0	R/W	8	
P80 Register 0E	00C3001C	7:0	R/W	8	
P80 Register 0F	00C3001E	7:0	R/W	8	
P80 Register 10	00C30020	7:0	R/W	8	
P80 Register 11	00C30022	7:0	R/W	8	
P80 Register 12	00C30024	7:0	R/W	8	
P80 Register 13	00C30026	7:0	R/W	8	
P80 Register 14	00C30028	7:0	R/W	8	
P80 Register 15	00C3002A	7:0	R/W	8	
P80 Register 16	00C3002C	7:0	R/W	8	
P80 Register 17	00C3002E	7:0	R/W	8	
P80 Register 18	00C30030	7:0	R/W	8	
P80 Register 19	00C30032	7:0	R/W	8	
P80 Register 1A	00C30034	7:0	R/W	8	
P80 Register 1B	00C30036	7:0	R/W	8	
P80 Register 1C	00C30038	7:0	R/W	8	
P80 Register 1D	00C3003A	7:0	R/W	8	
P80 Register 1E	00C3003C	7:0	R/W	8	
P80 Register 1F	00C3003E	7:0	R/W	8	

23.3 Firmware Operation

Note: ALL SSD and P80 CORE registers are 8-Bit registers.

24. Countach Imaging DSP Bus Subsystem

The Countach Imaging DSP Bus Subsystem contains the Countach Imaging DSP subsystem for video/scan image signal processing and all peripherals/memory needed for the whole operations. The main memory for image data storage is the external DRAM/SDRAM. These peripherals are Countach Subsystem Interface, Scan IA, Video/Scan Controller, SDRAM Controller, Video/Scan Interface, ARM Bus Interface, CDMA Controller, and Countach Bus Unit. The Countach Imaging DSP subsystem should operate at 100 MHz or 85.7MHz. This is the separate bus system from the ARM bus system and runs in parallel with the ARM bus system to get maximum performance out of the MFC2000. ARM Bus Interface logic is the bridge between the ARM bus system and the Countach Imaging DSP Bus Subsystem. Both CPU access and DMA access mastered by the ARM side are supported to coordinate the entire system operation.

Several blocks are used to complete the connection between these peripherals:

1. ARM Bus Interface—connects the ARM subsystem to the internal busses.
2. Countach Subsystem Interface—connects the Countach core to the internal busses.
3. Sync. DRAM Controller—connects the external DRAM/SDRAM to the internal busses.
4. Video/Scan Interface—connects the Video/Scan Controller to the internal busses.

Several blocks are used for internal functions:

1. Countach Bus Unit—connects all internal blocks in the Countach Bus Subsystem together.
2. Countach DMA Controller—paces all DMA transaction between the Video/Scan Interface for video capture and scanner, ARM Bus Interface for ARM bus system, and the Countach Subsystem Interface for the Countach subsystem to and from the SDRAM.

Several I/O accesses on the Countach Bus Subsystem are planned:

1. ARM accesses the scratch pad of the Countach subsystem through ARM Bus Interface, Countach Bus Unit, and Countach Subsystem IF.

> *Note*: *ARM will be able to access all registers in the sub-block of the Countach Bus Subsystem through the IPB bus of the ARM bus system.*
>
> *The ARM must not access the scratchpad while DMA to or from the Countach Imaging DSP Subsystem is in progress. Failure to do so can result in corruption and/or loss of DMA data. The ARM may access the scratchpad during all other DMA transfers, including video/scanner data to SDRAM and SDRAM to and from ARM DRAM.*

2. Put the DMA setting data to the Countach DMA Controller from the scratch pad of the Countach subsystem through Countach Bus Unit, and Countach Subsystem IF after the Countach Bus Unit is informed by the GPO signal from Countach Subsystem.

Several DMA accesses on the Countach Bus Subsystem are planned:

1. The DMA operation to send video and scan image data to the external SDRAM/DRAM from the Video/Scan Controller through the Video/Scan Interface.
2. The DMA operation to send/receive data from/to memory on the ARM bus system through ARM Bus Interface.
3. The DMA operation to send/receive data from/to the scratch pad of the Countach subsystem through Countach Subsystem Interface.
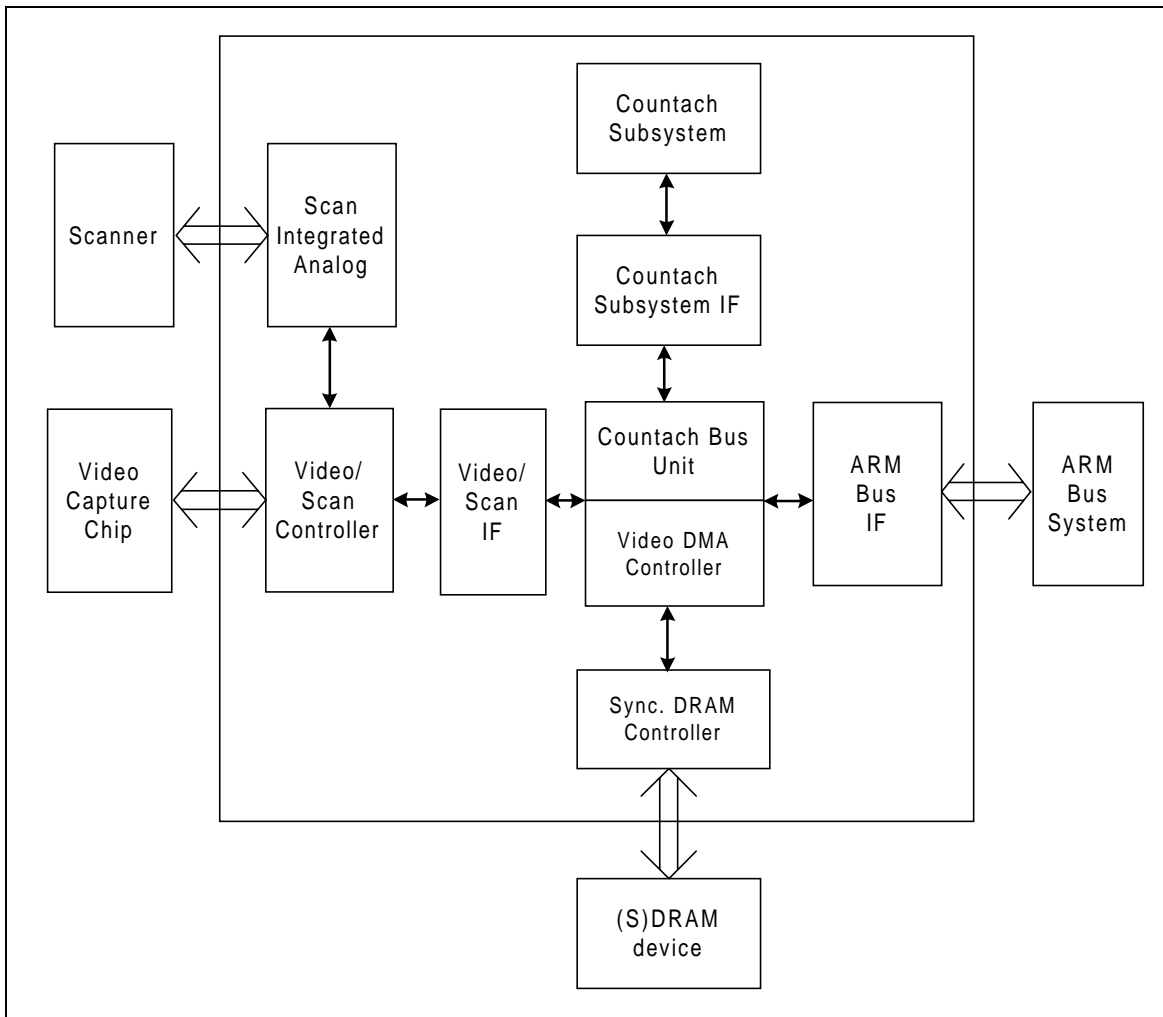
Figure 24-1. The ARM Bus System Block Diagram

24.1 Countach Imaging DSP Subsystem

24.1.1 Description

The Countach Imaging DSP Subsystem is mainly responsible for monochrome and color image processing and JPEG compression and decompression. The Countach Imaging DSP subsystem contains the Countach core, data RAM and ROM, program RAM and program ROM, scratch pad, and DMA channels.

24.1.2 Countach Imaging DSP Subsystem Memory

Program Memory:

- $2K \times 40$ RAM
- ROM for program initialization

Data RAM:

- $1K \times 16$ register file RAM
- $4K \times 16$ DMA-accessible RAM
- $4K \times 16$ RAM
- 8K x 16 ROM

24.1.3 Countach Imaging DSP Subsystem DMA

Data DMA:

- 4 channels
- DMA accesses are made via the scratch pad interface
- Countach Imaging DSP controls which bank DMA transfers occur in – DMA channel does not automatically switch after completion of a block transfer
- Consecutive DMA cycles must be able to occur at a minimum rate of 6 Countach clocks per transfer

Program DMA:

- Minimum of 1 channel
- DMA accesses are made via the scratch pad interface
- Consecutive DMA cycles must be able to occur at a minimum rate of 6 Countach clocks per transfer
- An entire instruction is formed by transferring it in three 16-bit segments to scratch pad addresses 0x1b0, 0x1b1, and 0x1b2

24.1.4 Running Frequency

The Countach Imaging DSP Subsystem operates at either 87.5MHz or 100 MHz. It is configurable through the CLK_CONFIG[2] signal which is muxed on the ROMCSn pin.

24.2 COUNTACH IMAGING DSP BUS UNIT

24.2.1 Function Description

The Countach Bus Unit is the conduit for all data and address between the various block of the Countach Bus System. It also decides which blocks become master of these busses.

There are four busses in the Countach Bus System:

- CPB—address and data to and from the Countach Subsystem Interface
- DPB—address and data to and from the Synchronous DRAM Interface
- IPB—address and data to and from the ARM Bus Interface
- VPB—data from the Video/Scan Interface

Because of these separate busses (as opposed to a single tri-state muxed bus), it is possible to have multiple bus masters. The possible connections between these busses are:

- IPB ↔ CPB, VPB ↔ DPB: ARM Bus Interface makes discrete I/O accesses to Countach Subsystem scratch pad while Video/Scan Interface makes DMA accesses to Synchronous DRAM Interface.
- CPB ↔ DPB: Countach Subsystem Interface makes DMA accesses to Synchronous DRAM Interface.
- IPB ↔ DPB, {Countach DMA Controller} ↔ cpb: ARM Bus Interface makes either discrete I/O accesses or DMA accesses to Synchronous DRAM Interface while Countach DMA Controller fetches DMA parameters from Countach Subsystem scratch pad.

24.2.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Deferred Read High Address *(DefRdHiAddr)* $01FF82A9	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Deferred Read High Address *(DefRdHiAddr)* $01FF82A8	(Not Used)	Deferred Read Address [22]	Deferred Read Address [21]	Deferred Read Address [20]	Deferred Read Address [19]	Deferred Read Address [18]	Deferred Read Address [17]	Deferred Read Address [16]	Rst Value x0000000b Read Value 00h

Bit 15-7　　　　　　　　　　　　　Not used

Bit 6-0　　Deferred Read Address [22:16]　These seven bits, in conjunction with the sixteen bits of the Deferred Read Low Address register, make up the deferred read address. It is a halfword address, not a byte address.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Deferred Read Low Address *(DefRdLoAddr)* **$01FF82AB**	Deferred Read Address [15]	Deferred Read Address [14]	Deferred Read Address [13]	Deferred Read Address [12]	Deferred Read Address [11]	Deferred Read Address [10]	Deferred Read Address [9]	Deferred Read Address [8]	Rst. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Deferred Read Low Address *(DefRdLoAddr)* **$01FF82AA**	Deferred Read Address [7]	Deferred Read Address [6]	Deferred Read Address [5]	Deferred Read Address [4]	Deferred Read Address [3]	Deferred Read Address [2]	Deferred Read Address [1]	Deferred Read Address [0]	Rst Value 00h Read Value 00h

Bit 15-0   Deferred Read Address [15:0]   These sixteen bits, in conjunction with the seven bits of the Deferred Read High Address register, make up the deferred read address. It is a halfword address, not a byte address.

Notes:

1. *The deferred read address is a halfword address in the Countach Bus Subsystem address space. Countach DRAM address space is $000000–$3FFFFF and Countach Subsystem scratch pad address space is $400000–$4001FF.*

2. *Writing this register begins the deferred access operation. This register must be the last one written when initiating a deferred read access.*

3. *During a deferred read access, all deferred registers are locked out from ARM writes.*

4. *When the deferred read access completes, an interrupt is generated via the Countach DMA Done bit in the ARM Bus Interface Interrupt Status register.*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Deferred Write High Address *(DefWrHiAddr)* **$01FF82AD**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Deferred Write High Address *(DefWrHiAddr)* **$01FF82AC**	(Not Used)	Deferred Write Address [22]	Deferred Write Address [21]	Deferred Write Address [20]	Deferred Write Address [19]	Deferred Write Address [18]	Deferred Write Address [17]	Deferred Write Address [16]	Rst Value x0000000b Read Value 00h

Bit 15-7   Not used

Bit 6-0   Deferred Write Address [22:16]   These seven bits, in conjunction with the sixteen bits of the Deferred Write Low Address register, make up the deferred write address. It is a halfword address, not a byte address.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Deferred Write Low Address *(DefWrLoAddr)* **$01FF82AF**	Deferred Write Address [15]	Deferred Write Address [14]	Deferred Write Address [13]	Deferred Write Address [12]	Deferred Write Address [11]	Deferred Write Address [10]	Deferred Write Address [9]	Deferred Write Address [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Deferred Write Low Address *(DefWrLoAddr)* **$01FF82AE**	Deferred Write Address [7]	Deferred Write Address [6]	Deferred Write Address [5]	Deferred Write Address [4]	Deferred Write Address [3]	Deferred Write Address [2]	Deferred Write Address [1]	Deferred Write Address [0]	Rst Value 00h Read Value 00h

Bit 15-0   Deferred Write Address [15:0]   These sixteen bits, in conjunction with the seven bits of the Deferred Write High Address register, make up the deferred write address. It is a halfword address, not a byte address.

Notes:

3. The deferred write address is a halfword address in the Countach Bus Subsystem address space. Countach DRAM address space is $000000–$3FFFFF and Countach Subsystem scratch pad address space is $400000–$4001FF.

4. Writing this register begins the deferred access operation. This register must be the last one written when initiating a deferred write access.

5. During a deferred write access, all deferred registers are locked out from ARM writes.

6. When the deferred write access completes, an interrupt is generated via the Countach DMA Done bit in the ARM Bus Interface Interrupt Status register.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Deferred Read Data (R) *(DefRdData)* **$01FF82B1**	Deferred Read Data [15] (R)	Deferred Read Data [14] (R)	Deferred Read Data [13] (R)	Deferred Read Data [12] (R)	Deferred Read Data [11] (R)	Deferred Read Data [10] (R)	Deferred Read Data [9] (R)	Deferred Read Data [8] (R)	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Deferred Read Data (R) *(DefRdData)* **$01FF82B0**	Deferred Read Data [7] (R)	Deferred Read Data [6] (R)	Deferred Read Data [5] (R)	Deferred Read Data [4] (R)	Deferred Read Data [3] (R)	Deferred Read Data [2] (R)	Deferred Read Data [1] (R)	Deferred Read Data [0] (R)	Rst Value 00h Read Value 00h

Bit 15-0   Deferred Read Data   This read-only register contains the deferred read data at the completion of a deferred read access operation. Its contents are valid when the Countach DMA Done bit in the ARM Bus Interface Interrupt Status register is set.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Deferred Write Data *(DefWrData)* **$01FF82B3**	Deferred Write Data [15]	Deferred Write Data [14]	Deferred Write Data [13]	Deferred Write Data [12]	Deferred Write Data [11]	Deferred Write Data [10]	Deferred Write Data [9]	Deferred Write Data [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Deferred Write Data *(DefWrData)* **$01FF82B2**	Deferred Write Data [7]	Deferred Write Data [6]	Deferred Write Data [5]	Deferred Write Data [4]	Deferred Write Data [3]	Deferred Write Data [2]	Deferred Write Data [1]	Deferred Write Data [0]	Rst Value 00h Read Value 00h

Bit 15-0   Deferred Write Data   This register contains the data to be written during a deferred write access operation.

***Notes**:*

This register must be written before beginning the deferred write access via a write to the Deferred Write Low Address register.

During a deferred write access, all deferred registers are locked out from ARM writes.

24.3 ARM BUS INTERFACE

24.3.1 Function Description

The Arm Bus interface block attaches the Countach Bus Subsystem to the ARM Bus System. The ARM can directly access the various registers of the Countach Bus Subsystem. It can access the Countach Bus Subsystem's SDRAM and the Countach Imaging DSP subsystem's scratch pad memory and the SDRAM with discrete I/O accesses with some wait state penalty. It can also access the SDRAM via DMA accesses. To accommodate the ARM DMA interchange, two channels are provided: one for transfers into the SDRAM and one for transfers out of the SDRAM.

In addition, two handshaking signals are provided for interrupting of each processor and allow the ARM to interrupt the Countach core. It is implemented through a register in the Countach Subsystem Interface. The arm_attn signal comes directly from the Countach core and is an input to the ARM interrupt controller.

24.3.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus Interface Interrupt Status *(ABIIrqStat)* **$01FF8283**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
ARM Bus Interface Interrupt Status *(ABIIrqStat)* **$01FF8282**	(Not Used)	(Not Used)	Countach Subsystem Interrupt	VSI Overrrun	Deferred Access Done	C2A DMA Done	Countach DMA Done	Video/ Scan DMA Done	Rst Value xx000000b Read Value 00h

Bit 15-6		Not used
Bit 5	Countach Subsystem Interrupt	Countach Subsystem has interrupted the ARM either in IRQP or IRQP2, depending on the setting of the Countach Interrupt Source bit of the ABICountachCtrl register.
Bit 4	VSI Overrun	The second VSI ping-pong buffer has been filled before the first has been emptied, resulting in loss of data.
Bit 3	Deferred Access Done	A deferred access to/from scratch pad or SDRAM has completed. If a deferred read was requested, the data in the Deferred Read Access Data register is now valid.
Bit 2	C2A DMA Done	C2A (SDRAM to Arm Bus System) block size has been reached.
Bit 1	Countach DMA Done	DMA operation specified by Countach scratch pad parameters has completed.
Bit 0	Video/Scan DMA Done	Two-dimensional DMA from Video/Scan Interface to SDRAM has completed (two fields) or one scan line has completed.

> **Note**: *Each of these bits are cleared by writing it to a one. This allows individual interrupts to be cleared without affecting the others.*

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus Interface Interrupt Enable *(ABIIrqEnable)* $01FF8285	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
ARM Bus Interface Interrupt Enable *(ABIIrqEnable)* $01FF8284	(Not Used)	(Not Used)	Countach Interrupt Enable	VSI Overrrun Interrupt Enable	Deferred Access Interrupt Enable	C2A DMA Interrupt Enable	Countach DMA Interrupt Enable	Video/ Scan DMA Interrupt Enable	Rst Value xx000000b Read Value 00h

Bit 15-6   Not used

Bit 5   Countach Interrupt Enable   Enables interrupt when the Countach has interrupted the ARM Bus via the IRQP or IRQP2 signals.

Bit 4   VSI Overrun Interrupt Enable   Enables interrupt when the second VSI ping-pong buffer has been filled before the first has been emptied, indicating in loss of data.

Bit 3   Deferred Access Interrupt Enable

Enables interrupt when a deferred access to/from scratch pad or SDRAM has completed.

Bit 2   C2A DMA Interrupt Enable   Enables interrupt when C2A (SDRAM to Arm Bus System) block size has been reached.

Bit 1   Countach DMA Interrupt Enable

Enables interrupt when DMA operation specified by Countach scratch pad parameters has completed.

Bit 0   Video/Scan DMA Interrupt   Enables interrupt when two-dimensional DMA from Video/Scan Interface to SDRAM has completed (two fields) or one scan line has been completed.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus Interface Countach Control (W) *(ABICountachCtrl)* $01FF8287	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
ARM Bus Interface Countach Control (W) *(ABICountachCtrl)* $01FF8286	(Not Used)	(Not Used)	Countach Interrupt Source	Reset Countach Bus Subsystem (W)	Start Countach DMA (W)	Abort Countach DMA (W)	Reset Countach Subsystem (W)	Interrupt Countach Subsystem (W)	Rst Value xxxx0000b Read Value 00h

Bit 15-6   Not used

Bit 5   Countach Interrupt Source   This bit selects which interrupt output from the Countach DSP Subsystem is used to interrupt the ARM via the Countach Interrupt bit in the ARM Bus Interface Interrupt Status Register. A value of 0 selects the IRQP signal and a value of 1 selects the IRQP2 signal.

Bit 4   Reset Countach Bus Subsystem

Writing a one to this bit resets the entire Countach Bus Subsystem (CBSS.)

Bit 3	Start Countach DMA	Writing a one to this bit causes the Countach DMA controller to fetch parameters from scratch pad and begin a DMA transfer.
Bit 2	Abort Countach DMA	Writing a one to this bit causes any Countach DMA operation (CDMA channel 1) to be aborted. The Countach receives the Countach DMA Done interrupt after the DMA has aborted (EXTIRQ2).
Bit 1	Reset Countach Subsystem	Writing a one to this bit resets the Countach Subsystem and the Countach Subsystem Interface which contains the DMA FIFOs.
Bit 0	Interrupt Countach Subsystem	Writing a one to this bit generates an interrupt to the Countach Subsystem (EXTIRQ).

24.3.3 Internal Memory

The ARM Bus Interface contains two 4×16 buffers for DMA transfers between memory attached to the ARM Bus Subsystem and the Countach SDRAM. These buffers can be accessed by the ARM as either bytes or halfwords.

The two buffers are located at the following addresses:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM to Countach DMA Buffer *(A2CDmaBuf)* **$01FF830x**	A2C Buffer Data [15]	A2C Buffer Data [14]	A2C Buffer Data [13]	A2C Buffer Data [12]	A2C Buffer Data [11]	A2C Buffer Data [10]	A2C Buffer Data [9]	A2C Buffer Data [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM to Countach DMA Buffer *(A2CDmaBuf)* **$01FF830x**	A2C Buffer Data [7]	A2C Buffer Data [6]	A2C Buffer Data [5]	A2C Buffer Data [4]	A2C Buffer Data [3]	A2C Buffer Data [2]	A2C Buffer Data [1]	A2C Buffer Data [0]	Rst Value 00h Read Value 00h

$01FF8300-01	A2C DMA Buffer halfword 0
$01FF8302-03	A2C DMA Buffer halfword 1
$01FF8304-05	A2C DMA Buffer halfword 2
$01FF8306-07	A2C DMA Buffer halfword 3

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Countach to ARM DMA Buffer *(C2ADmaBuf)* **$01FF830x**	C2A Buffer Data [15]	C2A Buffer Data [14]	C2A Buffer Data [13]	C2A Buffer Data [12]	C2A Buffer Data [11]	C2A Buffer Data [10]	C2A Buffer Data [9]	C2A Buffer Data [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Countach to ARM DMA Buffer *(C2ADmaBuf)* **$01FF830x**	C2A Buffer Data [7]	C2A Buffer Data [6]	C2A Buffer Data [5]	C2A Buffer Data [4]	C2A Buffer Data [3]	C2A Buffer Data [2]	C2A Buffer Data [1]	C2A Buffer Data [0]	Rst Value 00h Read Value 00h

$01FF8308-09	C2A DMA Buffer halfword 0
$01FF830A-0B	C2A DMA Buffer halfword 1
$01FF830C-0D	C2A DMA Buffer halfword 2
$01FF830E-0F	C2A DMA Buffer halfword 3

ARM access to the ABI buffers are provided for diagnostic purposes only and should not be used as part of normal operation.

When the ARM makes an access to these buffers, the ARM is held off and clock domain synchronization occurs before the access is completed. As a result, a significant wait state penalty is incurred. This penalty will be even longer if a local DMA operation is in progress when the access is made as the ARM is held off for both the clock domain synchronization and the completion of the local DMA transfer.

There is a risk of data corruption when the ARM makes write accesses to these buffers while a DMA transfer is in progress. This can occur because the buffers alternate betweeen a DMA transfer sequence in the ARM Bus Subsystem domain and a DMA transfer sequence in the Countach Bus Subsystem domain. If the write operation occurs between these DMA transfers, the DMA data in the buffer will be replaced with the ARM data.

24.4 Countach Imaging DSP Subsystem Interface

ARM as either bytes or halfwords.

The buffer is located at the following addresses:

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Countach DMA Buffer (CSIDmaBuf) $01FF830x	CSI Buffer Data [15]	CSI Buffer Data [14]	CSI Buffer Data [13]	CSI Buffer Data [12]	CSI Buffer Data [11]	CSI Buffer Data [10]	CSI Buffer Data [9]	CSI Buffer Data [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Countach DMA Buffer (CSIDmaBuf) $01FF830x	CSI Buffer Data [7]	CSI Buffer Data [6]	CSI Buffer Data [5]	CSI Buffer Data [4]	CSI Buffer Data [3]	CSI Buffer Data [2]	CSI Buffer Data [1]	CSI Buffer Data [0]	Rst Value 00h Read Value 00h

Table 24-1. Needs a title

$01FF8300-01	CSI DMA Buffer halfword 0
$01FF8302-03	CSI DMA Buffer halfword 1
$01FF8304-05	CSI DMA Buffer halfword 2
$01FF8306-07	CSI DMA Buffer halfword 3
$01FF8306-07	CSI DMA Buffer halfword 4
$01FF8306-07	CSI DMA Buffer halfword 5
$01FF8306-07	CSI DMA Buffer halfword 6
$01FF8306-07	CSI DMA Buffer halfword 7

ARM access to the CSI buffers are provided for diagnostic purposes only and should not be used as part of normal operation.

When the ARM makes an access to these buffers, the ARM is held off and clock domain synchronization occurs before the access is completed. As a result, a significant wait state penalty is incurred. This penalty will be even longer if a local DMA operation is in progress when the access is made as the ARM is held off for both the clock domain synchronization and the completion of the local DMA transfer.

There is a risk of data corruption when the ARM makes write accesses to this buffer while a DMA transfer is in progress. It is also possible that an access to this buffer can cause spurious writes to the scratchpad space while a DMA transfer is in progress.

24.5 COUNTACH DMA CONTROLLER

24.5.1 Function Description

This block handles data transfers between several of the blocks of the Countach subsystem.

It has three potential bus masters: the Video/Scan Interface, the Arm Bus interface, and the Countach subsystem.

The VDMA employs a hierarchical round-robin arbitration scheme, with the Video/Scan Interface block at the highest priority and the Countach subsystem and Arm Bus interface in a round-robin scheme at the lowest priority. A round-robin arbitration scheme is one in which no requestor can be serviced until all other requestors at the same priority have been serviced if they are also requesting. A hierarchical arbitration scheme is one in which if a higher priority group is requesting service it is given priority over a lower priority group, after which the lower priority group is handled in its predetermined manner.

Below is a table of the four DMA channels, their functionality, and priorities.

Table 24-2. DMA Channels: Functionality and Priorities

Chan	Usage	++	—	2D	⇒	Th	✕	Priority	Description
0	VSI	✓		✓	✓			A-1	Scan/video data to (S)DRAM
1	CSI I/O	✓	✓	✓	✓		✓	B-1	Countach scratch pad data to/from (S)DRAM
2	ABI in	✓			✓	✓		B-2	ARM data to (S)DRAM
3	ABI out	✓			✓	✓	✓	B-3	ARM data from (S)DRAM

Legend:	
Chan	DMA channel
VSI	Video/Scanner Interface
CSI	Countach Subsystem Interface
ABI	ARM Bus Interface
++	Incrementing address
—	Decrementing address
2D	Address Jumping at boundaries.
⇒	Burst cycles
Th	Throttle: Limits number of DMA Cycles per CBSS clocks
✕	Block limit

Notes:	
(1)	To be supplied.

Channels 0,2, and 3 operate in a similar fashion as existing channels in the ARM DMA logic. These channels are programmed by ARM-accessible registers and contain address registers and, for channel 3, a block limit and enable register.

Channel 1 is very different in its operation. It contains no ARM-accessible registers. It is controlled by the Countach Subsystem. However, since the Countach Subsystem cannot be a bus master and program the registers directly, the Countach DMA logic must fetch the register values from the Countach Subsystem scratch pad.

The Countach Subsystem has no DMA handshaking signals as it is able to accept data at a fixed rate. DMA data is transferred to and from the Countach Subsystem via the scratch pad at fixed addresses. Thus, it appears as memory (SDRAM) to I/O (scratch pad) transfers.

Channel 1 is shared by five logical DMA channels within the Countach Subsystem, but they operate consecutively, not concurrently (i.e., DMA for one logical channel is completely before DMA for another logical channel is begun. Finally, logical DMA channel 4 (of physical DMA channel 1) is used for program DMA. Because

Countach instructions are 40 bits wide, they must be transferred in sections. This is accomplished by DMA transfers to three consecutive scratch pad addresses.

Each of the five logical channels is started by the Countach Subsystem by a high-going pulse on the Countach GPIO signal sp_gpio[0]. When the pulse is detected, the Countach DMA Controller fetches the parameters from the scratch pad, executes the desired DMA operation, and interrupts the Countach Subsystem at the completion.

The scratch pad addresses for the various DMA parameters are shown below:

Table 24-3. DMA Parameters Scratch Pad Addresses

Scratch pad address	Parameter
Countach Subsystem DMA Channel Parameters	
0xF9	ENB – Enables the DMA operation
0xFA	BAH – most significant bits of base address and DMA control bits
0xFB	BAL – least significant bits of base address
0xFC	SZ – number of halfwords to transfer
0xFD	NSR – number of sub ranges
0xFE	AST – address step between sub ranges
Scratch pad address	**Function**
0x100	Countach Subsystem DMA channel 0 data port
0x101	Countach Subsystem DMA channel 1 data port
0x102	Countach Subsystem DMA channel 2 data port
0x103	Countach Subsystem DMA channel 3 data port
0x1B0	Countach Subsystem DMA channel 4 program data port (bits 15:0)
0x1B1	Countach Subsystem DMA channel 4 program data port (bits 31:16)
0x1B2	Countach Subsystem DMA channel 4 program data port (bits 39:32)

To program and initiate a DMA transfer, firmware must first setup the parameter registers shown above. The process of reading the ENB register will cause hardware to generate either an ARM IRQ or start a DMA transfer. If the LSB of the ENB Register is 1 an ARM IRQ is generated. If the value of the LSB is 0, a DMA transfer is initiated.
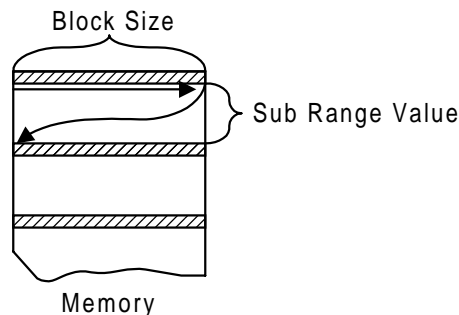
This diagram depicts how the SZ, NSR and AST register all work together. The block size is controlled by the SZ register and defines the number of DMA accesses before the Sub Range Jump occurs. At the end of the block, the last DMA address + one is added to the AST register, and is used as the starting address for the next block. This process continues for the number of cycles specified by the NSR register. Once the NSR value is met, an IRQ is generated.

The meaning of the ENB, BAH, BAL, SZ, NSR, and AST parameters are shown below:

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Not used			Program /Data	Logical Data Channel Number		Address direction	Data direction
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Not used			Base address (21:16)				

Parameter Description: BAH parameter

Bit 15-13		Not used
Bit 12		When high, logical channel 4 is requested (program DMA.) When low, one of logical channels 0-3 is requested as specified by the Logical Channel Number field (data DMA.)
Bit 11-10	Logical Data Channel Number	When bit 12 is low, this field indicates which logical channel of 0-3 is to be used (data DMA.) When bit 12 is high, this field is not used.
Bit 9	Address direction	When low, the DMA address should be incremented after each halfword is transferred. When high, the DMA address should be decremented after each halfword is transferred.
Bit 8	Data direction	When low, DMA data is read from Countach scratch pad space. When high, DMA data is written to Countach scratch pad space.
Bit 7-6		Not used
Bit 5-0		Upper bits of DMA base address

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Base address (15:8)							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Base address (7:0)							

Parameter Description: BAL parameter

Bit 15-8   Lower bits of DMA base address

This parameter should only be fetched if the Enable channel bit of the BAH parameter is high.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Block size (15:8)							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Block size (7:0)							

Parameter Description: SZ parameter (Refer to the above diagram)

Bit 15-0   Block size in halfwords   Block size = Block Size Register + 1.

If this parameter is zero, the channel should be considered improperly configured and not enabled. Also, no further parameters need to be fetched in this circumstance. This register programs the data block size (number of DMA accesses) between address steps, AST.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Number of sub ranges (15:8)							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Number of sub ranges (7:0)							

Parameter Description: NSR parameter (Refer to the above diagram)

Bit 15-0   Number of sub ranges   Total number of sub ranges = register value + 1; When this register is set to $0000, one block will be transferred before the IRQ occurs. This register controls the number of number of sub ranges (NSR), or address steps that occur in the complete DMA transfer.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Address step between sub ranges (15:8)							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Address step between sub ranges (7:0)							

Parameter Description: AST parameter (Refer to the above diagram)

Bit 15-0   Address step between sub ranges

This parameter should only be fetched if the Enable channel bit of the BAH parameter is high. This register controls the address step size between data blocks. The register value is added to the last DMA address accessed in a block plus 1.

24.5.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan High Address One *(VsHiAddr1)* **$01FF828D**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Video/Scan High Address One *(VsHiAddr1)* **$01FF828C**	(Not Used)	Video /Scan Addr[22]	Video /Scan Addr[21]	Video /Scan Addr[20]	Video /Scan Addr[19]	Video /Scan Addr[18]	Video /Scan Addr[17]	Video /Scan Addr[16]	Rst Value x0000000b Read Value 00h

- The value written to this register will not take effect until the lower address value is written. Therefore this value should be set first.
- Reading this register will return the Address Counter status, not the value written to this register.

Bit 15-7   Not Used

Bit 6-0   Video/Scan Interface High Address One [22:16]

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan Low Address One *(VsLoAddr1)* **$01FF828F**	Video /Scan Addr[15]	Video /Scan Addr[14]	Video /Scan Addr[13]	Video /Scan Addr[12]	Video /Scan Addr[11]	Video /Scan Addr[10]	Video /Scan Addr[9]	Video /Scan Addr[8]	Rst. Value 00h Read Value 00h
Address:	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default:**
Video/Scan Low Address One *(VsLoAddr1)* **$01FF828E**	Video /Scan Addr[7]	Video /Scan Addr[6]	Video /Scan Addr[5]	Video /Scan Addr[4]	Video /Scan Addr[3]	Video /Scan Addr[2]	Video /Scan Addr[1]	(Not Used)	Rst Value 0000000xb Read Value 00h

- Writing to the Video/Scan Low Address Register will load the address counter. Therefore the upper address value should be set first.
- Reading this register will return the Address Counter status, and not the value written to this register.

Bit 15-1   Video/Scan Interface Low Address One [15:1]

Bit 0   Not Used

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan High Address Two *(VsHiAddr2)* **$01FF8291**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Video/Scan High Address Two *(VsHiAddr2)* **$01FF8290**	(Not Used)	Video /Scan Addr[22]	Video /Scan Addr[21]	Video /Scan Addr[20]	Video /Scan Addr[19]	Video /Scan Addr[18]	Video /Scan Addr[17]	Video /Scan Addr[16]	Rst Value x0000000b Read Value 00h

Bit 15-7   Not Used

Bit 6-0   Video/Scan Interface High Address Two [22:16]

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan Low Address Two *(VsLoAddr2)* **$01FF8293**	Video /Scan Addr[15]	Video /Scan Addr[14]	Video /Scan Addr[13]	Video /Scan Addr[12]	Video /Scan Addr[11]	Video /Scan Addr[10]	Video /Scan Addr[9]	Video /Scan Addr[8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Video/Scan Low Address Two *(VsLoAddr2)* **$01FF8292**	Video /Scan Addr[7]	Video /Scan Addr[6]	Video /Scan Addr[5]	Video /Scan Addr[4]	Video /Scan Addr[3]	Video /Scan Addr[2]	Video /Scan Addr[1]	(Not Used)	Rst Value 0000000xb Read Value 00h

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan High Address Step *(VSHiAddrStep)* **$01FF8295**	(Not used)	(Not used)	(Not used)	(Not used)	(Not used)	(Not used)	(Not used)	(Not used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Video/Scan High Address Step *(VSHiAddrStep)* **$01FF8294**	(Not used)	Video /Scan Step [22]	Video /Scan Step [21]	Video /Scan Step [20]	Video /Scan Step [19]	Video /Scan Step [18]	Video /Scan Step [17]	Video /Scan Step [16]	Rst Value x0000000b Read Value 00h

Bit 15-5   Not Used

Bit 6-0   Video/Scan Interface High Address Step [22:16]

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan Low Address Step *(VSLoAddrStep)* **$01FF8297**	Video /Scan Step[15]	Video /Scan Step[14]	Video /Scan Step [13]	Video /Scan Step [12]	Video /Scan Step [11]	Video /Scan Step [10]	Video /Scan Step [9]	Video /Scan Step [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Video/Scan Low Address Step *(VSLoAddrStep)* **$01FF8296**	Video /Scan Step [7]	Video /Scan Step [6]	Video /Scan Step [5]	Video /Scan Step [4]	Video /Scan Step [3]	Video /Scan Step [2]	Video /Scan Step [1]	(Not used)	Rst Value 0000000xb Read Value 00h

Bit 15-1   Video/Scan Interface Low Address Step [15:1]

Bit 0   Not Used

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scan Mode *(VSMode)* **$01FF8299**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Video/Scan Mode *(VSMode)* **$01FF8298**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Video /Scan Select	Rst Value xxxxxxx0b Read Value 00h

Bit 15-1   Not Used

Bit 0   Video/Scan select   A zero selects video capture mode and a one selects scan mode.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus A2C High Address *(Aba2cHiAddr)* **$01FF829D**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus A2C High Address *(Aba2cHiAddr)* **$01FF829C**	(Not Used)	ARM Bus Input Addr[22]	ARM Bus Input Addr[21]	ARM Bus Input Addr[20]	ARM Bus Input Addr[19]	ARM Bus Input Addr[18]	ARM Bus Input Addr[17]	ARM Bus Input Addr[16]	Rst Value x0000000b Read Value 00h

Bit 15-7   Not Used

Bit 6-0   ARM Bus Interface Input (A2C) Data High Address [22:16]

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus A2C Low Address *(Aba2cLoAddr)* **$01FF829F**	ARM Bus Input Addr[15]	ARM Bus Input Addr[14]	ARM Bus Input Addr[13]	ARM Bus Input Addr[12]	ARM Bus Input Addr[11]	ARM Bus Input Addr[10]	ARM Bus Input Addr[9]	ARM Bus Input Addr[8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus A2C Low Address *(Aba2cLoAddr)* **$01FF829E**	ARM Bus Input Addr[7]	ARM Bus Input Addr[6]	ARM Bus Input Addr[5]	ARM Bus Input Addr[4]	ARM Bus Input Addr[3]	ARM Bus Input Addr[2]	ARM Bus Input Addr[1]	(Not Used)	Rst Value 0000000xb Read Value 00h

Bit 15-1   ARM Bus Interface Input (A2C) Data Low Address [15:1].

Bit 0   Not Used

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus C2A High Address *(Abc2aHiAddr)* **$01FF82A1**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus C2A High Address *(Abc2aHiAddr)* **$01FF82A0**	(Not Used)	ARM Bus Output Addr[22]	ARM Bus Output Addr[21]	ARM Bus Output Addr[20]	ARM Bus Output Addr[19]	ARM Bus Output Addr[18]	ARM Bus Output Addr[17]	ARM Bus Output Addr[16]	Rst Value x0000000b Read Value 00h

Bit 15-7   Not Used

Bit 6-0   ARM Bus Interface Output (C2A) Data High Address [22:16]

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus C2A Low Address *(Abc2aLoAddr)* **$01FF82A3**	ARM Bus Output Addr[15]	ARM Bus Output Addr[14]	ARM Bus Output Addr[13]	ARM Bus Output Addr[12]	ARM Bus Output Addr[11]	ARM Bus Output Addr[10]	ARM Bus Output Addr[9]	ARM Bus Output Addr[8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus C2A Low Address *(Abc2aLoAddr)* **$01FF82A2**	ARM Bus Output Addr[7]	ARM Bus Output Addr[6]	ARM Bus Output Addr[5]	ARM Bus Output Addr[4]	ARM Bus Output Addr[3]	ARM Bus Output Addr[2]	ARM Bus Output Addr[1]	(Not Used)	Rst Value 0000000xb Read Value 00h

Bit 15-1   ARM Bus Interface Output (C2A) Data Low Address [15:1]

Bit 0   Not Used

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus C2A Block Size (Abc2aBlkSiz) **$01FF829B**	ARM Bus Output Enable	ARM Bus Output Block Size [14]	ARM Bus Output Block Size [13]	ARM Bus Output Block Size [12]	ARM Bus Output Block Size [11]	ARM Bus Output Block Size [10]	ARM Bus Output Block Size [9]	ARM Bus Output Block Size [8]	Rst. Value 00h Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus C2A Block Size (Abc2aBlkSiz) **$01FF829A**	ARM Bus Output Block Size [7]	ARM Bus Output Block Size [6]	ARM Bus Output Block Size [5]	ARM Bus Output Block Size [4]	ARM Bus Output Block Size [3]	ARM Bus Output Block Size [2]	ARM Bus Output Block Size [1]	ARM Bus Output Block Size [0]	Rst. Value 00h Read Value 00h

Bit 15   ARM Bus Interface Output (C2A) Data DMA Enable (1 = enabled)

Bit 14-0   ARM Bus Interface Output Data Block Size

> *Note*: Writes to bits 14-0 are ignored when the channel is enabled. To write a new block size, the channel must first be disabled by writing a "0" to bit 15 of this register.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus A2C Throttle (Aba2cThrottle) **$01FF82A5**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	ARM Bus Input Throttle[10]	ARM Bus Input Throttle[9]	ARM Bus Input Throttle[8]	Rst. Value xxxxx000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus A2C Throttle (Aba2cThrottle) **$01FF82A4**	ARM Bus Input Throttle[7]	ARM Bus Input Throttle[6]	ARM Bus Input Throttle[5]	ARM Bus Input Throttle[4]	ARM Bus Input Throttle[3]	ARM Bus Input Throttle[2]	ARM Bus Input Throttle[1]	ARM Bus Input Throttle[0]	Rst Value 00h Read Value 00h

Bit 15-11   Not used

Bit 10-0   ARM Bus Interface Input (A2C) DMA Throttle value

A value of zero indicates no throttle. A value of 3FFh indicates 2047 Countach Bus Subsystem clocks (25 MHz) between successive DMA acknowledges.

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
ARM Bus C2A Throttle *(Abc2aThrottle)* **$01FF82A7**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	ARM Bus Output Throttle[10]	ARM Bus Output Throttle[9]	ARM Bus Output Throttle[8]	Rst. Value xxxxx000b Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
ARM Bus C2A Throttle *(Abc2aThrottle)* **$01FF82A6**	ARM Bus Output Throttle[7]	ARM Bus Output Throttle[6]	ARM Bus Output Throttle[5]	ARM Bus Output Throttle[4]	ARM Bus Output Throttle[3]	ARM Bus Output Throttle[2]	ARM Bus Output Throttle[1]	ARM Bus Output Throttle[0]	Rst Value 00h Read Value 00h

Bit 15-11   Not used

Bit 10-0   ARM Bus Interface Output (C2A) DMA Throttle value

A value of zero indicates no throttle. A value of 3FFh indicates 2047 Countach Bus Subsystem clocks (25 MHz) between successive DMA acknowledges.

24.5.3 Firmware Operation

CDMA Channel 0 Setup and Operation

1. Program the DMA Controller

    •   Select Mode
    •   For Video Mode set up Address Register 2
    •   Write Jump Value
    •   Write Address Register 1. This will also initialize the channel by setting up the Address Holder and Address Counter.

2. Operation

    •   At the end of each line of data the vsi_eol will pulse, causing the holder value to be added to the jump value and placed back into the Address Holder.
    •   For Video Mode, multiple vsi_eols will occur, then a vsi_eof. Upon the 1st vsi_eof, the Address Register2 will be loaded into the holder and counter.
    •   Additional vsi_eols will occur and then a 2nd vsi_eof. Upon the 2st vsi_eof the IRQ will be asserted.

24.6 VIDEO/SCANNER INTERFACE

24.6.1 Function Description

The Video/Scan Interface connects the Video/Scan Controller to the SDRAM via DMA cycles. This interface has a unidirectional data flow: Video/Scan Controller to SDRAM. It consists of two 64 halfword buffers that ping-pong, address counters for Video/Scan Controller input, and CDRAM output and byte to halfword assembly.

The Video/Scan Controller transmits a single line when attached to the scanner input or a single frame when attached to the video input. In either case, the Video/Scan Controller data flow is as follows:

1. The Video/Scan Controller initiates a transfer by a single high-going pulse on the start signal to VSI .
2. As each analog sample becomes available, it is placed on the VSC output bus and the data ready signal is pulsed high to VSI. In the case of video capture, it is expected that the data ready signal will remain high for long periods of time as video data is available at each rising edge of the data clock. In the case of scanner capture, it is expected that the data ready signal will be inactive for several data clocks between successive pixels.
3. As the Video/Scan Interface receives data from the Video/Scan Controller, it is assembled into halfwords and then stored into one of the two ping-pong buffers. When the buffer is filled, data storing is switched to the remaining buffer and a signal is passed to the opposite side of the interface indicating a buffer is available for transfer.
4. The Video/Scan Controller terminates the transfer by a single high-going pulse on the stop signal.
5. On the Countach Bus Subsystem side of the Video/Scan Interface, the data flow is as follows:
6. A pulse is received from the Video/Scan Controller side of the Video/Scan Interface indicating a buffer is available for transfer.
7. VSI asserts the DMA request signal to CBU. The sequential DMA signal is also asserted if the buffer contains more than one halfword of data.
8. Once the DMA acknowledge signal is detected, transfer begins, one data being transferred for each SDRAM ready pulse.

24.6.2 Register Description

Address:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default:
Video/Scanner Interface Mode *(VSIMode)* **$01FF8289**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Rst. Value xxh Read Value 00h
Address:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default:
Video/Scanner Interface Mode *(VSIMode)* **$01FF8288**	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	(Not Used)	Video/Scanner DMA Hog Mode[1]	Video/Scanner DMA Hog Mode[0]	Rst Value xxxxxx00b Read Value 00h

Bit 15-2   Not used

Bit 1-0   The Video/Scanner DMA Hog Mode field is used to throttle and/or disable other DMA masters during video capture. It has the following settings:

Hog Mode	Countach Subsystem Interface	Arm Bus Interface
0	Burst and single DMA allowed	Burst and single DMA allowed
1	No burst – single DMA only	No burst – single DMA only
2	No DMA allowed	No burst – single DMA only
3	No DMA allowed	No DMA allowed

24.6.3 Internal Memory

The Video/Scanner Interface contains two 64x16 buffers for DMA transfers from the Video/Scanner Controller and the Countach SDRAM. The buffers can be accessed by the ARM as halfwords only.

The two buffers are located at the following address:

Ping pong buffer 1: $01FFA400—$01FFA47F

Ping pong buffer 2: $01FFA480—$01FFA4FF

ARM access to the VSI buffers is provided for diagnostic purposes only and should not be used as part of normal operation. When the ARM makes an access to the buffers, the ARM is held off and clock domain synchronization occurs before the access is completed. As a result, a significant wait state penalty is incurred.

24.7 (S)DRAM Controller ((S)DRAMC)

24.7.1 Function Description

The (S)DRAM Controller is used to communicate between the Countach Bus Unit (CBU) and external FPDRAM or SDRAM memory. This memory will primarily be used for reading and writing large amounts of sequential data necessary for carrying out the programs running on the Countach Imaging DSP core. Whether DRAM or SDRAM is used is an option, but only one memory unit is accessible to the memory controller.

The (S)DRAMC state machine uses a 100Mhz or 87.5Mhz clock to control the SDRAM and DRAM. The CBU will communicate with (S)DRAMC at 25 Mhz or 21.875Mhz, so all data received from either RAM type must be synchronized to this speed before sent to the CBU. A third clock, an inverted version of the 100Mhz or 87.5Mhz clock, will be sent to the SDRAM memory. In this document, 100Mhz clock will be used for all descriptions. From the timing point of view, 87.5Mhz will be covered if it meets 100Mhz requirements.

The supported DRAM characteristics are listed in the following table:

Table 24-4: Supported FPDRAM Chip Characteristics

Addressing Size:	2 MB, 8 MB
Organization:	16 bits
Access Speed:	50 ns, 60 ns

Table 24-5: Supported SDRAM Chip Characteristics

Addressing Size:	2 MB, 8 MB
Organization:	8 bits or 16 bits
Access Speed:	10 ns

The maximum memory size that is supported for either memory type is 8MB. Six possible row/column pinouts types are supported. A more detailed description can be found in the address multiplexing table.

SDRAM will always be used in burst mode, with one access per burst. The CAS latency will be set to 3 cycles which is normal for SDRAM running at 100MHz. The two possible sizes and two possible organizations are programmable in the (S)DRAMC control register.

Fast page mode DRAMs (50ns and 60ns) are supported. The speed requirement is due to the bandwidth requirements of the Countach Imaging DSP. The DRAMS will be used in both fast page burst mode and single access mode. The two possible densities which are supported are programmable in the (S)DRAMC control register.

24.7.1.1 Memory Bank Structure

There will only be one bank of memory allowed. Accesses to those banks must be done as halfword (16 bit) accesses. SDRAM with 8-bit data bus will not be byte addressable as far as the system in concerned. Instead, each 16-bit access will write or read two consecutive bytes of data to/from the 8-bit SDRAM. For instance, a request to access to byte location 0x100 or 0x101 in 8-bit SDRAM will both access the same halfword (16-bit) location. Hardware does NOT support byte access to external memory connected to (S)DRAM.

24.7.1.2 Wait State Profile

50 ns FPDRAM read requests will have two 25Mhz wait states for the first read access and zero wait states for each successive read. So the FPDRAM read wait state profile is considered to be 2-0-0-0. For example, a burst of four reads will take six cycles to complete; three for the first read, and three total for the next three reads. For 60ns FPDRAM, the read wait state profile is 2-0-1-0-1-0, etc.

The write wait state profile for 50ns FPDRAM is 1-0-0-0. The write wait state profile for 60 ns FPDRAM is 1-0-1-0-, etc.

SDRAMs will have an identical wait state profile regardless of whether they use a 16 bit data bus or 8 bit data bus. Writes will have a profile of 1-0-0-0. Reads will have a wait state profile of 2-0-0-0. In either case when the burst or single access is finished, there will be a 1 cycle penalty in order to satisfy Trc.

24.7.1.3 Address Multiplexing

When reading chart, note that the address bits and data bit on the leftmost column correspond to the actual bits which are output of the (S)DRAMC block (also outputs of ASIC), whereas the address bits in the row/col columns correspond to the input addresses to the (S)DRAMC block (which are aligned to halfword boundary). In other words, the address requested by CBU will be a halfword address, not a byte address. As a result, the CBU and (S)DRAMC will communicate via a 22-bit address bus.

In the following charts, MA[x] refers to the memory address pin located on the MFC2000 ASIC (i.e.,. cdao[x]). MD[x] refers to the memory data pin located on the MFC2000 ASIC (i.e.,cddo[x]).

BA[x] refers to the bank pin on the SDRAM part. EA[x] refers to the external address pin located on the RAM part.

So the External address column of the chart shows the actual connection between the MFC2000 and the RAM. (i.e., looking at the first row of the chart, there will be a wire connecting MA[12] of the MFC2000 to the BA1 of the SDRAM).

A[x], within the ROW/COL columns of the chart, refers to the 22-bit address bus within the chip

Table 24-6. Untitled table

Address Multiplexing Register	SDRAM 64 Mb (16 bit data bus) (14 x 8)		SDRAM 64 Mb (8-bit data bus) (14 x 9)		SDRAM 16 Mb (16 bit data bus) (12 x 8)		SDRAM 16 Mb (8 bit data bus) (12 x 9)	
External Address	ROW	COL.	ROW	COL.	ROW	COL.	ROW	COL.
MA[12]/BA[1]	A[21]	A[21]	A[21]	A[21]	0	0	0	0
MA[11]/BA[0]	A[20]	A[20]	A[20]	A[20]	0	0	0	0
MA[10]/EA[11]	A[19]	0	A[19]	0	A[19]	A[19]	A[19]	A[19]
MA[9]/EA[10]	A[18]	0[1]	A[18]	0[1]	A[18]	0[1]	A[18]	0[1]
MD[0]/EA[9]	A[17]	NC[3]	A[17]	NC[3]	A[17]	NC[3]	A[17]	NC[3]
MA[8]/EA[8]	A[16]	0	A[16]	(0 or 1)[2]	A[16]	0	A[16]	(0 or 1)[2]
MA[7]/EA[7]	A[15]	A[7]	A[15]	A[7]	A[15]	A[7]	A[15]	A[7]
MA[6]/EA[6]	A[14]	A[6]	A[14]	A[6]	A[14]	A[6]	A[14]	A[6]
MA[5]/EA[5]	A[13]	A[5]	A[13]	A[5]	A[13]	A[5]	A[13]	A[5]
MA[4]/EA[4]	A[12]	A[4]	A[12]	A[4]	A[12]	A[4]	A[12]	A[4]
MA[3]/EA[3]	A[11]	A[3]	A[11]	A[3]	A[11]	A[3]	A[11]	A[3]
MA[2]/EA[2]	A[10]	A[2]	A[10]	A[2]	A[10]	A[2]	A[10]	A[2]
MA[1]/EA[1]	A[9]	A[1]	A[9]	A[1]	A[9]	A[1]	A[9]	A[1]
MA[0]/EA[0]	A[8]	A[0]	A[8]	A[0]	A[8]	A[0]	A[8]	A[0]

1. EA[10] is used on SDRAMs for auto-precharge during column access. Auto-precharge will not be utilized by CDRAM so EA[10] will always be set to 0 during column access.

2. EA[8] will be used as the byte control bit for 8-bit SDRAMs. Addresses sent to CDRAM from the CBU will always be halfword addressable. In order to communicate with 8-bit SDRAMs, CDRAM toggles EA[8] and obtains two sequential pieces of 8-bit data and sends it back to the system as one 16-bit data chunk.

3. Since MD[0] is used both as a data pin and an address pin, during the column portion of a SDRAM access, the SDRAM will not read from the EA[9] pin. So EA[9] is no-care [NC] during this time. Although at that time, MD[0] may very well be carrying valid data to the data bus.

Table 24-7. Untitled table

Address Multiplexing Register	DRAM 16 or 64 Mb (16 bit data bus) 12 x 10 or 10 x 10[1]		DRAM 16 Mb (16 bit data bus) 12 x 8	
Physical Address	**ROW**	**COL.**	**ROW**	**COL.**
MA[12]/EA[11]	A[21]	0	A[19]	0
MA[11]/EA[10]	A[20]	0	A[18]	0
MA[10]/EA[9]	A[19]	A[9]	A[17]	0
MA[8]/EA[8]	A[18]	A[8]	A[16]	0
MA[7]/EA[7]	A[17]	A[7]	A[15]	A[7]
MA[6]/EA[6]	A[16]	A[6]	A[14]	A[6]
MA[5]/EA[5]	A[15]	A[5]	A[13]	A[5]
MA[4]/EA[4]	A[14]	A[4]	A[12]	A[4]
MA[3]/EA[3]	A[13]	A[3]	A[11]	A[3]
MA[2]/EA[2]	A[12]	A[2]	A[10]	A[2]
MA[1]/EA[1]	A[11]	A[1]	A[9]	A[1]
MA[0]/EA[0]	A[10]	A[0]	A[8]	A[0]

[1.] In the 10x10 version, MA[11] and MA[12] will be left unconnected

Table 24-8. Untitled table

Memory Size/Type	Supported/Not Supported	Row/Column Configuration	
		Row	**Column**
512K x 16 x 2 bank SDRAM (16 Mb)	Supported	11 2 banks (1 bit)	8
1M x 8 x 2 bank SDRAM (16 Mb)	Supported	11 2 banks (1 bit)	9
1M x 16 x 4 bank SDRAM (64 Mb)	Supported	12 4 banks (2 bits)	8
2M x 8 x 4 bank SDRAM (64 Mb)	Supported	12 4 banks (2 bits)	9
2M x 16 x 2 bank SDRAM (64 Mb)	Supported	13 2 banks (1 bit)	8
4M x 8 x 2 bank SDRAM (64 Mb)	Supported	13 2 banks (1 bit)	9
1M x 16 DRAM (16 Mb)	Supported Supported	12 10	8 10
4M x 16 DRAM (64 Mb)	Supported Not Supported	12 13	10 9

As seen in the preceding tables, special care must be taken when connecting a FPDRAM or SDRAM to the MFC2000 for use with (S)DRAMC. In summary, when using FPDRAMs, A[8:0] of the DRAM should be connected to MA[8:0] of the (S)DRAMC. A[11:9] of the DRAM should be connected to MA[12:10] of the (S)DRAMC.

When using SDRAMs, for 64 Mb type, A[8:0] of the SDRAM should be connected to MA[8:0] of the (S)DRAMC. A[9] of the SDRAM should be connected to MD0 of the (S)DRAMC. A[13:10] of the SDRAM should be connected to MA[12:9] of the (S)DRAMC. Consequently, the MD0 pin out of the (S)DRAMC is time multiplexed with address information during the row address portion of a SDRAM access and contains data information during the column address portion of a SDRAM access. This was done to conserve pin usage of the MFC2000. For the same reason, any DQM pins of an SDRAM should be connected to ground. The CKE pin of the SDRAM should be connected high.

Also, if two bank 64Mb SDRAM is being used (the chart describes four bank), MA[11] should be connected to EA[12] since there will only be one bank pin (as compared to the four bank SDRAM in which there are two bank pins).

For board layout: In order to maintain low delay and minimized crosstalk, the SDRAM should be placed no farther than 0.5 inch from the MFC2000.

24.8 Register Description

Name/Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
DRAM Configuration **0x01FF8281**	NA	NA	NA	NA	NA	NA	NA	NA	Rst. Value 'h00 Read Value 'h00
Name/Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
DRAM Configuration **0x01FF8280**	NA	NA	NA	CDRAM Clock Enable	Configuration Code				Rst. Value 'h00 Read Value 'h00

This register contains the information necessary to decide whether DRAM or SDRAM is being used, the size of memory being used, and whether an 8- or 16-bit DRAM is being used. **This register should be written to ONCE and only once as the CDRAM may not work properly if this is written to a second or more times.** In addition, most RAMs require a minimum of 100us delay before use in order for the RAM to stabilize, but some others require more time. To properly use the RAM, this register should be written to ONLY after the appropriate delay for that particular RAM is met. Writing to this register will cause the CDRAM to perform the proper ammount of refreshes, the precharge, and mode set necessary for correct startup of DRAM or SDRAM.

Bit 15-5

Bit 4   **CDRAM Clock Enable**   **This bit enables or disables the CDRAM external clock. When using FPDRAM, the clock does not need to be enabled.**
0: Enable
1: Disable

Bit 3-0

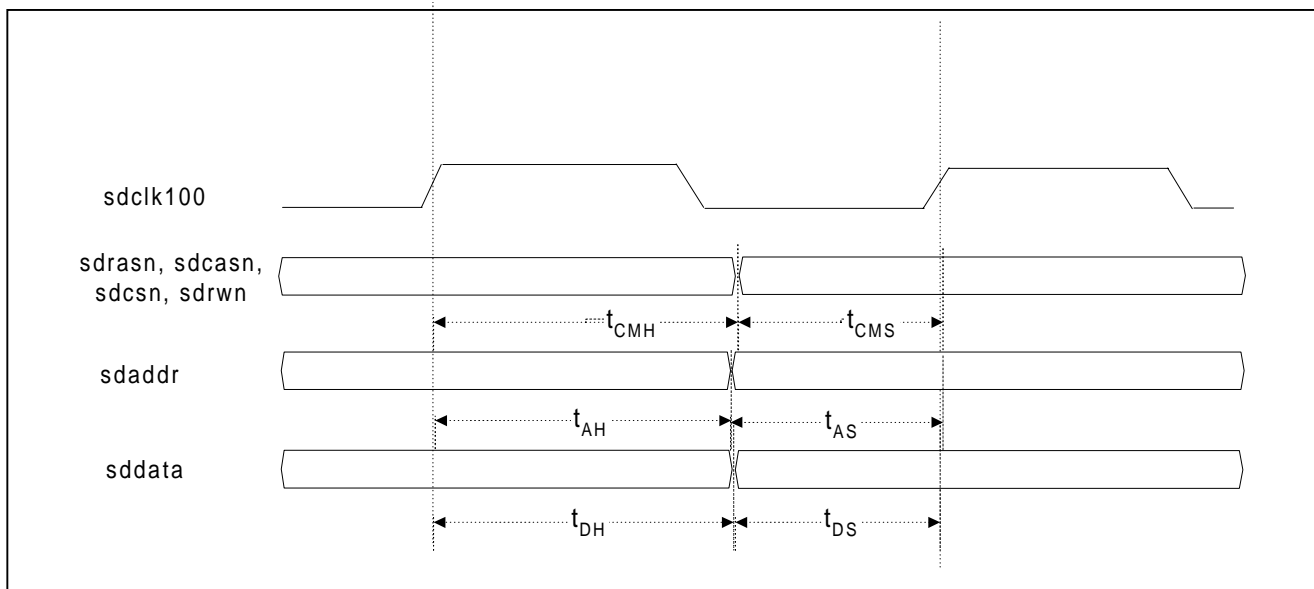| DRAM Type	Configuration Code
SDRAM 8-bit 16 Mb	0010
SDRAM 8-bit 64 Mb	0110
SDRAM 16-bit 16 Mb	0000
SDRAM 16-bit 64 Mb	0100
FPDRAM 60ns 16 Mb (10x10)	0111
FPDRAM 60ns 16 Mb (12x8)	0011
FPDRAM 60ns 64 Mb (12x10)	0111
FPDRAM 50ns 16 Mb (10x10)	1111
FPDRAM 50ns 16 Mb (12x8)	1011
FPDRAM 50ns 64 Mb (12x10)	1111

24.8.1 Timing

24.8.1.1 Detailed Timing Measurements

Figure 24-2. SDRAM Setup and Hold Timing

Table 24-9. SDRAM Setup and Hold Timing

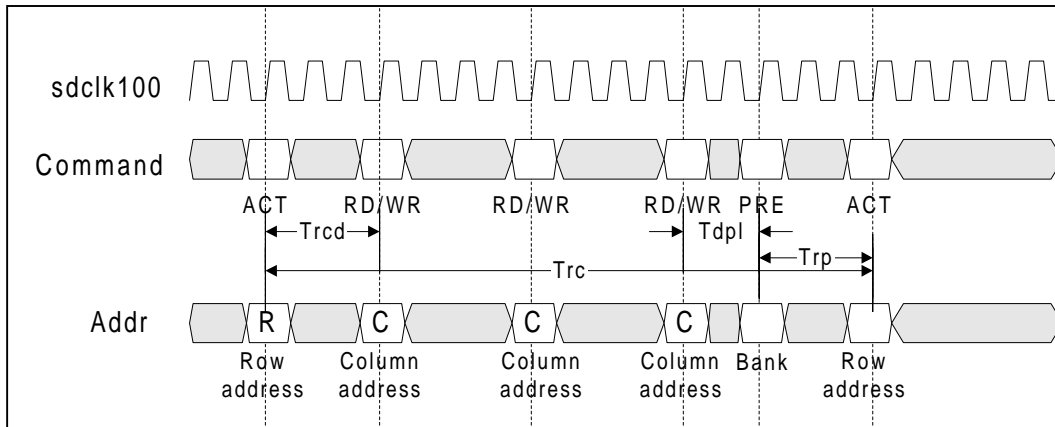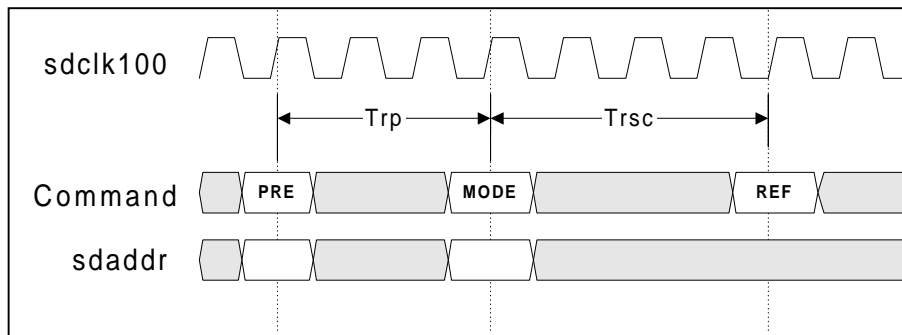| Parameter	Symbol	Min.	Max.	Units
Command setup time (includes rasn, casn, wen, csn)	t_{CMS}	4		ns
Command hold time	t_{CMH}	3		ns
Address setup time	t_{AS}	4		ns
Address hold time	t_{AH}	3		ns
Data setup time	t_{DS}	4		ns
Data hold time	t_{DH}	3		ns

Figure 24-3. SDRAM Read or Write Timing

Figure 24-4. SDRAM Mode Timing

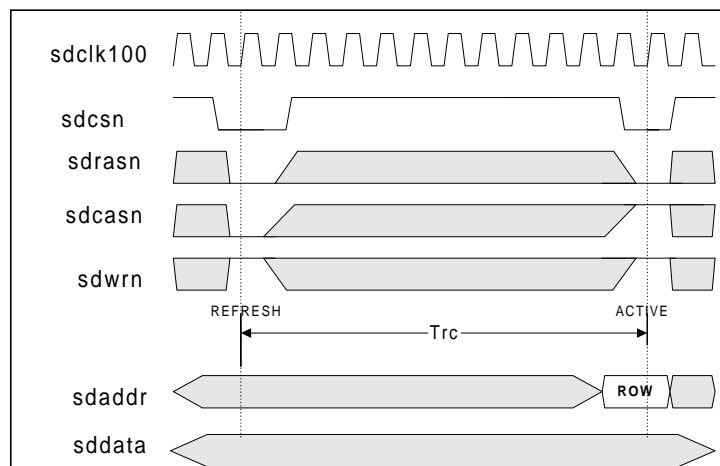*Note: In the above SDRAM timing waveforms "Command" is a combination of rasn, casn, wen, and csn.*

Figure 24-5. SDRAM Refresh Timing

Table 24-10. Timing Parameters for 16-bit SDRAM Read and Write

Parameter	Symbol	Min.	Max.	Units
Clock Period	T_{CLK}	10		ns
Ras precharge time	T_{RP}	3 (5 for single access)		T_{CLK}
Mode register set to Active delay	T_{RSC}	4		T_{CLK}
Ras cycle time	T_{RC}	12		T_{CLK}
Ras-to-cas delay	T_{RCD}	3		T_{CLK}
Data-in to Pre command period	T_{DPL}	2 (4 for single access)		T_{CLK}

Table 24-11. Timing Parameters for 8-bit SDRAM Read and Write

Parameter	Symbol	Min.	Max.	Units
Clock Period	T_{CLK}	10		ns
Ras precharge time	T_{RP}	5 (5 for single access)		T_{CLK}
Mode register set to Active delay	T_{RSC}	4		T_{CLK}
Ras cycle time	T_{RC}	12		T_{CLK}
Ras-to-cas delay	T_{RCD}	3		T_{CLK}
Data-in to Pre command period	T_{DPL}	3 (3 for single access)		T_{CLK}
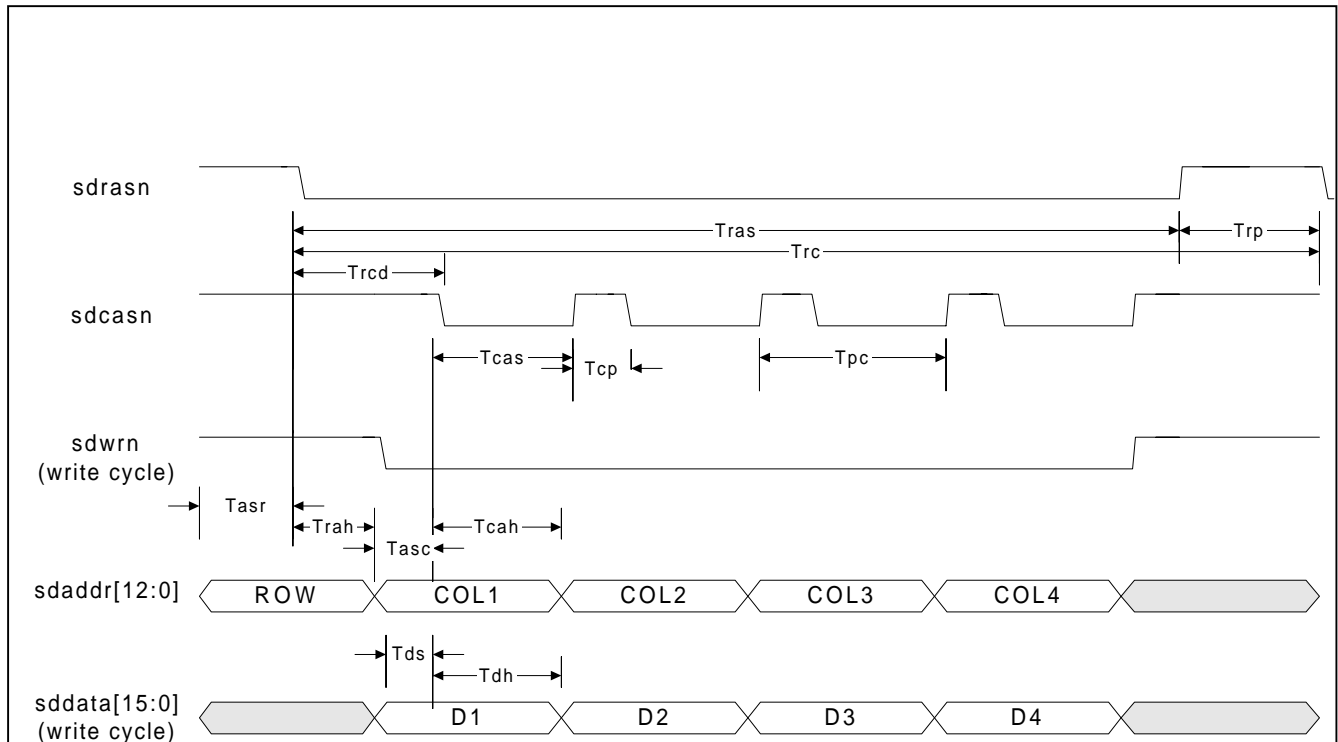
Figure 24-6. FPDRAM Timing (Read or Write)

Table 24-12. 60ns Timing

Parameter	Symbol	Min.	Max.	Units
Ras pulse width	T_{RAS}	70		ns
Cas pulse width	T_{CAS}	20(30 for read)		ns
Ras-to-cas delay	T_{RCD}	40		ns
Ras precharge	T_{RP}	50		ns
Cas precharge	T_{CP}	40		ns
Column address setup	T_{ASC}	10		ns
Column address hold	T_{CAH}	30		ns
Row address setup	T_{ASR}	10		ns
Row address hold	T_{RAH}	20		ns
Fast page mode cycle	T_{PC}	60		ns
Ras-to-ras delay	T_{RC}	120		ns
Data setup time	T_{DS}	20		ns
Data hold time	T_{DH}	30		ns

Table 24-13. 50ns Timing

Parameter	Symbol	Min.	Max.	Units
Ras pulse width	T_{RAS}	70		ns
Cas pulse width	T_{CAS}	20(30 for read)		ns
Ras-to-cas delay	T_{RCD}	30		ns
Ras precharge	T_{RP}	50		ns
Cas precharge	T_{CP}	20(10 for read)		ns
Column address setup	T_{ASC}	10		ns
Column address hold	T_{CAH}	20(30 for read)		ns
Row address setup	T_{ASR}	10		ns
Row address hold	T_{RAH}	20		ns
Fast page mode cycle	T_{PC}	40		ns
Ras-to-ras delay	T_{RC}	120		ns
Data setup time	T_{DS}	20		ns
Data hold time	T_{DH}	30		ns

Note: The most notable difference between the 50ns and 60ns timing is the Tpc parameter. During bursts, 50ns is able to perform much faster.
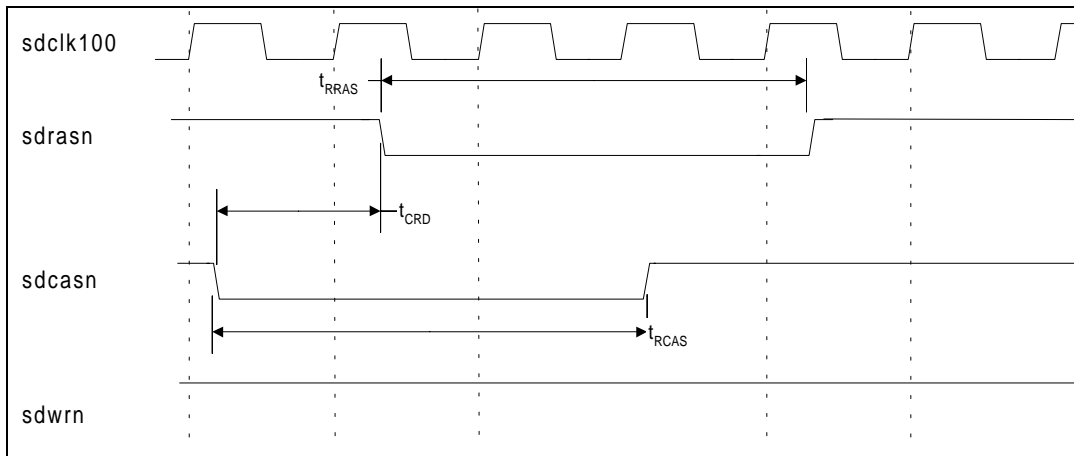
Figure 24-7. FPDRAM Timing (Refresh)

Table 24-14. FPDRAM Timing (Refresh)

Parameter	Symbol	Min.	Max.	Units
OE delay	t_{OED}		6	ns
CASN pulse width	t_{RCAS}	28		ns
RASN pulse width	t_{RRAS}	60		ns
CASN to RASN delay	t_{CRD}	9		ns

24.8.2 Firmware Operation

There is only one operation in which firmware need be concerned with the CDRAM block. That operation is the powerup operation. During startup, DRAM and SDRAM both have a specific sequence which needs to be performed on them in order for them to function properly. In order to assure that this sequence is properly performed the following must be done:

1. Wait a minumum of 200us. Some memories require as little as 100us, while a few others require as great as 500us. Waiting 200us will assure that MOST memories will work correctly.
2. Write to the CDRAM configuration register. This register should be written to ONCE and ONLY ONCE. Unless this register is written to, CDRAM will not know what type of RAM is being used.
3. That's it. After about a 1 us delay the startup procedure should be finished and the memory ready to be accessed. An access request can be made after the startup procedure is done and before it is finished, but the access will not occur until the complete startup procedure is finished.

This page is intentionally blank

25. Configuration

25.1 Hardware Version

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Hardware Version *(HWVer)* **$01FF8045 (R)**	(Not Used)								Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Hardware Version *(HWVer)* **$01FF8044 (R)**	0:= First release 1-FF:= other releases								Rst. Value 00h Read Value 00h

This register is used to content the hardware version number. The value in this register is 0 for first release of a product. Whenever the hardware design is modified, this register value is incremented. This register is not effected by any reset.

00h—MFC2000_x1 (the PQFP package)

25.2 Product Code

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Default
Product Code *(ProductCode)* **$01FF8047 (R)**	(Not Used)								Rst. Value xxh Read Value 00h
Address	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default**
Product Code *(ProductCode)* **$01FF8046 (R)**	Product Code: BFh: 11626-11, with Data Modem Function, with Voice Codec/Speaker Phone Function, and with SmartDAA support BDh: 11626-12, with Data Modem Function, without Voice Codec/Speaker Phone Function, and with SmartDAA support BBh: 11626-13, without Data Modem Function, with Voice Codec/Speaker Phone Function, and with SmartDAA support B9h: 11626-14, without Data Modem Function, without Voice Codec/Speaker Phone Function, and with SmartDAA support B8h: 11626-15, without Data Modem Function, without Voice Codec/Speaker Phone Function, and without SmartDAA support								Rst. Value (must match with the product code)

This register is not affected by any reset.

This page is intentionally blank

INSIDE BACK COVER NOTES

Further Information
literature@conexant.com
(800) 854-8099 (North America)
(949) 483-6996 (International)
Printed in USA

World Headquarters
Conexant Systems, Inc.
4311 Jamboree Road
Newport Beach, CA
92660-3007
Phone: (949) 483-4600
Fax 1: (949) 483-4078
Fax 2: (949) 483-4391

Americas

**U.S. Northwest/
Pacific Northwest – Santa Clara**
Phone: (408) 249-9696
Fax: (408) 249-7113

U.S. Southwest – Los Angeles
Phone: (805) 376-0559
Fax: (805) 376-8180

U.S. Southwest – Orange County
Phone: (949) 483-9119
Fax: (949) 483-9090

U.S. Southwest – San Diego
Phone: (858) 713-3374
Fax: (858) 713-4001

U.S. North Central – Illinois
Phone: (630) 773-3454
Fax: (630) 773-3907

U.S. South Central – Texas
Phone: (972) 733-0723
Fax: (972) 407-0639

U.S. Northeast – Massachusetts
Phone: (978) 367-3200
Fax: (978) 256-6868

U.S. Southeast – North Carolina
Phone: (919) 858-9110
Fax: (919) 858-8669

U.S. Southeast – Florida/
South America
Phone: (727) 799-8406
Fax: (727) 799-8306

U.S. Mid-Atlantic – Pennsylvania
Phone: (215) 244-6784
Fax: (215) 244-9292

Canada – Ontario
Phone: (613) 271-2358
Fax: (613) 271-2359

Europe

Europe Central – Germany
Phone: +49 89 829-1320
Fax: +49 89 834-2734

Europe North – England
Phone: +44 1344 486444
Fax: +44 1344 486555

Europe – Israel/Greece
Phone: +972 9 9524000
Fax: +972 9 9573732

Europe South – France
Phone: +33 1 41 44 36 51
Fax: +33 1 41 44 36 90

Europe Mediterranean – Italy
Phone: +39 02 93179911
Fax: +39 02 93179913

Europe – Sweden
Phone: +46 (0) 8 5091 4319
Fax: +46 (0) 8 590 041 10

Europe – Finland
Phone: +358 (0) 9 85 666 435
Fax: +358 (0) 9 85 666 220

Asia – Pacific

Taiwan
Phone: (886-2) 2-720-0282
Fax: (886-2) 2-757-6760

Australia
Phone: (61-2) 9869 4088
Fax: (61-2) 9869 4077

China – Central
Phone: 86-21-6361-2515
Fax: 86-21-6361-2516

China – South
Phone: (852) 2 827-0181
Fax: (852) 2 827-6488

China – South (Satellite)
Phone: (86) 755-518-2495

China – North
Phone: (86-10) 8529-9777
Fax: (86-10) 8529-9778

India
Phone: (91-11) 692-4789
Fax: (91-11) 692-4712

Korea
Phone: (82-2) 565-2880
Fax: (82-2) 565-1440

Korea (Satellite)
Phone: (82-53) 745-2880
Fax: (82-53) 745-1440

Singapore
Phone: (65) 737 7355
Fax: (65) 737 9077

Japan
Phone: (81-3) 5371 1520
Fax: (81-3) 5371 1501

C O N E X A N T ™
What's next in communications technologies

www.conexant.com