GMS30C7201

Data Sheet

Issued: December 1998
Copyright Advanced RISC Machines Ltd (ARM) 1998
Copyright Hynix Semiconductor Inc.1999
All rights reserved

Proprietary Notice

Hynix logo are trademarks of Hynix Semiconductor Inc.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by Hynix in good faith. However, all warranties implied or expressed, including but not limited to implied warranties or merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. Hynix Inc shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

Document Status

The document's status is displayed in a banner at the bottom of each page. This describes the document's confidentiality and its information status.

Information status is one of:

Advance	Information on a potential product
Preliminary	Current information on a product under development
Final	Complete information on a developed product

Change Log

Issue	Date	By	Change
A-01	August 1997	PAW	First draft
A-02	December 1997	JM, PG	Second draft
A	January 1998	JM, PG	First Release
B-01	December 1998	PS,KC	Add timing details
C-01	June 1999	NAMIL	Second Release

z

Preface

Introduction

The GMS30C7201 is a highly-integrated microprocessor for personal digital assistants, and other applications described below. The device incorporates an ARM720T CPU, Piccolo DSP, and system interface logic to interface with various types of devices. GMS30C7201 is a highly-modular design based on the AMBA bus architecture between CPU and internal modules.

The on-chip peripherals include keyboard controller, VGA and LCD controller complete with DMA support for external SDRAM memory. The GMS30C7201 also supports voice recording, sound playback and a touch panel interface. Piccolo enables softmodems to be used, and an on chip CODEC interface enables a low-cost modem solution.

The power management features result in very low power consumption.

The GMS30C7201 provides an excellent solution for personal digital assistants(PDAs)

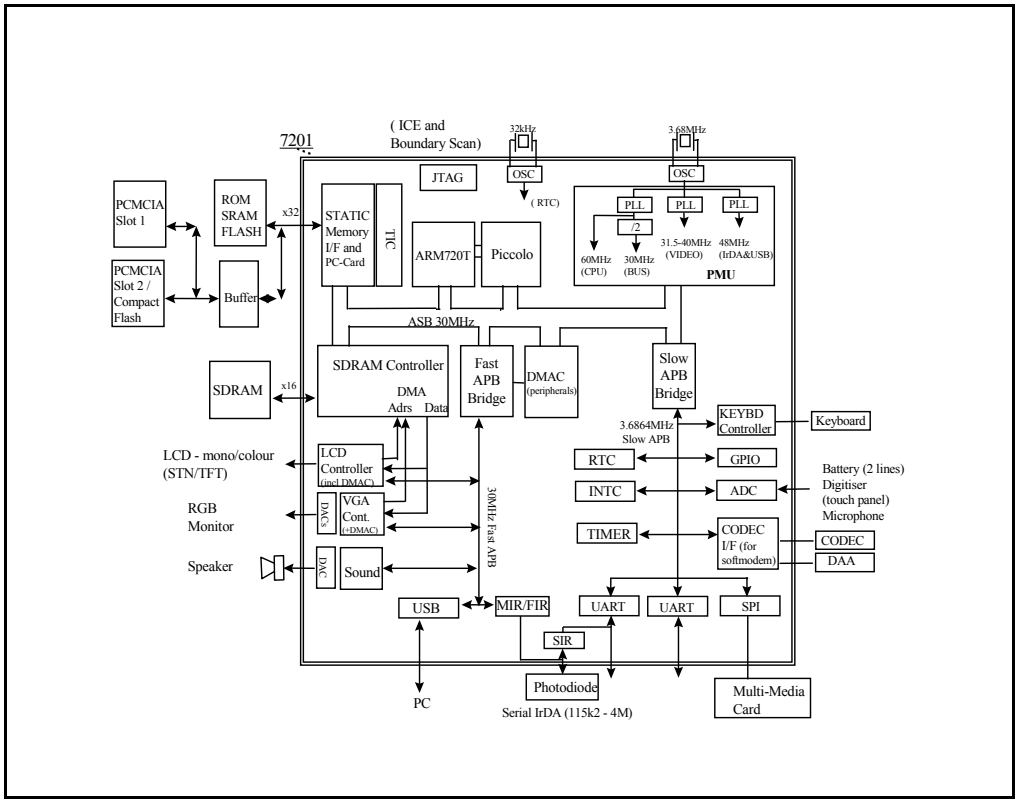
,and data terminal running the Microsoft Windows CE operating system. Other applications include smart phones, Internet appliances, car navigation, particularly where Windows CE support is required.

Features

- 32-bit ARM7TDMI RISC static CMOS CPU core
- 8Kbytes combined instruction/data cache
- Memory management unit for WindowsCE
- Piccolo DSP(supports softmodem)
- 512-byte instruction cache for Piccolo DSP
- Supports Little Endian operating system
- On-chip peripherals with individual power-down:
  - Multi-channel DMA
  - Timer
  - Interrupt Controller
  - Memory controller for ROM, Flash, SRAM, SDRAM
  - PCMCIA II and Compact Flash Controller
  - Power management unit
  - LCD Controller for mono/color STN and TFT LCD
  - VGA Controller with on-chip DACs (for direct drive of monitors)
  - Real-time clock (32.768kHz oscillator)
  - Infrared communications (IrDA support for 4Mbps and lower rates)
  - 2 UARTs (16C550 compatible)
  - AFE (Analog Front End or CODEC) interface
  - Keyboard control interface
  - GPIO
  - Synchronous Serial Interface module for MMC Card
  - USB (target)
  - ADC and interface module (touch panel)
  - DAC and interface module (sound output)
  - PLL
- JTAG debug interface and boundary scan
- 0.35mm process
- 3.3V supply voltage
- 360-pin BGA package

- 60MHz operation frequency
- Low power consumption

GMS30C7201 System Overview



Contents

1 Introduction

Introduction

1.1 Introduction

The GMS30C7201 is a high-performance, low-power, single-chip computer optimized for WinCE applications. It incorporates the ARM720T WinCE-enabled core, which also incorporates ARM's novel Thumb code compression mechanism. The GMS30C7201 also incorporates ARM's unique Piccolo DSP coprocessor, which gives the GMS30C7201 enough DSP horse-power to perform software modem functions simultaneously with WinCE operation.

1.1.1 Processor

The ARM720T core incorporates an 8K unified write-through cache, and an 8 data entry, 4 address entry write buffer. It also incorporates an MMU with a 64 entry TLB, and WinCE enhancements. The Piccolo SP7 core incorporates a 512-byte, instruction-only cache. Piccolo data is supplied by the ARM720T core, via the coprocessor interface, and hence may be cached in the ARM720T's 8K unified cache.

1.1.2 Piccolo

Piccolo is an ARM coprocessor that boosts the performance of the standard ARM720T CPU to state-of-the-art DSP levels. It integrates:

- DSP-oriented datapath
- associated DSP instruction set

in addition to the standard ARM 32-bit RISC/16-bit Thumb system.

The design/implementation of the Piccolo coprocessor allows data re-use - both the ARM720T and the coprocessor share the same single system bus. The implementation is therefore cost-effective and power-efficient. Other advantages of this approach include integrated hardware and software development. The ARM software development toolkit and emulator support both the CPU and the coprocessor. The GMS30C7201 will run a V34bis (33k6bps) softmodem.

1.1.3 Video

The GMS30C7201 has direct support for mono and color passive LCD displays, as well as color TFT LCD displays, with resolution programmable up to 640x480 VGA resolution. In addition, a separate independent VGA port allows simultaneous display on a VGA resolution monitor, of either the same image as the LCD, or alternatively an entirely different image. The GMS30C7201 has on-chip video DACs, allowing the chip to drive a monitor with the minimum of external circuitry.

1.1.4 Memory and PC-Cards

GMS30C7201 incorporates two separate memory interfaces. A high speed 16-bit wide interface connects directly to one to four 16, 64 or 128MBit SDRAM devices, supporting DRAM memory sizes in the range 2 to 64MB. In addition, a separate lower speed 32-bit data path interfaces to ROM or Flash devices. Burst mode ROMs are supported, for increased performance, allowing operating system code to be executed directly from ROM. Since the ROM and SDRAM interfaces are separate, the ARM processor core can access O/S code in ROM simultaneously with video DMA access to the SDRAM, thus increasing total effective memory bandwidth, and hence overall performance. In addition, running code directly from ROM reduces total system cost, since ROM is significantly cheaper on a $/bit basis than DRAM.

The ROM/Flash interface also allows control of one or two PC-Card interfaces, although in this case, external buffers and level translators are needed to interface to the card. The GMS30C7201 generates all signals to control these buffers directly. Since CompactFlash is a subset of the PC-Card standard, one of the PC-Card slots may be used as a CompactFlash interface. The PC-Card interface supports the 16-bit slave-only interface, with no DMA support. It does not support the CardBus 32-bit bus master option in the standard.

1.1.5 Peripherals and communications

Communications are well-catered for, by two UARTs, an IrDA interface (supporting slow, medium and fast protocols), a serial interface to a modem CODEC chip, for use by the soft-modem, and an on-chip keyboard controller, which directly scans the key matrix. One of the two UARTs is used to implement the IrDA protocol. If IrDA is not being used, then this UART is available for general use. A synchronous serial interface allows connection to a variety of devices, such as an RF modem or a Multimedia Card. A slave USB port supports connection of an GMS30C7201-based PDA as a peripheral to a PC or other USB host controller. A group of general purpose I/O pins can be utilized as required in a PDA design. An on-chip DAC supports audio output, and an on-chip ADC supports microphone input, the digitizer tablet and battery status monitoring functions. Internally, three general-purpose timers and a real-time clock provide timer functionality to be used as required by the O/S, and a two-channel general purpose DMA controller can be allocated to the communications peripherals, as required.

1.1.6 Power management

The GMS30C7201 incorporates advanced power management functions, allowing the whole device to be put into a standby mode, when only the real time clock runs. The SDRAM is put into low-power self refresh mode to preserve it's contents. The GMS30C7201 may be forced out of this state by either a real-time clock wake-up interrupt, a user wake-up event (which would generally be a user pressing the "on" key) or by the UART ring-indicate input. The power management unit (PMU) controls the safe exit from standby mode to operational mode, ensuring that SDRAM contents are preserved. In addition, halt and slow modes allow the processor to be halted, or run more slowly than usual, to reduce power consumption. The processor can be quickly brought out of the halted state by a peripheral interrupt. The advanced power management unit controls all this functionality. In addition, individual devices and peripherals may be powered down when they are not in use. For example, the VGA controller can be disabled when an external monitor is not in use (which saves not only the power of the digital controller, but also of the analog DACs), or the Piccolo DSP coprocessor can be powered-down when DSP support (in soft-modem code, for example) is not required.

1.1.7 Test and debug

The GMS30C7201 incorporates the ARM standard test interface controller (TIC) allowing 32-bit parallel test vectors to be passed onto the internal bus. This allows access to the ARM720T macro-cell core, and also to memory mapped devices and peripherals within the GMS30C7201. In addition, the ARM720T and Piccolo include support for the ARM debug architecture (Embedded ICE), which makes use of a JTAG boundary scan port to support debug of code on the embedded processor and DSP cores. The same boundary scan port is also used to support a normal pad-ring boundary scan for board level test applications.

2 Signal Description

Signal Description

2.1 Signal Description for GMS30C7201

Key to signal types:

O	Output
I	Input
OA	Analog output
IA	Analog input
IO	Input/output
IOA	Analog input/output
IS	Input with Schmitt level input threshold
P	Power input
u	Suffix to indicate integral pullup
d	Suffix to indicate integral pulldown

Signal name	Type	Description
LD[11:0]	O	LCD data bus. Allow 4:4:4 TFT, color (using [7:0]) or mono, using [3:0] or [7:0]
LCP	O	LCD clock pulse
LLP	O	LDC line pulse (HSync for TFT)
LFP	O	LCD frame pulse (VSync for TFT)
LAC	O	LCD AC bias (clock enable for TFT)
LCDEN	O	Display enable signal for LCD. Enables high voltage to LCD
LBLEN	O	LCD backlight enable
VGAROUTP	OA	VGA red output. Drives 75 ohm terminated load directly
VGAROUTM	OA	VGA red output. Drives 75 ohm terminated load directly
VGAGOUTP	OA	VGA green output. Drives 75 ohm terminated load directly
VGAGOUTM	OA	VGA green output. Drives 75 ohm terminated load directly
VGABOUTP	OA	VGA blue output. Drives 75 ohm terminated load directly
VGABOUTM	OA	VGA blue output. Drives 75 ohm terminated load directly
VGAHS	O	VGA HSync output
VGAVS	O	VGA VSync output
VGAREF	IA	VGA DAC reference current input
VGAAVDD[1:0]	P	VGA DAC analog Vdd supply
VBIAS	OA	VGA DAC analog bias
VT[1:0]	OA	VGA voltage reference for comparator
VGAAVSS	P	VGA DAC analog Vss supply

Table 2-1: Signal description table

Signal name	Type	Description
RA[25:0]	O	ROM and PC-Card address bus
RD[31:0]	IO	ROM and PC-Card data bus, plus test bus
nRCS[2:0]	O	ROM chip select outputs.
nRCS[5:3], or **PORTD[3:1]**	IO	ROM chip select outputs, or general IO pins controlled by the PMPS register in GPIO subsystem.
nROE	O	ROM and PC-Card output enable signal
nRWE[3:0]	O	ROM and PC-Card write enable signals. One per byte.
EXPRDY	I	Wait from external I/O
EXBCLK, or **PORTD[0]**	IO	Bus clock output, or general IO pin controlled by the PMPS register in GPIO subsystem.
BOOTBIT[1:0]	I	16/32 bit ROM selection
nPCBCE[1:0]	O	PC-Card Card enable signals
nPCACE[1:0]	O	PC-Card Card enable signals
nPCBIORD	O	PC-Card I/O read enable
nPCAIORD	O	PC-Card I/O read enable
nPCBIOWR	O	PC-Card I/O write enable
nPCAIOWR	O	PC-Card I/O write enable
PCARESET	O	PC-Card A reset signal
PCBRESET	O	PC-Card B reset signal
nPCAWAIT	I	PC-Card A wait signal
nPCBWAIT	I	PC-Card B wait signal
nPCREG	O	PC-Card attribute memory select
PCAREADY	I	PC-Card A ready input, or interrupt request in I/O mode
PCBREADY	I	PC-Card B ready input, or interrupt request in I/O mode
PCABVD[1:0]	I	PC-Card A battery status, or nSPKR/nSTSCHG in I/O mode
PCBBVD[1:0]	I	PC-Card B battery status, or nSPKR/nSTSCHG in I/O mode
nPCACD[1:0]	Iu	PC-Card A card detect signals
nPCBCD[1:0]	Iu	PC-Card B card detect signals
PCCADRV	O	PC-Card A is selected
PCCBDRV	O	PC-Card B is selected
PCBIPORTE	O	PC-Card data buffer external bidirectional buffer control
PCAWP	I	PC-Card A write protect/IOIS16

Table 2-1: Signal description table (continued)

Signal name	Type	Description
PCBWP	I	PC-Card B write protect/IOIS16
PCAVS[1:0]	Iu	PC-Card A voltage sense signals
PCBVS[1:0]	Iu	PC-Card B voltage sense signals
PCAVPPEN[1:0]	O	PC-Card A Vpp control signals
PCBVPPEN[1:0]	O	PC-Card B Vpp control signals
nPCBOE	O	PC-Card output enable detect signals
nPCAOE	O	PC-Card output enable detect signals
nPCBWE	O	PC-Card write enable for memory card
nPCAWE	O	PC-Card write enable for memory card
PCAVCCEN[1:0]	O	PC-Card A Vcc control signals
PCBVCCEN[1:0]	O	PC-Card B Vcc control signals
SCLK	O	SDRAM clock output
SCKE[3:0]	O	SDRAM clock enable output
nSRAS	O	SDRAM RAS output
nSCAS	O	SDRAM CAS output
nSWE	O	SDRAM write enable output
nSCS[3:0]	O	SDRAM chip select outputs
SDQML	O	SDRAM lower data byte enable
SDQMU	O	SDRAM upper data byte enable
SD[15:0]	IO	SDRAM data bus
SA[13:0]	O	SDRAM address bus
nUDCD[1:0]	Iu	UART data carrier detect inputs
nUDSR[1:0]	Iu	UART data set ready inputs
nUCTS[1:0]	Iu	UART clear to send inputs
USIN[1:0]	Iu	UART serial data inputs
USOUT[1:0]	O	UART serial data outputs
nUDTR[1:0]	O	UART data terminal ready
nURTS[1:0]	O	UART request to send
SSDIN	I	Synchronous serial data input
SSOUT	O	Synchronous serial data output
SSCLK	O	Synchronous serial clock output

Table 2-1: Signal description table (continued)

Signal name	Type	Description
nSSCS	O	Synchronous serial chip select
IRDIN	I	IrDA infra-red data input
IRDOUT	O	IrDA infra-red data output
GPIO: PORTA[7:0] PORTB[7:6] PORTC[7:0] PORTD[7:4]	 IO IO IO IO	 GPIO interface. 8 general-purpose I/O lines. GPIO interface. 2 general-purpose I/O lines. GPIO interface. 8 general-purpose I/O lines. GPIO interface. 4 general-purpose I/O lines.
MRING	I	Ring indicator (wake-up signal to PMU)
MRLY	O	Modem relay control
MCLK	I	Modem clock input
MDFR	I	Modem data frame input
MDIN	I	Modem data input
MDOUT	O	Modem data output
nMCON	O	Modem control data select (HIGH for data, LOW for control)
UVPO, or **PORTB[0]**	IO	USB output differential drive (+), or general IO pin controlled by the PMPS register in GPIO subsystem.
UVMO, or **PORTB[1]**	IO	USB output differential drive (-), or general IO pin controlled by the PMPS register in GPIO subsystem.
nUSBOE, or **PORTB[2]**	IO	USB output enable, or general IO pin controlled by the PMPS register in GPIO subsystem.
URCVIN, or **PORTB[3]**	IO	USB receive data, or general IO pin controlled by the PMPS register in GPIO subsystem.
UVP, or **PORTB[4]**	IO	USB gated input differential drive (+), or general IO pin controlled by the PMPS register in GPIO subsystem.
UVM, or **PORTB[5]**	IO	USB gated input differential drive (-), or general IO pin controlled by the PMPS register in GPIO subsystem.
USUSPEND	O	USB low power state
ATSXP	O	Touch screen switch X output
ATSXM	O	Touch screen switch X output
ATSYP	O	Touch screen switch Y output
ATSYM	O	Touch screen switch Y output
ADIN[4:0]	IA	ADC inputs for MIC and battery
AVDDDAC	P	DAC Analog Vdd and reference
AVSSDAC	P	DAC Analog Vss

Table 2-1: Signal description table (continued)

Signal Description

Signal name	Type	Description
AVREFADC	IA	ADC reference voltage
AVDDADC	P	ADC Analog Vdd
AVSSADC	P	ADC Analog Vss
ADAC[1:0]	OA	Sound DAC outputs
KSCANO[10:0]	O	Keyboard scan outputs
KSCANI[7:0]	Iu	Keyboard scan inputs
nPOR	ISu	Power on reset input. Schmitt level input, with pullup.
nPMWAKEUP	ISu	Wake-up “on-key” input. Low causes PMU to exit standby state.
nRESET	IOu	Reset input (also driven out in POR, until the PLL is locked)
PMADAPOK	I	Adapter power OK.
PMBATOK	I	Main battery OK.
RTCOSCIN	IA	RTC oscillator input
RTCOSCOUT	OA	RTC oscillator output
OSCIN	IA	Main oscillator input
OSCOUT	OA	Main oscillator output
nPLLENABLE	Id	Low to enable PLL. High to bypass PLL with clock from OSCIN.
PLLVDD[1:0]	P	PLL analog power supply
PLLVSS[1:0]	P	PLL analog ground input
PLLFILT[2:0]	IOA	External PLL loop filter input/output pins (1 per PLL)
TCK	Iu	JTAG boundary scan and debug test clock
nTRST	Id	JTAG boundary scan and debug test reset
TMS	Iu	JTAG boundary scan and debug test mode select
TDI	Iu	JTAG boundary scan and debug test data input
TDO	O	JTAG boundary scan and debug test data output
nTEST	Iu	Test mode select
VDDCore[3:0]	P	Core Vdd supply
VSSCore[3:0]	P	Core Vss supply
VDD[25:0]	P	Pad Vdd supply
VSS[25:0]	P	Pad Vss supply

Table 2-1: Signal description table (continued)

2.2 Package Details

This diagram,***Figure 2-1: Side View of the PBGA Package***, gives a side view of the PBGA package.

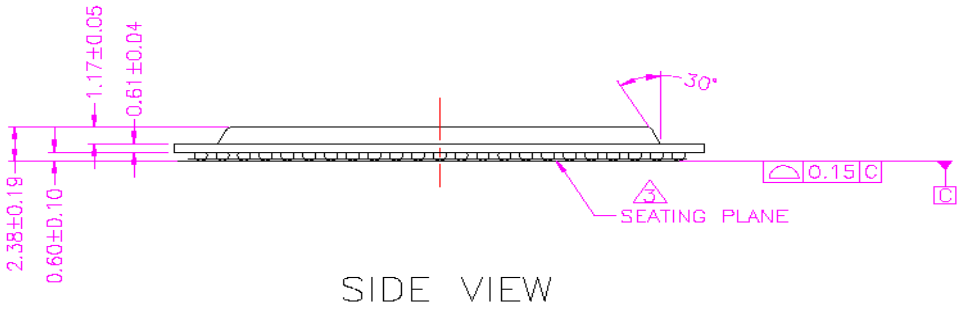


Figure 2-1: Side View of the PBGA Package

This diagram, ***Figure 2-2: Dimensions of the PBGA Package***, gives the dimensions of the package in millimeters.

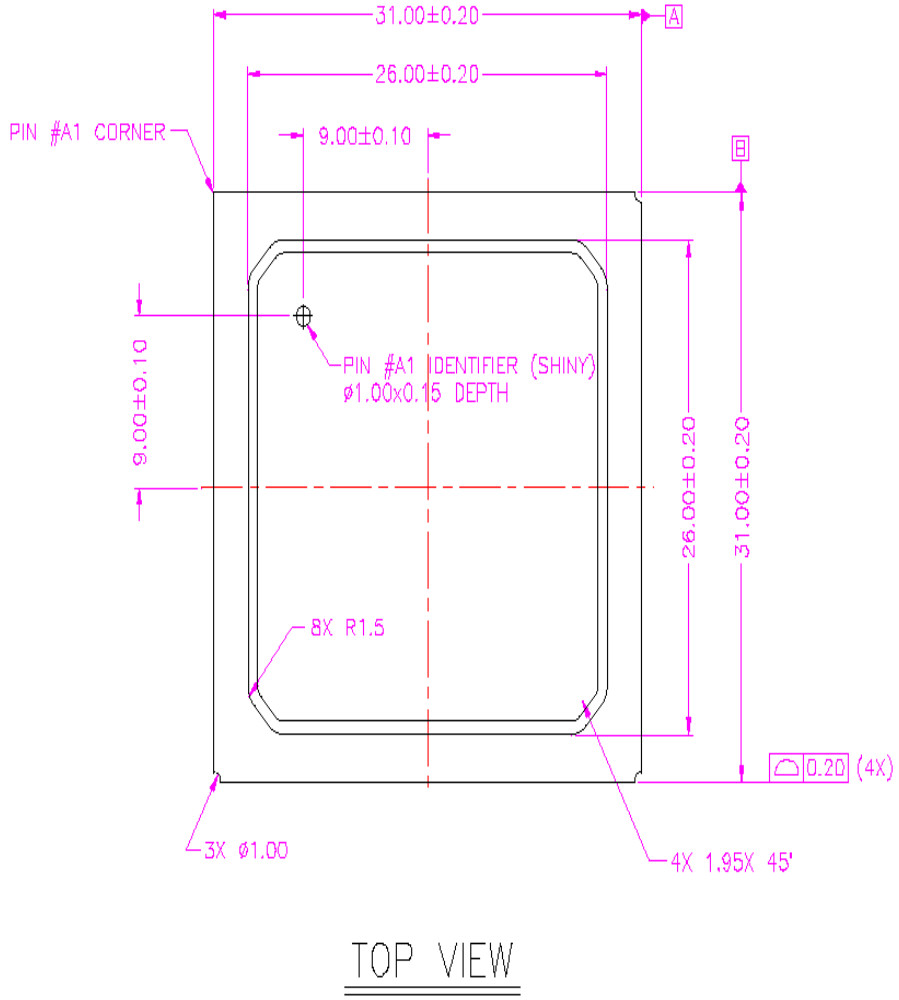


Figure 2-2: Dimensions of the PBGA Package

2.3 Pin List

Use ***Figure 2-3: Pin Location and Signal Cross-References (Bottom View)*** and ***Table 2-2: Pin Reference Numbers and Signal Names*** on page 2-8 to match pins and signals. Pin A1 (GND) is in the top right-hand corner of the diagram marked “PIN 1 CORNER”. This is a bottom view of the package.

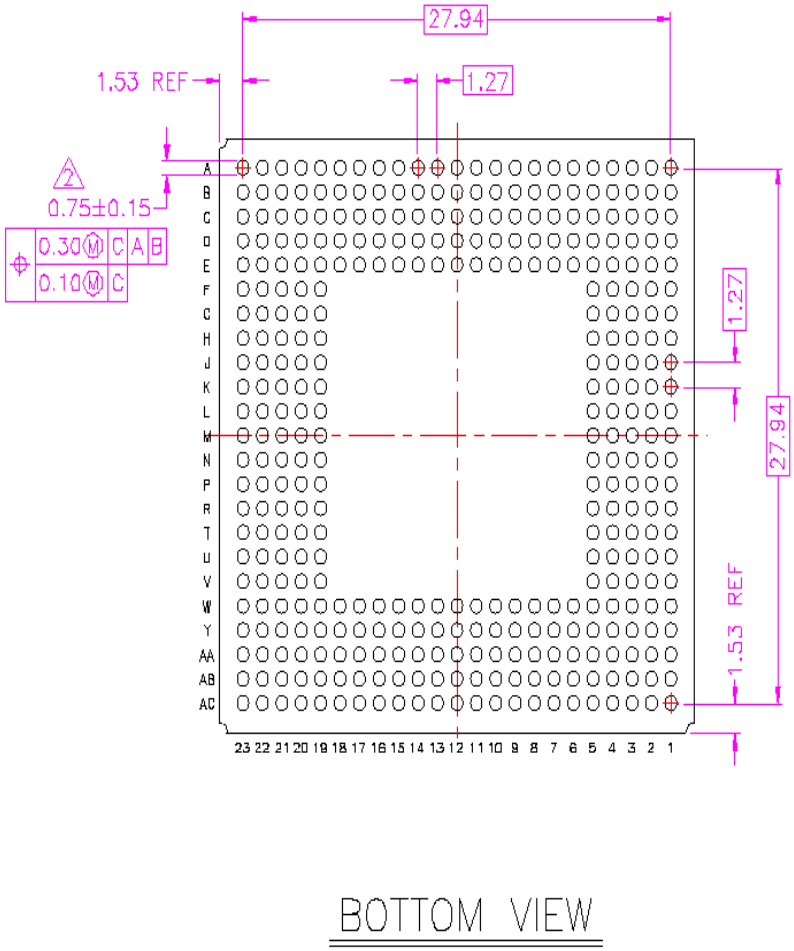


Figure 2-3: Pin Location and Signal Cross-References (Bottom View)

Pin	Signal	Pin	Signal
A1	GND	**A8**	nSCS[3]
A2	nTEST	**A9**	SCKE[3]
A3	SA[3]	**A10**	SDQML
A4	SA[4]	**A11**	SD[8]
A5	SA[5]	**A12**	SD[9]
A6	SA[0]	**A13**	SD[10]
A7	SA[12]	**A14**	SD[12]

Table 2-2: Pin Reference Numbers and Signal Names

Pin	Signal	Pin	Signal
A15	SD[14]	**B23**	USUSPEND
A16	PORTA[0]	**C1**	TDI
A17	PORTA[1]	**C2**	OSCIN
A18	PORTA[2]	**C3**	GND
A19	PORTA[3]	**C4**	nPLLENABLE
A20	UVPO	**C5**	SA[6]
A21	UVMO	**C6**	SA[10]
A22	nUSBOE	**C7**	SA[13]
A23	GND	**C8**	nSCS[1]
B1	OSCOUT	**C9**	SCKE[1]
B2	GND	**C10**	SDQMU
B3	nPOR	**C11**	PMADAPOK
B4	SA[2]	**C12**	PMBATOK
B5	SA[1]	**C13**	SD[11]
B6	SA[7]	**C14**	SD[13]
B7	SA[9]	**C15**	SD[15]
B8	nSCS[2]	**C16**	PORTC[0]
B9	SCKE[2]	**C17**	PORTC[1]
B10	nSWE	**C18**	PORTC[2]
B11	SD[7]	**C19**	PORTC[3]
B12	SD[6]	**C20**	UVM
B13	SD[5]	**C21**	GND
B14	SD[3]	**C22**	nPCACE[0]
B15	SD[1]	**C23**	nPCACE[1]
B16	PORTA[4]	**D1**	TMS
B17	PORTA[5]	**D2**	TCK
B18	PORTA[6]	**D3**	TDO
B19	PORTA[7]	**D4**	GND
B20	URCVIN	**D5**	nRESET
B21	UVP	**D6**	SA[8]
B22	GND	**D7**	SA[11]

Table 2-2: Pin Reference Numbers and Signal Names (continued)

Pin	Signal	Pin	Signal
D8	nSCS[0]	**E16**	PORTB[7]
D9	SCKE[0]	**E17**	VDD
D10	nSCAS	**E18**	PORTB[6]
D11	VDDcore	**E19**	GND
D12	VSScore	**E20**	nPCAIORD
D13	SD[4]	**E21**	nPCAIOWR
D14	SD[2]	**E22**	PCARESET
D15	SD[0]	**E23**	nPCAWAIT
D16	PORTC[4]	**F1**	LFP
D17	PORTC[5]	**F2**	LAC
D18	PORTC[6]	**F3**	LCP
D19	PORTC[7]	**F4**	LD[11]
D20	GND	**F5**	IRDIN
D21	nPCREG	**F19**	PCBIPORTE
D22	nPCBCE[0]	**F20**	nPCBIORD
D23	nPCBCE[1]	**F21**	nPCBIOWR
E1	LLP	**F22**	PCBRESET
E2	nTRST	**F23**	nPCBWAIT
E3	LCDEN	**G1**	LD[10]
E4	LBLEN	**G2**	LD[9]
E5	GND	**G3**	LD[8]
E6	IRDOUT	**G4**	LD[7]
E7	VDD	**G5**	VDD
E8	nSRAS	**G19**	VDD
E9	VDD	**G20**	PCABVD[0]
E10	SCLK	**G21**	PCABVD[1]
E11	nPMWAKEUP	**G22**	PCCADRV
E12	GND	**G23**	PCAREADY
E13	VDD	**H1**	LD[6]
E14	PORTD[6]	**H2**	LD[5]
E15	VDD	**H3**	LD[4]

Table 2-2: Pin Reference Numbers and Signal Names (continued)

Pin	Signal	Pin	Signal
H4	LD[3]	**L5**	VDD
H5	LD[2]	**L19**	PORTD[7]
H19	nPCAWE	**L20**	VDDcore
H20	PCBBVD[0]	**L21**	PCAVPPEN[0]
H21	PCBBVD[1]	**L22**	PCAVS[0]
H22	PCCBDRV	**L23**	PCAVS[1]
H23	PCBREADY	**M1**	VT[1]
J1	LD[1]	**M2**	VGAREF
J2	LD[0]	**M3**	VT[0]
J3	VGAAVDD[0]	**M4**	VSScore
J4	VGAHS	**M5**	GND
J5	VDD	**M19**	GND
J19	VDD	**M20**	VSScore
J20	nPCAOE	**M21**	PCAVPPEN[1]
J21	PCAWP	**M22**	PCBVS[0]
J22	nPCACD[0]	**M23**	PCBVS[1]
J23	nPCACD[1]	**N1**	AVDDDAC
K1	VGAROUTP	**N2**	VBIAS
K2	VGAROUTM	**N3**	ADAC[0] (R)
K3	VGAGOUTM	**N4**	VDDcore
K4	VGAVS	**N5**	PLLVSS[0]
K5	VGAAVSS	**N19**	VDD
K19	nPCBWE	**N20**	PCBVPPEN[0]
K20	nPCBOE	**N21**	PCBVPPEN[1]
K21	PCBWP	**N22**	PCAVCCEN[0]
K22	nPCBCD[0]	**N23**	PCAVCCEN[1]
K23	nPCBCD[1]	**P1**	PLLVDD[0]
L1	VGABOUTM	**P2**	ADAC[1] (L)
L2	VGABOUTP	**P3**	PLLFILT[0]
L3	VGAAVDD[1]	**P4**	AVSSDAC
L4	VGAGOUTP	**P5**	AVDDADC

Table 2-2: Pin Reference Numbers and Signal Names (continued)

Pin	Signal	Pin	Signal
P19	RD[23]	**U20**	RD[13]
P20	RD[27]	**U21**	RD[17]
P21	RD[30]	**U22**	RD[20]
P22	PCBVCCEN[0]	**U23**	RD[24]
P23	PCBVCCEN[1]	**V1**	nSSCS
R1	PLLFILT[2]	**V2**	SSDIN
R2	PLLFILT[1]	**V3**	ATSYM
R3	ADIN[0]	**V4**	SSCLK
R4	PLLVDD[1]	**V5**	MCLK
R5	VDD	**V19**	RD[5]
R19	VDD	**V20**	RD[9]
R20	RD[22]	**V21**	RD[12]
R21	RD[26]	**V22**	RD[16]
R22	RD[29]	**V23**	RD[19]
R23	RD[31]	**W1**	ATSXP
T1	ADIN[1]	**W2**	ATSYP
T2	PLLVSS[1]	**W3**	ATSXM
T3	ADIN[4]	**W4**	MDOUT
T4	ADIN[2]	**W5**	GND
T5	PORTD[5]	**W6**	nURTS[1]
T19	RD[14]	**W7**	VDD
T20	RD[18]	**W8**	KSCANO[6]
T21	RD[21]	**W9**	VDD
T22	RD[25]	**W10**	KSCANI[5]
T23	RD[28]	**W11**	VDD
U1	AVREFADC	**W12**	GND
U2	ADIN[3]	**W13**	PORTD[4]
U3	SSOUT	**W14**	RA[13]
U4	AVSSADC	**W15**	VDD
U5	VDD	**W16**	RA[4]
U19	VDD	**W17**	VDD

Table 2-2: Pin Reference Numbers and Signal Names (continued)

Pin	Signal	Pin	Signal
W18	nRWE[3]	**AA3**	GND
W19	GND	**AA4**	USIN[1]
W20	RD[4]	**AA5**	nUDCD[1]
W21	RD[8]	**AA6**	nUDSR[1]
W22	RD[11]	**AA7**	KSCANO[9]
W23	RD[15]	**AA8**	KSCANO[4]
Y1	MDFR	**AA9**	KSCANO[0]
Y2	MRLY	**AA10**	KSCANI[3]
Y3	MDIN	**AA11**	nRCS[5]
Y4	GND	**AA12**	EXBCLK
Y5	nUDTR[1]	**AA13**	RA[23]
Y6	USOUT[1]	**AA14**	RA[20]
Y7	KSCANO[10]	**AA15**	RA[16]
Y8	KSCANO[5]	**AA16**	RA[11]
Y9	KSCANO[1]	**AA17**	RA[7]
Y10	KSCANI[4]	**AA18**	RA[2]
Y11	KSCANI[0]	**AA19**	nRCS[2]
Y12	VSScore	**AA20**	nRWE[1]
Y13	VDDcore	**AA21**	GND
Y14	RA[17]	**AA22**	RD[2]
Y15	RA[12]	**AA23**	RD[6]
Y16	RA[8]	**AB1**	RTCOSCIN
Y17	RA[3]	**AB2**	GND
Y18	nROE	**AB3**	nMCON
Y19	nRWE[2]	**AB4**	nUCTS[1]
Y20	GND	**AB5**	nURTS[0]
Y21	RD[3]	**AB6**	nUDTR[0]
Y22	RD[7]	**AB7**	KSCANO[8]
Y23	RD[10]	**AB8**	KSCANO[3]
AA1	RTCOSCOUT	**AB9**	KSCANI[7]
AA2	MRING	**AB10**	KSCANI[2]

Table 2-2: Pin Reference Numbers and Signal Names (continued)

Pin	Signal	Pin	Signal
AB11	nRCS[4]	**AC19**	RA[5]
AB12	EXPRDY	**AC20**	RA[0]
AB13	RA[25]	**AC21**	nRCS[0]
AB14	RA[22]	**AC22**	RD[0]
AB15	RA[19]	**AC23**	GND
AB16	RA[15]		
AB17	RA[10]		
AB18	RA[6]		
AB19	RA[1]		
AB20	nRCS[1]		
AB21	nRWE[0]		
AB22	GND		
AB23	RD[1]		
AC1	GND		
AC2	USOUT[0]		
AC3	USIN[0]		
AC4	nUCTS[0]		
AC5	nUDSR[0]		
AC6	nUDCD[0]		
AC7	KSCANO[7]		
AC8	KSCANO[2]		
AC9	KSCANI[6]		
AC10	KSCANI[1]		
AC11	nRCS[3]		
AC12	BOOTBIT[0]		
AC13	BOOTBIT[1]		
AC14	RA[24]		
AC15	RA[21]		
AC16	RA[18]		
AC17	RA[14]		
AC18	RA[9]		

Table 2-2: Pin Reference Numbers and Signal Names (continued)

3 Architecture Overview

Architecture Overview

3.1 Internal bus structure

Figure 3-1: GMS30C7201 shows a block diagram of GMS30C7201. GMS30C7201 consists of the ARM720T processor core, a Piccolo SP7 DSP coprocessor and a set of peripherals.

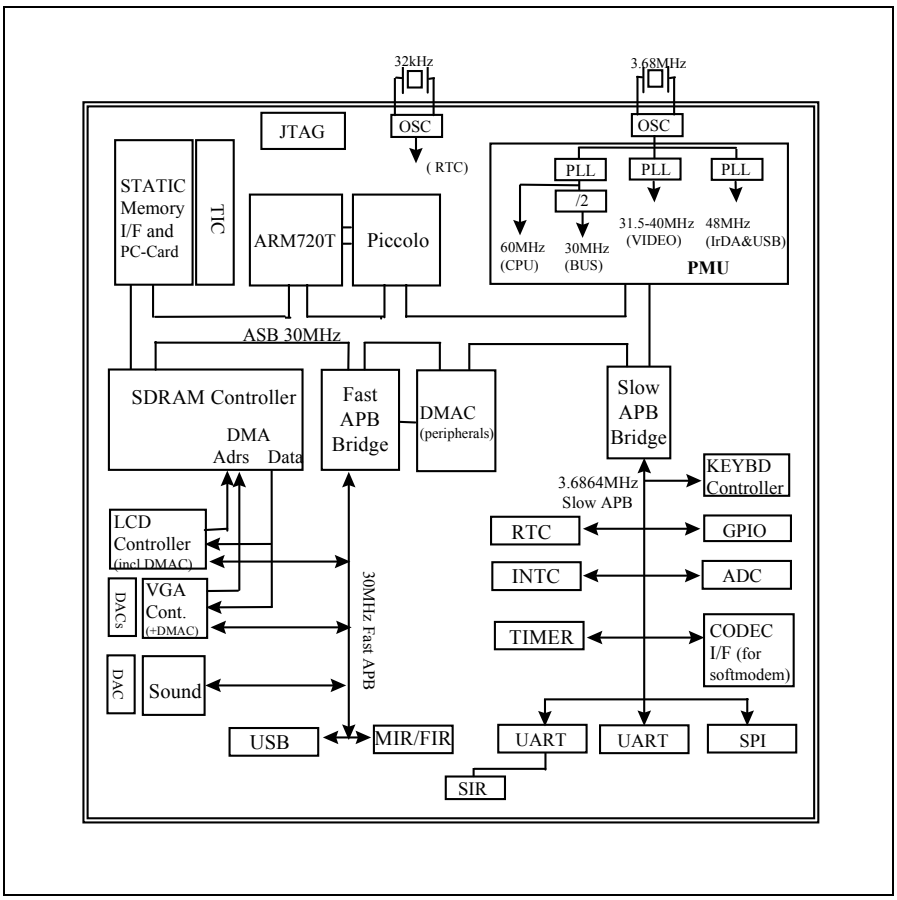


Figure 3-1: GMS30C7201

3.1.1 GMS30C7201 bus structure

The GMS30C7201 internal bus organization is based upon the AMBA standard, but with some minor modifications to the peripheral buses (the APBs). There are three main buses in the GMS30C7201:

1. the main system bus (the ASB) to which the CPU and memory controllers are connected
2. the fast APB to which high-bandwidth peripherals are connected
3. the slow APB (to which timers, the UART and other low-bandwidth peripherals are connected)

There is also a separate video DMA bus.

3.1.2 ASB

The ASB is designed to allow the ARM to have continuous access to both the ROM/PCMCIA interface and the SDRAM. The SDRAM controller straddles both the ASB and the video DMA bus so the LCD and VGA can access the SDRAM controller simultaneously with activity on the ASB. This means that the ARM or Piccolo can read code from ROM, or access a peripheral, without being interrupted by video DMA.

The GMS30C7201 uses a modified arbiter to control mastership on the main ASB bus. The arbiter only arbitrates on quad-word boundaries, or when the bus is idle. This is to get the best performance with the ARM720T, which uses a quad-word cache line, and also to get the best performance from the SDRAM, which uses a burst size of eight halfwords per access. By arbitrating only when the bus is idle or on quad-word boundaries (A[3:2] = 11), it ensures that cache line fills are not broken up, hence SDRAM bursts are not broken up.

Video ASB arbitration is controlled by the SDRAM controller. This is explained in ***3.2.2 Arbitration*** on page 3-4.

3.1.3 Video bus

The video bus hosts the LCD controller and the VGA controller DMA. The video bus consists of separate address inputs, a request / acknowledge to / from the SDRAM controller, for each of the LCD and VGA blocks, and a shared data bus. The LCD and VGA registers are programmed from the fast APB. The SDRAM controller arbitrates between ASB, VGA and LCD access requests. Video always has higher priority than ASB access requests. The split ASB/video bus architecture of the SDRAM controller allows slow device accesses—such as access to a PC-Card that asserts a **WAIT** signal for several microseconds—without blocking video DMA.

3.1.4 APBs

There are two APB buses. These are the *fast* and *slow* APB buses. The fast APB bus operates at the speed of the ASB (30 MHz), and hosts the Fast and Medium speed infra-red interface, the USB interface, the sound output interface, and the LCD and VGA registers. These are the high performance peripherals, which are generally DMA targets.

The slow APB peripherals generally operate at the UART crystal clock frequency of 3.6864MHz, though register access via the APB is at ASB speed. The slow APB peripherals do not support DMA transfers. This arrangement of operating most of the peripherals from a slower clock, and reducing the load on the faster bus, results in significantly reduced power consumption. Both APB buses connect to the main ASB bus via specially modified bridges.The slow APB bridge takes care of all re-synchronization, handing over data and control signals between the ASB and UART clock domains in a safe and reliable manner.

The fast APB bridge is modified from the normal AMBA bridge, to allow DMA access to fast APB peripherals. Additional signals from the DMA controller to the APB bridge request, select and acknowledge DMA transfers to and from DMA-aware peripherals.

3.2 SDRAM controller

3.2.1 Overview

The SDRAM controller is a key part of the GMS30C7201 architecture. The SDRAM controller has two data ports—one for video DMA and one for the main ASB—and interfaces to a single 16-bit wide SDRAM. One to four 16, 64 or 128Mbit x16-bit devices are supported, giving a memory size ranging from 2 to 64Mbytes.

3.2.2 Arbitration

The main ASB and video DMA buses are independent, and operate concurrently. The SDRAM controller contains the arbitration logic, for selecting between the two buses, and between the LCD and VGA on the video bus. The video bus is always higher priority than the main bus. The SDRAM controller uses a modified round-robin arbitration between the LCD and the VGA, but allows the highest priority device on the video bus to be programmed as either the LCD or the VGA.

The video interface consists of separate address and request inputs to the SDRAM controller, and shared data but individual acknowledge outputs to the VGA and the LCD. The video access burst size is fixed to 16 words. The address is non-incrementing for words within a burst (as the SDRAM controller only makes use of the first address for each burst request).

The arbitration scheme is modified round-robin. When the bus is idle, prioritization is fixed, with whichever LCD or VGA is programmed as highest priority getting data next, if they request data at the same time. When the bus is busy, the prioritization becomes round-robin, so if the higher priority device wants two bursts one immediately after the other, but the other device also requested a burst at the same time as the higher priority device, (or after, but before the first burst to the other device completed), then the lower priority device would get the second burst, and the higher priority device the third burst.

In use, the highest bandwidth video device (VGA or LCD) should be programmed as highest priority in the SDRAM controller. If both devices are equal priority, and use the same bandwidth, an arbitrary decision can be made. If only one video device is being used (for example, only LCD is being used, as will most often be the case), then that device should have the highest priority.

3.3 Peripheral DMA

3.3.1 Overview

GMS30C7201 incorporates a three-channel, general-purpose DMA controller which operates on the ASB. The DMA controller is an AMBA compliant ASB bus master with a higher arbitration priority than either the ARM or Piccolo DSP coprocessor, to ensure low DMA latency. Since, however, the main ASB bus always has lower priority access to the SDRAM controller than the video bus, it will always get lower priority access to SDRAM than the LCD and VGA.

3.3.2 Transfer sizes

The devices that make use of the peripheral DMA are:

1 USB
2 Fast/Medium IR
3 Sound output

The USB and FIR are bidirectional but half-duplex, so only one DMA channel is required at a time. The data rate for the USB is 12Mbit/sec, which translates to 1.5Mbyte/sec. The data rate for the FIR is a maximum of 4Mbit/sec, which translates to 0.5Mbyte/sec. The sound output data rate is 88.2KB/sec. To ensure reasonable usage of SDRAM, APB and ASB bandwidth, the transfer sizes to these device are:

USB	Quad-word
FIR	Word
Sound	Word

The SDRAM controller will do a complete quad-word access for every SDRAM access. With the transfer sizes above, the approximate SDRAM bandwidth taken by the devices is:

USB	3%
FIR	4%
Sound	0.75%

The maximum total of SDRAM bandwidth taken by all three devices running concurrently is 7.75%.

DMA accesses to FIR and Sound blocks are fully AMBA compliant, meaning that a word transfer takes a minimum of two bus cycles to complete. The APB protocol however, for USB DMA accesses, has been slightly modified to allow burst accesses.

3.3.3 Fly-by

The DMA controller is tightly coupled to the fast APB bridge. In order for the DMA Controller to start a transfer, it must first receive a DMA data request from one of the peripherals; it will then request mastership of the ASB.

Once granted, the DMA Controller will retain mastership of the ASB until the requested DMA transaction is completed, which ensures correct data in the DMA peripherals (that is data in the DMA peripherals cannot be modified by the ARM processor while a DMA transfer is in progress).

The DMA transfer request is monitored by the Fast APB bridge, who will perform the correspondent APB transfer by inverting the read/write line with respect to the ASB, to generate a **PWRITE** signal on the APB. The DMA transfer is acknowledged on the APB by asserting a **PSELDMA** signal for the given peripheral. The data is timed by **PSTB** as on a normal APB transfer. The APB address **PA** is not used for DMA transfers.

The APB bridge receives two signals from the DMA controller called **CHAN[1:0]**, which tells it which DMA channel (peripheral) the DMA access is for. All other information comes from monitoring the ASB bus signals. For example, the direction of transfer comes from **BWRITE** (the sense is inverted to get the APB signal), and when the SDRAM transfer completes, comes from the bridge monitoring the **BWAIT** ASB signal.

3.3.4 Timing

This is detailed in ***Chapter 12, Fast AMBA Peripherals***.

3.3.5 Slow APB peripherals

Since the DMA controller is not coupled with the slow APB bridge, it is not possible to use DMA with devices on the slow APB bus. However, since devices on the slow APB bus are inherently low performance, this is not a serious restriction. Devices on the slow APB bus must use the ARM acting under interrupt control to simulate DMA. The highest data rate peripheral on the slow APB bus is the modem CODEC interface, at a maximum of 48KB/sec. The ARM FIQ is used to transfer data to the CODEC.

3.3.6 Sound output

In the GMS30C7201, the sound peripheral is located on the fast APB bus, and is supported by the DMA controller. (Note that this is compatible with some operating systems, which require DMA-support sound hardware.)

3.4 Power management

The GMS30C7201 is designed for battery-powered portable applications and incorporates innovative design features in the bus structure and the PMU to reduce power consumption. The slow APB bus allows peripherals to be clocked slowly hence reducing power consumption. The use of three buses reduces the number of nodes that are toggled during a data access, thereby further reducing power consumption. In addition, clocks to peripherals which are not active can also be gated.

3.4.1 Clock gating

The high performance peripherals, such as the SDRAM controller and the LCD controller, run most of the time at high frequencies and careful design, including the use of clock gating, has minimized their power consumption. The VGA controller can be powered down completely when not in use (that is, when not connected to an external monitor).

3.4.2 PMU

The Power Management Unit (PMU) is used to control the overall state the system is in. The system can be in one of five states:

Run

The system is running normally. All clocks are running (except where gated locally), and the SDRAM controller is performing normal refresh.

Slow

The system operates normally, except the ARM is placed into Fast Bus mode, and hence is clocked at half its normal rate.

Idle

In this mode, the PMU becomes the bus master until there is an interrupt for the CPU, or the peripheral DMA controller requests mastership of the bus.

Sleep

The SDRAM is placed into self-refresh mode, and internal clocks are gated off. This mode can only be entered from Idle mode (that is, the PMU must be ASB master before this mode can be entered). The PMU must get bus mastership to ensure that the system is stopped in a safe state and not, for example, half-way through an SDRAM write.

Usually this state is only to be entered briefly, on the way to entering deep sleep mode.

Deep Sleep

In deep sleep mode, the 3.6864MHz oscillator and the PLLs are disabled. This is the lowest power state available. Only the 32kHz oscillator runs. The real time clock and wakeup sections of the PMU are operated from this clock. Everything else is powered down, and SDRAM is in self-refresh mode. This is the normal system “off” mode.

Sleep and Deep Sleep modes are exited either by a user wake-up event (generally pressing the “On” key), an RTC wake-up alarm, a device reset request, or by a modem ring indicate event. These interrupt sources go directly to the PMU. In addition, the modem ring indicate signal also goes to the normal interrupt controller to signal an interrupt if there is a ring indicate event in a non-sleep mode.

3.5 Performance

ARM720T operation

The actual performance of the device will be highly application specific as well as dependent on the speed of memory attached and the rate at which data is being transferred to the DMA peripherals, in particular the LCD and VGA.

If a particular application or part of application is executing entirely from within the 720T cache then the memory speed and peripheral DMA bandwidth may not have any affect at all. However most applications will require access to either the static memory interface or the SDRAM interface for its data structures and for instructions whenever there are cache misses. The cache miss rate will have a large impact on the achievable performance, however it is impossible to predict this for any general application.

It is possible though to give some indication of the potential performance of the device when operating out of cache.

Run Mode operation (Cache on and operating at 60MHz, ASB clock at 30MHz): Approximately 75K Dhrystones/s (~42.6 DMIPS)

Slow Mode operation (Cache on and operating at the ASB clock frequency of 30MHz): Approximately 44K Dhrystones/s (~25 DMIPS)

* Both sets of figures assume two wait state memory external memory is available and no peripheral DMA is active.

SP7 (Piccolo) operation

Once again it is not possible to generalize about the performance achievable on ARM720T when the SP7 co-processor (Piccolo) is being used, it will depend entirely on the frequency with which the ARM720T must transfer data to or from the SP7 co-processor which in turn is entirely application dependent.

The performance of the ARM7TDSP processing element is best illustrated by its performance for particular applications - it not possible to specify its performance in generalized MIPS terms.

SoftModem Performance

The following performance benchmarks have two columns. The first indicates the peak processor requirement when averaged over two frames (triple frame buffering is used to ensure no processor time overflow occurs). The peak occurs during the modem startup (training) sequence. The average value represents the softmodem requirement once the startup sequence has finished. All figures represent a system with data held in SDRAM, code held in 2 wait state burst ROM, and 620x240 monochrome 4bpp LCD being displayed. The table below assumes the CPU clock is running at 60MHz and the memory clock at 30MHz.

Modem standard	peak/MHz	average/MHz
V34bis with V42bis	54	42
v32bis	29	21
V17fax	14	14

Table 3-1: CPU clock cycles used for SoftModem

FFT performance

The following table illustrates the number of cycles it would take for an ARM720T and an ARM720T with SP7 to perform a number of FFT points. All values assume perfect memory:

FFT points	ARM720T cycles	ARM720T+ SP7 cycles	Speedup
2048	450,000	83,788	5.4
1024	200,000	38,215	5.2
512	90,000	17,282	5.2
256	40,000	7,740	5.2

Table 3-2: FFT performance for ARM versus ARM+Piccolo

4 ARM720T Macrocell

4.1 ARM720T Macrocell

For details of the ARM720T, please refer to the *ARM720T Data Sheet* (DDI 0087).

5 Piccolo Macrocell

5.1 Piccolo Macrocell

This section outlines

- the features and benefits of Piccolo
- the integration of Piccolo into the GMS30C7201

and provides references to other sources of information.

5.1.1 Features and Benefits of Piccolo

The full performance and functionality of a 16-bit fixed-point DSP is added to the ARM CPU through the Piccolo coprocessor (SP7 core). This is achieved without the high cost of a Harvard memory system as used in a conventional DSP architecture.The ARM architecture allows sustained single-cycle, multiply-accumulate type instructions from a single memory system. The 32-bit ARM memory system is free for high bandwidth data access through the use of a 128-word instruction cache which decouples Piccolo instruction fetches from the bus.

5.1.2 Piccolo DSP integrated into the GMS30C7201

Piccolo is connected as a coprocessor to the ARM720T, as shown in ***Figure 5-1: ARM720TDSP Block Diagram*** below. The combination is referred to as the ARM720TDSP.

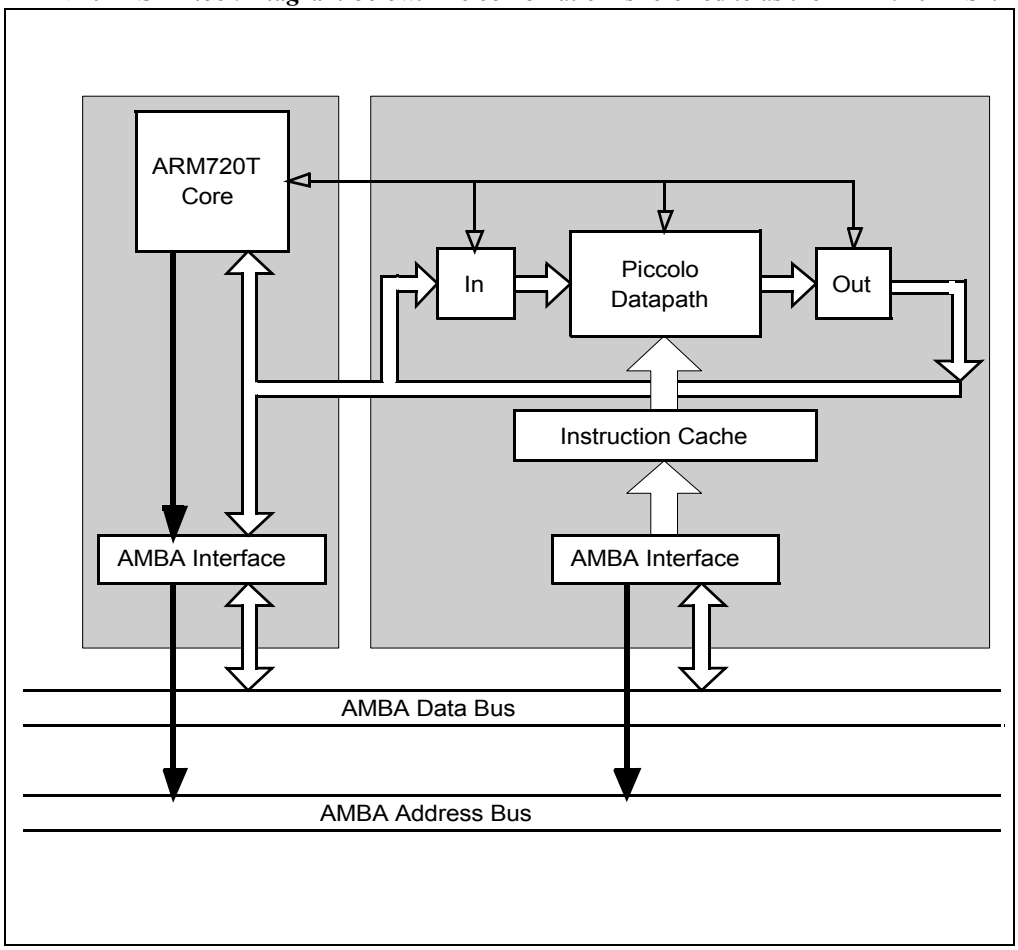


Figure 5-1: ARM720TDSP Block Diagram

The simplified diagram ***Figure 5-2: ARM720TDSP Connected to External Memory*** below shows how the ARM720TDSP is connected to external memory.

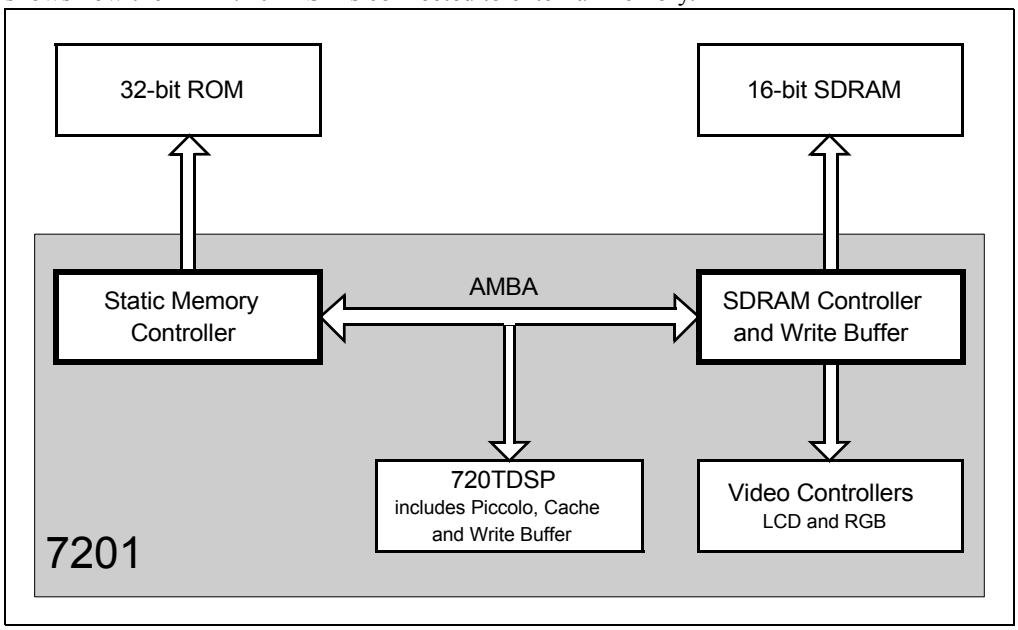


Figure 5-2: ARM720TDSP Connected to External Memory

The above enables a low cost/high-performance softmodem solution. V34bis (with V42bis compression) softmodem can comfortably run on the GMS30C7201, assuming a 640x240 color LCD and 8bpp operation.

5.1.3 References to other documentation

For details of the Piccolo Macrocell, please refer to the *ARMSP7 Data Sheet* (ARM DDI 0089).

Further information can be found in the following documents:

- *Introduction to Piccolo* (ARM DVI 0006)
- *ARM Piccolo DSP Benchmarks* (ARM DVI 0008)
- *ARM Signal Processing Architecture*, (ARM DDI 0128)
- *ARM Signal Processing Architecture, Architecture Reference Manual* (ARM IP 0025)

6 Memory Map

Memory Map

6.1 Introduction

There are five main memory map divisions, outlined in ***Table 6-1: Top-level address map***.

Base Address	Size	Description
0Mbyte	64Mbytes	ROM chip select 0
64Mbytes	64Mbytes	ROM chip select 1
128Mbytes	64Mbytes	ROM chip select 2
192Mbytes	64Mbytes	ROM chip select 3
256Mbytes	64Mbytes	ROM chip select 4
320Mbytes	64Mbytes	ROM chip select 5
384Mbytes	128Mbytes	Reserved
512Mbytes	64Mbytes	PCMCIA card 1Attribute memory
576Mbytes	64Mbytes	PCMCIA card 1Common memory
640Mbytes	64Mbytes	PCMCIA card 1I/O or Secondary Common memory
704Mbytes	64Mbytes	Reserved
768Mbytes	64Mbytes	PCMCIA card 2 Attribute memory
832Mbytes	64Mbytes	PCMCIA card 2 Common memory
896Mbytes	64Mbytes	PCMCIA card 2 I/O or Secondary Common memory
960Mbytes	64Mbytes	Reserved
1024Mbytes	16Mbytes	SDRAM chip select 0
1040Mbytes	16Mbytes	SDRAM chip select 1
1056Mbytes	16Mbytes	SDRAM chip select 2
1072Mbytes	16Mbytes	SDRAM chip select 3
1088Mbytes	16Mbytes	SDRAM mode register chip 0
1104Mbytes	16Mbytes	SDRAM mode register chip 1
1120Mbytes	16Mbytes	SDRAM mode register chip 2
1136Mbytes	16Mbytes	SDRAM mode register chip 3
1152Mbytes	896Mbytes	Reserved
2048Mbytes	336Kbytes	Peripherals

Table 6-1: Top-level address map

The ROM has an address space of 384Mbytes which is split equally between six external ROM chip selects.

PCMCIA 0 and 1 have three contiguous 64Mbyte pages.

Note: *The 64Mbyte address space at the top of the 256Mbyte allocated for each PCMCIA card is reserved.*

There is a maximum of 64Mbytes of SDRAM. The mode registers in the SDRAM are programmed by reading from the 64Mbyte address space immediately above the SDRAM.

The peripheral address space is subdivided into three main areas: those on the ASB, the fast APB and the slow APB. The base address for the peripherals is given in ***Table 6-2: Peripherals base addresses***.

AMBA Base Address (Hex)	Name	Description
ASB peripherals		
2Gbyte	SDRAMC Base	SDRAMC
2Gbyte + 0x1000	PMU Base	PMU/PLL
2Gbyte + 0x2000	PCMCIA Base	PCMCIA
2Gbyte + 0x3000	BUSC Base	Bus controller
2Gbyte + 0x4000	DMAC Base	DMAC
2Gbyte + 0x5000 - 2Gbyte + 0xFFFF	Reserved	
Fast APB peripherals		
2Gbyte + 0x10000	Video Base	LCD/VGA
2Gbyte + 0x11000	IR Base	MIR/FIR
2Gbyte + 0x12000	USB Base	USB
2Gbyte + 0x13000	Sound Base	SOUND
2Gbyte + 0x14000 - 2Gbyte + 0x1FFFF	Reserved	
Pseudo DMA		
2Gbyte + 0x51000	IR Base	MIR/FIR - ARM accesses as DMA bus master
2Gbyte + 0x52000	USB Base	USB - ARM accesses as DMA bus master
2Gbyte + 0x53000	Sound Base	SOUND - ARM accesses as DMA bus master
2Gbyte + 0x54000 - 4Gbyte-1	Reserved	
Slow APB peripherals		
2Gbyte + 0x20000	U1 Base	UART 1
2Gbyte + 0x21000	U2 Base	UART 2
2Gbyte + 0x22000	KBD Base	KBD
2Gbyte + 0x23000	GPIO Base	GPIO
2Gbyte + 0x24000	INTC Base	INTC
2Gbyte + 0x25000	Timer Base	TIMER
2Gbyte + 0x26000	SPI Base	SPI

Table 6-2: Peripherals base addresses

AMBA Base Address (Hex)	Name	Description
2Gbyte + 0x27000	Modem Base	MODEM
2Gbyte + 0x28000	RTCBase	RTC
2Gbyte + 0x29000	ADCBase	ADC
2Gbyte + 0x2A000 - 2Gbyte + 0x4FFFF	Reserved	

Table 6-2: Peripherals base addresses

6.2 Peripheral Register Map Summary

6.2.1 ASB peripherals

For details of the SDRAM controller registers, please refer to ***Table 8-4: SDRAM controller registers*** on page 8-5

For details of PMU/PLL registers, please refer to ***Table 7-1: PMU register map*** on page 7-8

6.2.2 PCMCIAC register address map

For details of PCMCIA register address maps, please see ***Table 10-31: Register Map Socket 1 and Socket 2 Addresses*** on page 10-21 and ***Table 10-32: Test Register Map*** on page 10-22

6.2.3 Static memory interface register address map

For details see ***Table 9-4: Static Memory Controller register map*** on page 9-7

6.2.4 DMAC register summary

For details of the DMAC registers address map, please refer to ***Table 12-2: DMAC register summary*** on page 12-6

6.3 High-speed APB Peripherals

6.3.1 Video system address map

For details of the video system address map, please refer to ***Table 11-15: LCD register map locations*** on page 11-25

6.3.2 Infra Red controller (MIR/FIR) register address map

The Ir Interface module registers occupy a 4k block of addresses within the fast APB peripheral area of the GMS30C7201 memory map. For details, please see ***Table 12-34: Ir Interface Block Registers and their Physical Addresses*** on page 12-52

6.3.3 USB register address map

For details of the USB register address map, please refer to ***Table 12-41: USB register address map*** on page 12-62.

6.3.4 Sound register address map

For details of the sound register address map, please refer to ***Table 12-44: Sound control unit register memory map*** on page 12-70.

6.3.5 Slow AMBA peripherals register map summary

For details, please see ***Table 13-1: Slow AMBA peripherals register map*** on page 13-2.

6.3.6 UART register address map

There are two UARTs implemented in the design. For details, please see ***Table 13-5: UART register address map*** on page 13-10.

6.3.7 Keyboard register address map

For details of the keyboard register address map, please see ***Table 13-16: Keyboard interface controller unit register memory map*** on page 13-25.

6.3.8 GPIO register address map

For details of the GPIO register address map, please see ***Table 13-28: PIO register memory map*** on page 13-36

Register Description	A[4:2]	Type (R/W)	Initial Value	Function
Data Register	000	W	00h	Outgoing data. Read returns the value of PAD.
Direction Register	001	R/W	00h	Selects the direction of each IO pin.
Interrupt Mask Register	010	R/W	00h	Masks each interrupt source: 0 = disable interrupt (default) 1 = enable interrupt
Interrupt Status Register	011	R	00h	Current interrupt request status (read only): 0 = no interrupt request 1 = interrupt pending (masked interrupt is always 0)

Table 6-3: GPIO registers

Register Description	A[4:2]	Type (R/W)	Initial Value	Function
Edge Mode Register	100	R/W	00h	Interrupt sources operate as edge mode: 0 = level mode (default) 1 = edge mode
Clear Register	101	W	00h	Clear pending interrupt source (edge mode only): 0 = no action (default) 1 = clear interrupt source (self-reset)
Polarity Register	110	R/W	00h	Interrupt sources operate as active HIGH/LOW 0 = active HIGH mode 1 = active LOW mode

Table 6-3: GPIO registers

6.3.9 Interrupt controller register address map

For details of the interrupt controller register address map, please refer to ***Table 13-30: Interrupt controller register map*** on page 13-40.

6.3.10 Timer registers address map

Please refer to ***Table 13-33: Timer port addresses*** on page 13-43 for details of the timer port registers.

6.3.11 SPI-MMC registers

Please refer to ***Table 13-35: SPI-MMC block register map*** on page 13-50 for details of the SPI-MMC registers.

6.3.12 AFE register address map

Please refer to ***Table 13-49: AFE Interface register memory map*** on page 13-62 for details of the AFE registers.

6.3.13 RTC register map

Please refer to ***Table 13-53: RTC register memory map*** on page 13-67 for details of the RTC register map.

6.3.14 AIC register address map

Please refer to ***Table 13-55: AIC unit register address map*** on page 13-71 for details of the AIC register address map.

Memory Map

7 PMU & PLL

PMU & PLL

7.1 Overview

The GMS30C7201 is designed primarily for HPC and other portable computing applications. Therefore there are 4 operating modes to reduce power consumption and extend battery life.

- RUN - normal operation (typically used when softmodem is in operation and other CPU-intensive tasks)
- SLOW - half-speed operation used when the application interacts with a user (e.g. word processing)
- IDLE - where the CPU operation is halted but peripherals operation continue (such as screen refresh, or serial communications)
- SLEEP & DEEP SLEEP - This mode will be perceived as ‘off’ by the user, i.e. the contents of SDRAM are maintained and only the real-time clock is running.

There are a number of Power management states, (see ***7.3 Power management states*** on page 7-5) as described above, and the transition between states is controlled by the PMU. The PMU is an ASB slave unit to allow the CPU to write to its control registers, and is an ASB master unit to provide the mechanism for stopping the ARM core’s internal clock).

7.2 Block Diagram

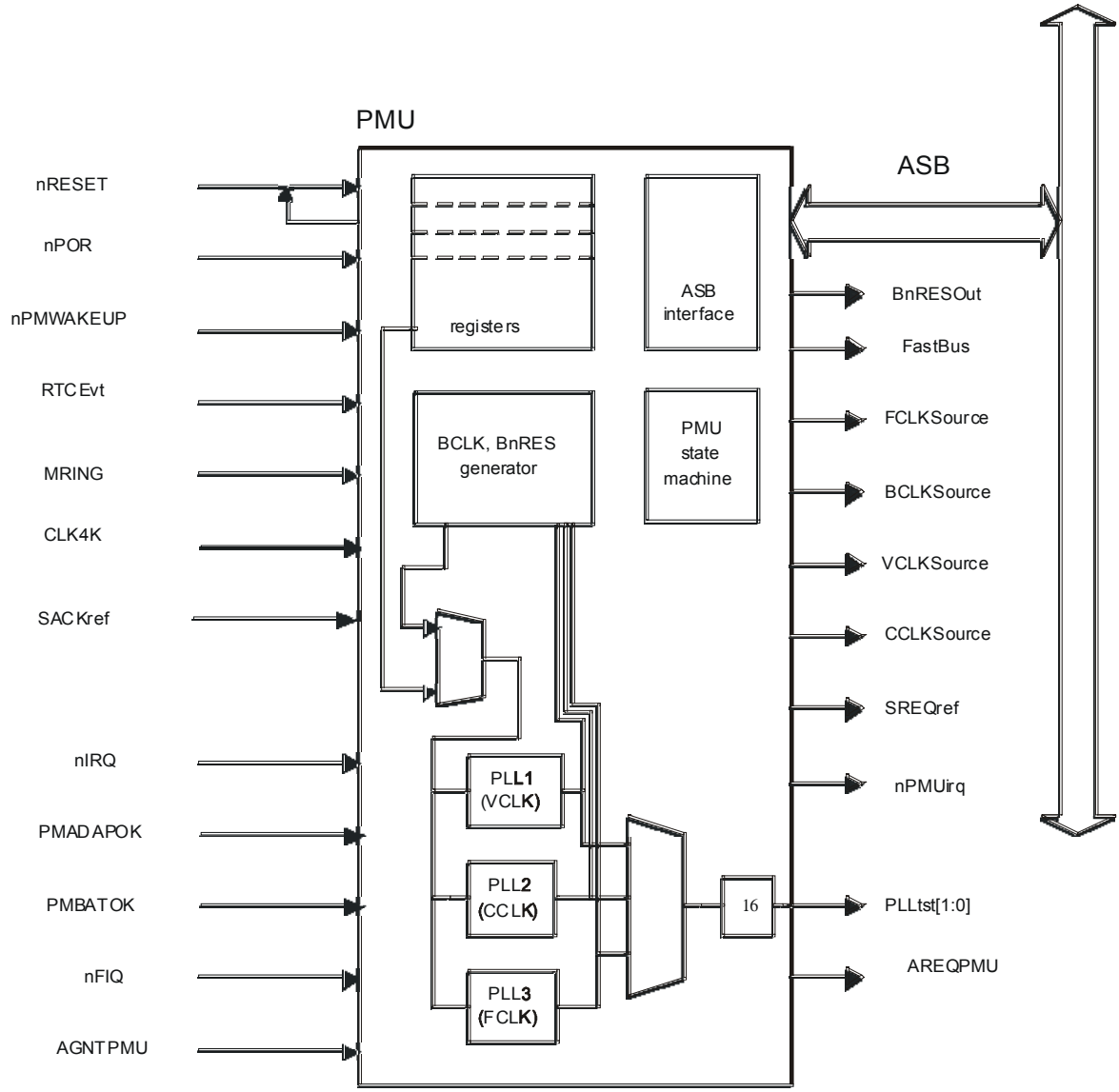


Figure 7-1: Block Diagram

Sub-Block Descriptions

CLOCK generator

The CLOCK generator module is responsible for controlling the PLL's and gating clocks while the outputs of the PLLs are uncertain and to ensure that clocks are available during test modes and during RESET sequences.

FCLK (ARM Processor and SDRAM controller clock)

Derived from PLL3 whose Frequency is controllable between 49.7664 MHz and 82.944 MHz. Frequency of operation is set using a 6 bit register.

There are two methods for updating frequency, depending upon the state of bit 6 of the Clock Control register ClkCtl (see ***ClkCtl register*** on page 7-11). If bit 6 is set, then any data written to bits [5:0] of the ClkCtl register are immediately transferred to the pins of PLL3, thus causing the loop to unlock and to mute FCLK. This is only a safe mode of operation if PLL3 frequency and mark-space ratio is guaranteed to be within limits immediately after the Lock Detect signal has become active. If bit 6 is NOT set, then the GMS30C7201 must enter DEEP sleep mode before bits [5:0] of the Clock Control register are transferred to PLL3.

To switch between the two frequencies when bit 6 is not set:

- Software writes the new value into the ClkCtl register
- Set a Real Time Clock Alarm to wake the GMS30C7201 in 2 seconds
- Enter DEEP SLEEP Mode by writing to the PMUMode Register
- The GMS30C7201 will power up with PLL3 running at the new frequency

BCLK

BusClock, which is generated by the PMU by dividing FCLK by 2.

VCLK

Clock for the LCD and VGA video controller. Frequency selectable between 31.5MHz or 40MHz. The VCLK PLL is disabled when on BnRES is active or when the PMU is put into DEEP SLEEP mode. On exit from either of these conditions, the VCLK PLL must be re-enabled by software.

Changing Frequency:

1. Software must first disable the VCLK pll, by writing a ‘0’ to the PLL1Enable bit of the ClkCtl register.
2. Write the new value to the PLL1Freq bit.
3. Re-enable the VCLK pll by writing 1 to the PLL1Enable bit.

CCLK

Clock for the IR comms and the USB. Nominally 48MHz. The CCLK PLL is disabled when BnRES active or when the PMU is put into DEEP SLEEP mode. On exit from either of these conditions, the CCLK PLL must be re-enabled by software.

PMU state machine

The state machine handles the transition between the power management states described below. The CPU can write to the PMU mode registers (which is what would typically happen when a user switches off the device) and the state machine will proceed to the commanded state.

7.3 Power management states

Run

The system is running normally. All Clocks running (except where gated locally). The SDRAM controller is performing normal refresh.

SLOW

The CPU is switched into FastBus mode, and hence runs at the BCLK rate (half the FCLK rate). This is the default mode after exiting SLEEP Mode.

IDLE

In this mode, the PMU becomes the bus master until there is either a fast or normal interrupt for the CPU, or the peripheral DMA controller requests master-ship of the bus. This will cause the clocks in the CPU to stop when it attempts an ASB access. Entry to this mode can be caused by the CPU writing the PMU_IDLE value to the PMU Mode Register when in RUN or SLOW modes, or a WakeUp signal becoming active when the PMU is SLEEP or DEEP SLEEP modes

SLEEP

In this mode, the SDRAM is put into self-refresh mode, and internal clocks are gated off. This mode can only be entered from IDLE mode (the PMU bus master must have mastership of the ASB before this mode can be entered). The PMU must be bus master to ensure that the system is stopped in a safe state, and is not half way through an SDRAM write (for example). Both the Video and Communication clocks should be disabled before entering this state.

Usually this state would only be entered briefly, on the way to entering DEEP SLEEP mode.

DEEP SLEEP

In DEEP SLEEP mode, the 3.6864MHz oscillator and the PLL are disabled. This is the lowest power state available. Only the 32KHz oscillator runs. The real time clock and the PMU are clocked from this clock. Clocked circuitry in the PMU runs from 4kHz (ie the RTC clock divided by 8). Everything else is powered down, and SDRAM is in self-refresh mode. This is the normal system “off” mode.

SLEEP and DEEP SLEEP modes are exited either by a user wake-up event (generally pressing the “On” key), or by an RTC wake-up alarm, or by a modem ring indicate event. These interrupt sources go directly to the PMU.

7.4 Power management

7.4.1 State Diagram

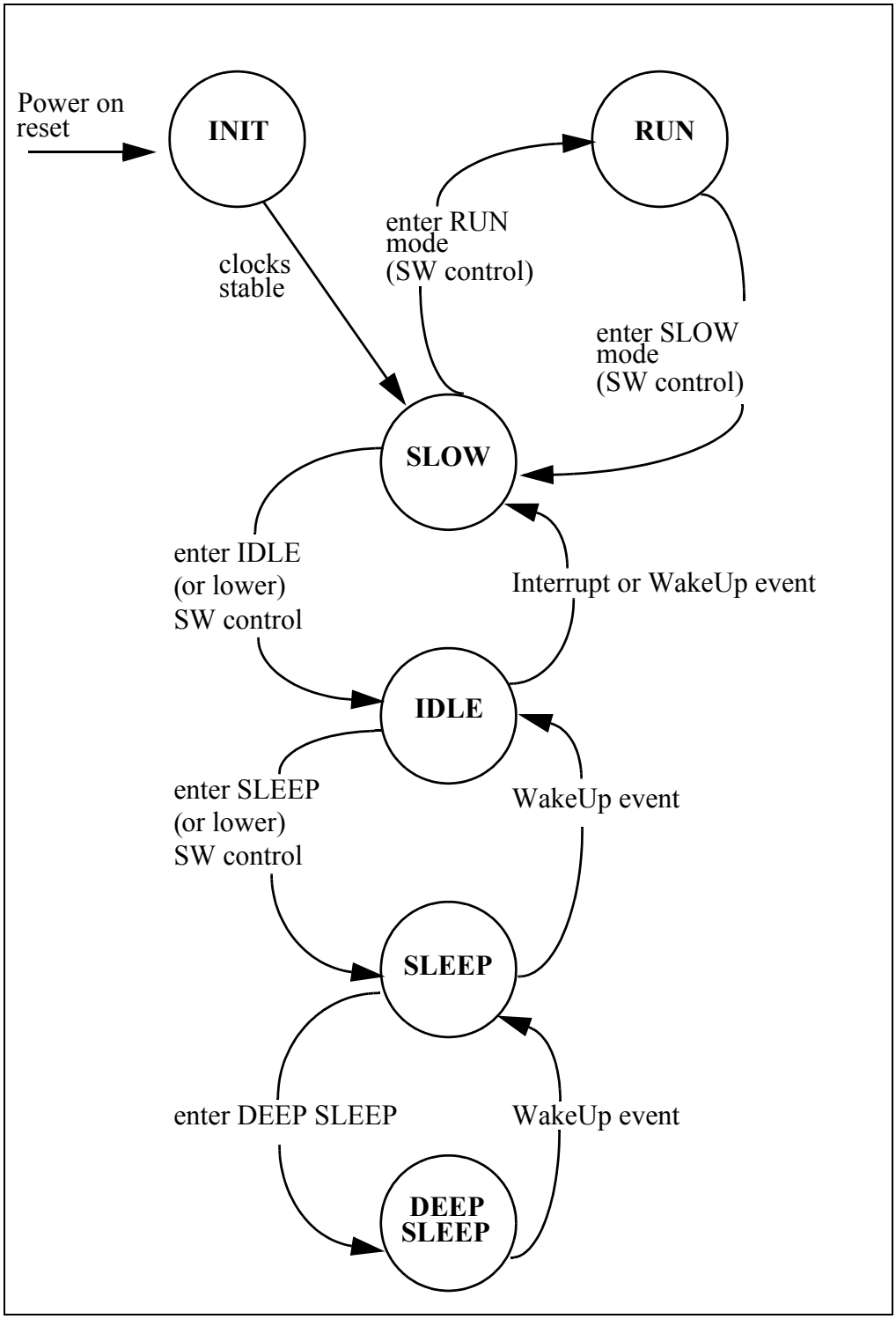


Figure 7-2: Power Management State Diagram

Wake-up Debounce and Interrupt

The Wake-up events are debounced as follows:

Each of the event signals which are liable to noise (**nRESET**, **RTC**, **nPMWAKEUP**, and **Modem Ring Indicator**, **Power Adapter Condition**) is re-timed to a 250Hz clock derived from the low power (4kHz) clock. After filtering to a quarter of 250 Hz, each event has an associated 'sticky' register bit. **nPMWAKEUP** is an external input, which may be typically connected to an "ON" key.

A 'sticky' bit is a register bit that is set by the incoming event, but is only reset by the CPU. Thus should a PLL drop out of lock momentarily (for example) the CPU will be informed of the event, even if the PLL has regained lock by the time the CPU can read its associated register bit.

The **nPMWAKEUP**, Modem, Real Time Clock and Power Adapter condition inputs are combined to form the PMU Interrupt. Each of these three interrupt sources may be individually enabled.

To make use of the **nPMWAKEUP** Interrupt, (for example) controlling software will need to complete the following tasks:

- Enable the **nPMWAKEUP** interrupt bit, by writing 0 to bit 9 of the ResetStatus register.
- Once an interrupt has occurred, read the RESET / Status register to identify the source(s) of interrupt. In the case of a **nPMWAKEUP** event, the register will return 0x10.
- Clear the appropriate 'sticky' bit by writing a 1 to the appropriate location (in the **nPMWAKEUP** case, this will be 0x10.).

PORTA Wake-up Sequence

The PORTA interrupt is OR gated with **nPMWAKEUP** to support additional wake up sources.

Each PORTA input signal can be used as a wake up source, they are enabled using the Interrupt MASK Register. After wake up, s/w should program the PORTA Interrupt MASK Register and/or the PMU ResetStatus Register.

One possible application is to use the **nDCD** signal, from the a UART interface, as a wake up source, by connecting **nDCD** to a PORTA input. In Deep Sleep mode, **nDCD** can wake up the system by generating a PORTA interrupt request to the PMU block. The PMU state machine then returns the system to the operational mode.

7.5 PMU Registers

The base address of the PMU registers (PMUBase) is defined in ***Table 6-2: Peripherals base addresses*** on page 6-3. The offsets from PMUBase of the PMU registers are described in ***Table 7-1: PMU register map***.

Address	Read location	Write location
PMUBase + 0x00	PMUMode	PMUMode
PMUBase + 0x08	Piccolo Enable	Piccolo Enable
PMUBase + 0x10	ID	Reserved
PMUBase + 0x18	Bus Retract	Bus Retract
PMUBase + 0x20	ResetStatus	ResetStatusClear
PMUBase + 0x28	ClkCtl	ClkCtl
PMUBase + 0x30	Debounce Counters (test only)	Debounce test register
PMUBase + 0x38	General Purpose Test	General Purpose Test

Table 7-1: PMU register map

PMU mode register

This read/write register is written to by the CPU to change mode from RUN mode or SLOW mode into a different mode. The encoding is shown below, in ***Table 7-2: Mode entry encodings***. Obviously the register can only be read and written to in RUN mode or SLOW mode, since these are the only modes in which the processor can access these registers. Therefore, the processor will never be able to read values for modes other than mode 0x00 and mode 0x 01. Other values may be read by a test controller so long as clocks are enabled with bit 8 of the DbCtr register. See ***Table 7-12: DbCtr Register Bit 8*** on page 7-13.

PMUMode[2:0] register value	PMU Mode
0x04	Initialisation mode
0x01	RUN mode
0x00	SLOW mode
0x02	IDLE mode
0x03	SLEEP mode
0x07	DEEP SLEEP mode
PMUMode[3]	Writing a ‘1’ to this bit allows PMU to exit DEEP SLEEP mode when pins **PMBATOK** and **PMADAPOK** are both low.Writing a ‘0’ to this bit prevents the PMU from leaving DEEP SLEEP mode when **PMBATOK** and **PMADAPOK** are both low

Table 7-2: Mode entry encodings

Note *All other values in the above table are undefined.*

Piccolo Enable Register

Bit	Name	Function
0	**PICENABLE**	Enable Piccolo coprocessor

Table 7-3: Piccolo Enable Register

Piccolo is enabled when ARM 7201 comes out of reset. Software should disable the Piccolo as soon as possible in the reset sequence to conserve power by writing a 0 to this location. Subsequently, software should only enable Piccolo when running an application that requires Piccolo (such as Soft Modem).

ID register

This read-only register returns a unique chip revision ID. Revision 0 of the GMS30C7201 device (the first revision), will return the constant value 0x00720100.

Bus Retract register

Bit	Name	Function
0	**BRDelay**	0: bus retracts after 8 cycles 1: bus retracts after 12 cycles
1	**BRENABLE**	Enables bus retracts

Table 7-4: Bus Retract Register

BRENABLE enables correct DMA operation when slow peripherals are connected to the external bus. When enabled, bus retracts occur when either **nPCAWAIT, nPCBWAIT** or **EXPRDY** are held active by a slow external peripheral for more than the number of clocks specified by **BRDelay.** The bus retract ensures the DMA is not stalled for the duration of the slow peripheral bus access

Reset / Status register

This read/write register provides status information on power on reset and the PLL status. The allocation is a shown in ***Table 7-6: ResetStatus Register Bits***. The bits in this register are ‘sticky’ bits. For a definition of a sticky bit please refer to ***Wake-up Debounce and Interrupt*** on page 7-7. Generally, this register will be read each time the ARM exits reset mode, so that the ARM can identify what event has caused it to exit from reset mode.

ResetStatus register READ bits	Register bit meaning
0	PORStatus
1	PLLLock1
2	PLLLock2
3	PLLLock3
4	OnEvt (debounced)
5	RIEvt (debounced)

Table 7-5: Table Reset and PLL Status Register

6	RTCEvt
7	ADPATOR NOT OK(debounced)
8	Warm RESET Event (debounced)
9	OnEvt Interrupt MASKPMU Interrupt Enable
10	RIEvt Interrupt MASKPMU Interrupt Request / Clear
11	RTCEvt Interrupt Mask
12	No External Power Interrupt Mask

Table 7-5: Table Reset and PLL Status Register

The meanings of the individual register bits are as defined in ***Table 7-6: ResetStatus Register Bits***.

Bit	Bit = 0	Bit = 1
PORStatus	No POR since last cleared	POR since last cleared
PLLLock1	VGA PLL has been locked since last cleared	VGA PLL has fallen out of lock since last cleared
PLLLock2	Comms PLL has been locked since last cleared	Comms PLL has fallen out of lock since last cleared
PLLLock3	System PLL has been locked since last cleared	System PLL has fallen out of lock since last cleared
OnEvt	No On key event since last cleared	On key event since last cleared
RIEvt	No Modem Ring Indicate wake-up event since last cleared	Modem Ring Indicate wake-up event since last cleared
RTCEvt	No Real Time Clock (RTC) calendar wake-up event since last cleared	Real Time Clock (RTC) calendar wake-up event since last cleared
PowerFailEvt	No PowerFail event since last cleared	A PowerFail event has occurred since last cleared
RESETEvt	No Warm RESET event has occurred	A Warm RESET event has ocurred since last cleared
Maskbits [9]	Disable PMU interrupt from **nPMWAKEUP**	Enable PMU interrupt from **nPMWAKEUP**
Maskbits [10]	Disable PMU interrupt from **MRING**	Enable PMU interrupt from **MRING**
Maskbits [11]	Disable PMU interrupt from **RTC**	Enable PMU interrupt from **RTC**
Maskbits [12]	Disable PMU interrupt from **PMADAPOK** LOW.	Enable PMU interrupt from **PMADAPOK** LOW.

Table 7-6: ResetStatus Register Bits

ResetStatus register WRITE bits	Register bit meaning
0	Writing a ‘1’ to this bit causes the nPOR event flag to be cleared. Writing a ‘0’ has no effect.
1	Writing a ‘1’ to this bit causes the PLL1 Unlock event flag to be cleared. Writing a ‘0’ has no effect.
2	Writing a ‘1’ to this bit causes the PLL2 Unlock event flag to be cleared. Writing a ‘0’ has no effect.
3	Writing a ‘1’ to this bit causes the PLL3 Unlock event flag to be cleared. Writing a ‘0’ has no effect.
4	OnEvt Interrupt Clear. Writing a ‘1’ to this bit clears a pending interrupt bit.
5	RI Interrupt Clear. Writing a ‘1’ to this bit clears a pending interrupt bit.
6	RTC Interrupt Clear. Writing a ‘1’ to this bit clears a pending interrupt bit.
7	Power Fail Interrupt Clear. Writing a ‘1’ to this bit clears a pending interrupt bit.
8	Warm Reset Clear. Writing a ‘1’ to this bit clears the event bit.
[12:9]	PMU Interrupts Enable. ‘1’ enables interrupts to the CPU, ‘0’masks such activity. Should the enable bit be set to one when one of the debounced event signals is set, then an interrupt WILL be generated (ie the interrupt is level sensitive, not edge sensitive).
13	Warm RESET. Writing a ‘1’ causes nRESET to be asserted. Writing ‘0’ has no effect.

Table 7-7: Status register WRITE bits

ClkCtl register

This register is used to control the frequency of PLL3, the system clock PLL and PLL1, the VGA clock. Six bits are defined which control the frequency of FCLK, and a further bit is used to control the frequency of PLL1, the VGA clock.

The Default (Power on Reset) value for this register is 0x1b.

ClkCtl[5:0]: PLL3Freq	Function
0x1B	49.7664 MHz
0x1C	51.6096 MHz
0x1D	53.4528 MHz
0x1E	55.2960 MHz
0x1F	57.1392 MHz
0x20	58.9824 MHz
0x21	60.8256 MHz
0x22	62.6688 MHz
0x23	64.5120 MHz

Table 7-8: ClkCtl Register

0x24	66.3552 MHz
0x25	68.1984 MHz
0x26	70.0416 MHz
0x27	71.8848 MHz
0x28	73.7280 MHz
0x29	75.5712 MHz
0x2A	77.4144 MHz
0x2B	79.2576 MHz
0x2c	81.1008 MHz
0x2D	82.9440 MHz
other values	reserved
ClkCtl[6]: PLL3 Frequency update	**Function**
0	PLL3 frequency control frequency is only updated when PMU exits DEEP SLEEP mode (default)
1	PLL3 frequency control frequency is updated instantaneously
ClkCtl[7]: PLL3Mute	**Function**
0	PLL3 is muted when Lock detect = 0 (default)
1	PLL3 only muted after nPOR or nRESET. Subsequent unlock condition does not mute the clock. Allows dynamic changes to the clock frequency without halting execution. Care: this only will be legal if PLL3 is under-damped (i.e. will not exhibit overshoot in its lock behavior).
ClkCtl[8]: PLL1Freq	**Function**
0	PLL1 set to max. frequency = 31.5MHz
1	PLL1 set to min. frequency = 40MHz
ClkCtl[9]: PLL1Enable	**Function**
0	PLL1 disabled
1	PLL1 enabled. Output will be gated until PLL1 Lock Detect (LD) is received
ClkCtl[10]: PLL2Enable	**Function**
0	PLL2 disabled
1	PLL2 enabled. Output will be gated until PLL2 Lock Detect (LD) is received

Table 7-8: ClkCtl Register

IF BIT 6 is ‘0’

When the CPU writes to bits 5:0 of this register, these bits are stored in a temporary buffer, which is not transferred to the PLL until the next time the PLL lock signal becomes inactive. This means that for a new value to take effect, it is necessary for the device to enter DEEP SLEEP mode first.

IF BIT 6 is ‘1’

The first effect that writing a new value to bits [5:0] will have is that PLL3 will go out of lock, and the Clock control circuit will immediately inhibit FCLK and BCLK, without first verifying that SDRAM operations have completed.

Debounce counter test register (read)

DbCtr[5:0]	Function
[3:0]	Prescaler bits
[5:4]	Selected debounce counter bits

Table 7-9: DbCtr Register (Read)

Debounce counter test register (write)

DbCtr[2:0]	Function
0x00	Selects debounce counter for nPMWAKEUP
0x01	Selects debounce counter for RING event
0x03	Selects debounce counter for POWER Adaptor event
0x04	Selects debounce counter for Warm Reset

Table 7-10: DbCtr Register Bits[2:0](Write)

DbCtr[3]	Function
1	forces local test mode
0	nTEST takes value from input pin
DbCtr[4]	**Function**
1	disables Bus Request from the PMU to allow CPU to read state machine for test purposes during PMU IDLE state.
0	normal operation

Table 7-11: DbCtr Register Bits[3:4]

DbCtr[8]	Function
1	Forces FCLK and BLCK to be active in all PMU states (test purposes only)
0	Normal operation

Table 7-12: DbCtr Register Bit 8

In order that the debounce counters (which would normally be clocked from 4kHz) may be independently exercised and observed, the counters may be triggered and observed using the above registers. These registers are for testing only and are not required in normal use.

RESET SEQUENCES

Power on RESET.

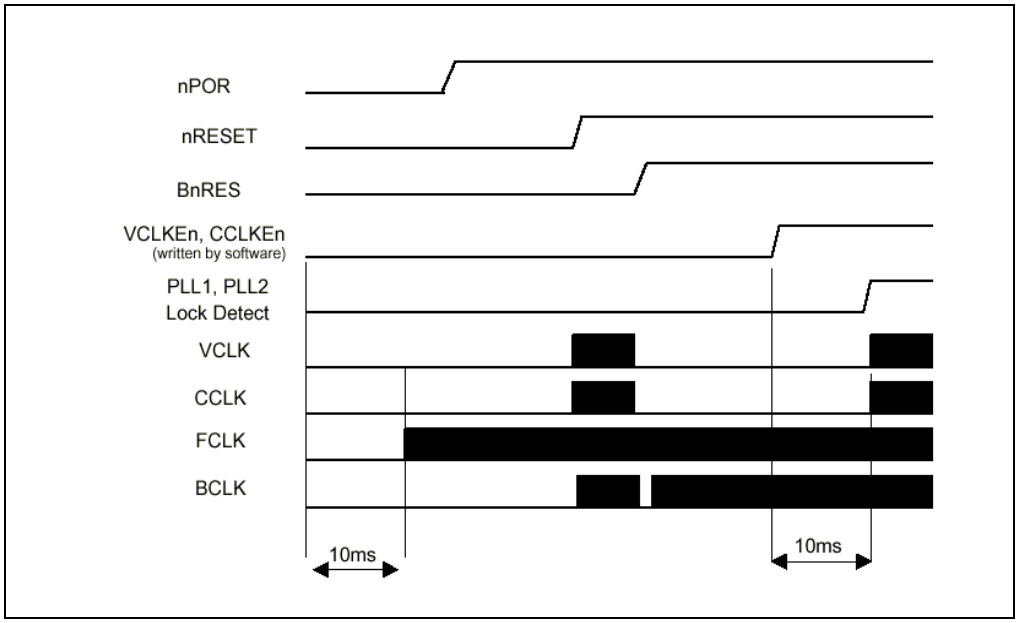


Figure 7-3: A Cold RESET event

In the event of removal and re-application of all power to the GMS30C7201, the following sequence may be typical:

- **nPOR** input is active. All internal registers are reset to their default values. The PMU drives **nRESETout** LOW to reset any off-chip peripheral devices.
- BnRES becomes active on exit from the **nPOR** condition. Clocks are enabled temporarily to allow synchronous resets to operate.
- The default frequency of FCLK on exit from **nPOR** will be 49.7664MHz.
- When FCLK is stable, the CPU clock is released. If the CPU were to read the RESET/ Status register at this time, it will return 0x10f:

Bit	Meaning
bit 0 set:	Power On Reset event has occurred
bit 1 set:	PLL1 has been ‘unlocked’
bit 2 set:	PLL2 has been ‘unlocked’
bit 3 set:	PLL3 has been ‘unlocked’

Table 7-13: Bit Settings for a Cold RESET Event within RESET STATUS register

- The CPU may write 0x10f to the RESET register to clear these flag bits.
- The CPU writes 0x20 to the clock control register, which will set a FCLK speed of 58.9824MHz. The new clock frequency, however, is not adopted until the PMU has entered and left DEEP SLEEP mode.
- The CPU sets a RTC timer alarm to expire in approximately 2 seconds
- The CPU sets DEEP SLEEP into the PMU Mode Register

- The PMU state machine will enter DEEP SLEEP mode (via the intermediate states shown in ***Figure 7-2: Power Management State Diagram*** on page 7-6).
- When the RTC timer alarm is activated, the PMU automatically wakes up into SLOW mode, but with the new FCLK frequency of 58.9824Mhz.
- The CPU may write 0x620 to the Clock Control register, which enables CCLK and VCLK, and retains the new FCLK frequency.

SOFTWARE GENERATED WARM RESET

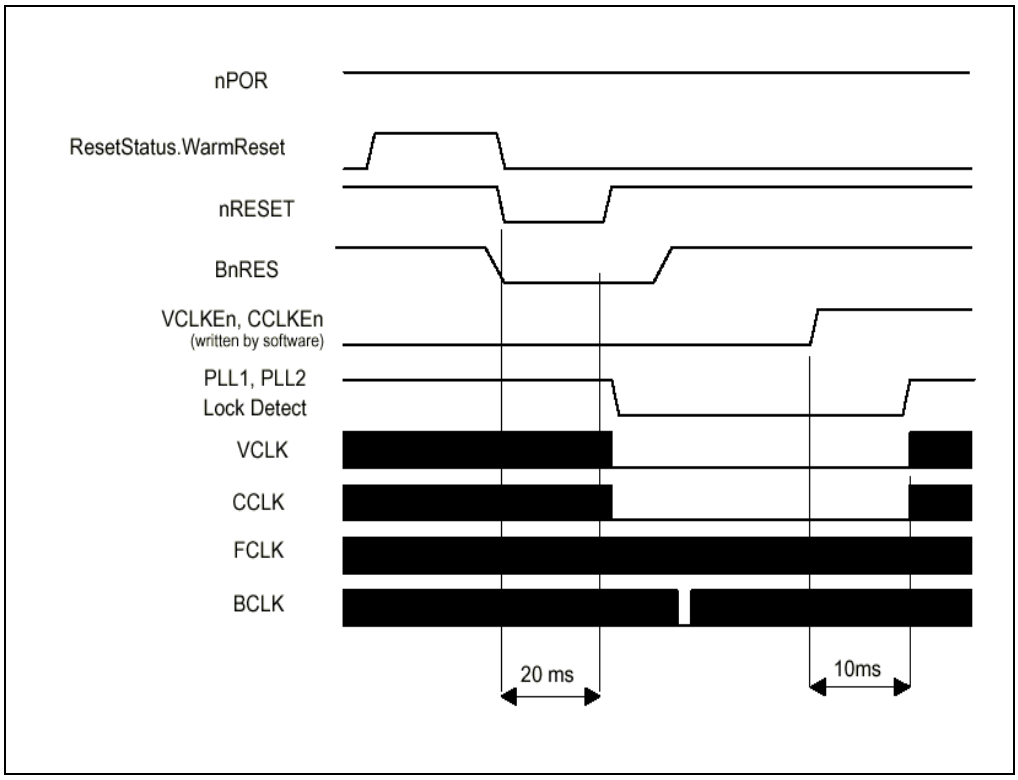


Figure 7-4: Software Generated Warm Reset

- The CPU writes '1' to the WarmReset bit of RESET/Status register. The PMU Drives **nRESET** low.The internal chip reset, **BnRES** is driven low.The PMU detects that the bidirectional **nRESET** pin is low. **nRESET** is filtered by a de-bounce circuit. Note that this means that **nRESET** will remain low for a minimum of 16ms. **BnRES** becomes active once the de-bounced **nRESET** goes high once more, which disables PLL1 and PLL2. The CPU may read the RESET register, which will return 0x106:

Bit	Meaning
bit 1 set:	PLL1 has been 'unlocked'
bit 2 set:	PLL2 has been 'unlocked'
bit 8 set:	A RESET event has occurred.

Table 7-14: Bit Settings for a Software Generated Warm RESET within RESET STATUS register

An Externally generated Warm RESET

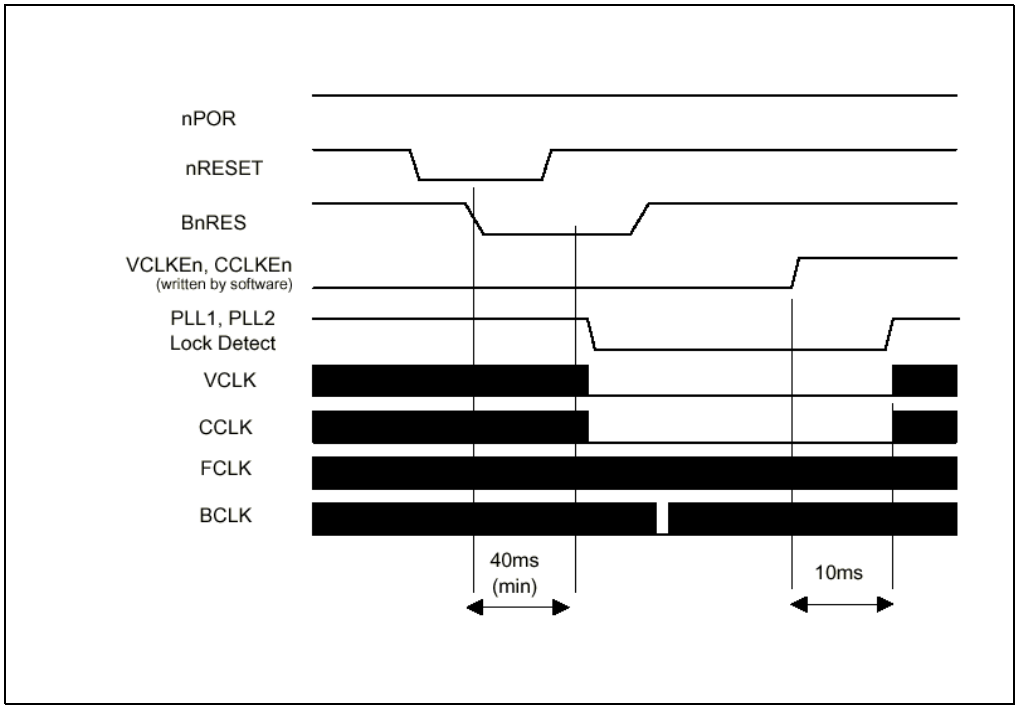


Figure 7-5: An Externally Generated Warm RESET

- **nRESET** is driven to '0' by external hardware. The **nRESET** input is filtered by a de-bounce circuit. Note that this means that **nRESET** must remain low for a minimum of 40ms. **BnRES** (the on-chip reset signal) becomes active as soon as **nRESET** is low, and high once the de-bounced **nRESET** goes high once more. **BnRES** disables PLL1 and PLL2. The CPU may read the RESET register, which will return 0x106:

Bit	Interpretation
bit 1 set:	PLL1 has been 'unlocked'
bit 2 set:	PLL2 has been 'unlocked'
bit 8 set:	A RESET event has occurred.

Table 7-15: Bit Settings for a Warm RESET within RESET STATUS register

Note *The internal chip reset, **BnRES,** remains active for 20ms after an externally generated **nRESET**. External devices should not assume that the GMS30C7201 is in an active state during this period.*

8 SDRAM Controller

SDRAM Controller

8.1 SDRAM Controller Specification

The system RAM resource is provided by SDRAM, on an interface that is run at the GMS30C7201's core clock frequency. Between 2 and 64Mbytes of external SDRAM are supported by one to four external devices. To reduce power consumption, each SDRAM device has its own Clock Enable (CKE), so each device may individually be placed in low power mode when idle. The SDRAMs are powered down into self-refresh mode when the whole system is placed in standby mode.

Internal to the GMS30C7201, the SDRAM controller arbitrates between access requests from the Main AMBA bus, and a custom Video bus.

The best use of an SDRAM is made when data is streamed in sequence, and future access requests can be predicted. It is in the nature of video data to be accessed in sequence at regular intervals; however, SDRAM accesses from the ARM are a lot less predictable. The SDRAM controller makes use of access predictability to maximize the use of memory interface bandwidth by having simultaneous access to both the LCD and VGA address buses. Video accesses to the SDRAM occur in fixed-burst lengths of 16 words, ARM and DMA controller accesses occur in a fixed-burst length of four words. If the requested accesses are shorter than four words, then the extra data is ignored.

8.2 Features

Clock Speed	60MHz	(0.35µm process)
	100MHz	(0.25µm process)
External Bus interface	16 bits wide (two accesses required for each word).	
Memory	2–64 Mbytes in up to four devices. The size of each memory device may be different. Programmable CAS delays of 1, 2 or 3. Supports SDRAMs organized with either two or four banks with page lengths of 256 or 512 half words. Note that the SDRAMs should be of the same speed grade.	
Programmable Auto Refresh Timer	Allows correct operation with large range of system clock frequencies.	

Video Resolution	Video Bandwidth Mbyte/sec	ARM Bandwidth (video active) Mbyte/sec	ARM Bandwidth (video inactive) Mbyte/sec	ARM Bandwidth (average) Mbyte/ sec
LCD 640 x 240	15	31.5	39.2	32.7
LCD 640 x 240 VGA 640 x 480	47	16	39.2	22.5
VGA 800 x 600	40	20	39.2	25.4

Table 8-1: 60MHz operation

Video Resolution	Video Bandwidth Mbyte/sec	ARM Bandwidth (video active) Mbyte/sec	ARM Bandwidth (video inactive) Mbyte/sec	ARM Bandwidth (average) Mbyte/ sec
LCD 640 x 240	15	48.8	57	50
LCD 640 x 240 VGA 640 x 480	47	23.6	57	32
VGA 800 x 600	40	29.2	57	37
LCD 640 x 480 VGA 640 x 480	64	12.5	57	25

Table 8-2: 100MHz operation

Note *The above timings represent the worst case timing. Best case timings are at least 25% better. Worst case timings assume that all accesses are from different pages in banks that are already open, and that no pipelining occurs between Video and ASB accesses.*

8.3 Supported Memory Devices

From 2–64Mbytes of SDRAM are supported with any mixture of up to four of 16-, 64- or 128Mbit devices. Each of the four external devices are mapped to a 16Mbyte boundary, and rely on the memory management unit to map different mixtures of devices (for example, 16- and 64Mbit devices) into a continuous address space for the ARM. Note that 16Mbit devices appear eight times, and the 64Mbit devices appear twice in the memory map.

Total Memory	16Mbit devices	64Mbit devices	128Mbit devices
2Mbyte	1	-	-
4Mbyte	2	-	-
8Mbyte	4	1	-
16Mbyte	-	2	1
32Mbyte	-	4	2
64Mbyte	-	-	4

Table 8-3: SDRAM upgrade path

Note The GMS30C7201 can use any mixture of 16-, 64- or 128Mbit SDRAMs. It is the responsibility of software to determine the actual external memory configuration, and to program the memory management unit appropriately.

The SDRAM controller allows up to four banks of memory to be open at once. The open banks may exist in different physical SDRAM devices.

8.4 SDRAM Control Registers

The SDRAM controller has three registers: the configuration, refresh timer and the Write Buffer Flush timer. The configuration register's main function is to specify the number of SDRAMs connected, and whether they are 2- or 4-bank devices. The refresh timer gives the number of **BCLK** ticks that need to be counted in-between each refresh period. The Write Buffer Flush timer is used to set the number of **BCLK** ticks since the last write operation, before the write buffer's contents are transferred to SDRAM.

Address	Name	Description
SDRAMCBase + 0x00	ConfigReg	#32-bit R/W
SDRAMCBase + 0x04	RefreshTmr	#16-bit R/W
SDRAMCBase + 0x08	Write buffer flush timer	#3-bit R/W

Table 8-4: SDRAM controller registers

In addition to the SDRAM control registers, the ARM may access the SDRAM mode registers by writing to a 64MByte address space referenced from the SDRAM mode register base address. Writing to the SDRAM mode registers is discussed further in ***8.5 Power-up Initialization of the SDRAMs*** on page 8-10.

8.4.1 Configuration Register

Address: SDRAM register base address

31							23								15								7							0
S_1	S_0	-	-	-	-	-	R	A	C_1	C_0	D	C	W	P	E_3	B_3	-	-	E_2	B_2	-	-	E_1	B_1	-	-	E_0	B_0	-	-

The SDRAM controller configuration register is a 32-bit wide split read/write register, such that bits [23:0] should be configured by the ARM, and bits [31:24] provide status information that are read-only.

All locations containing "-" are for future expansion, and should always be programmed with the binary value 0. Writes to bits [31:24] are always ignored.

$E_{[3:0]}$ Device Enable - indicates that there is a physical SDRAM present in each of the four slots in the address map. This bit is used to determine whether an auto-refresh command should be issued to a particular memory device.

slot 0 - address range 0–16Mbyte

slot 1 - address range 16–32MByte

slot 2 - address range 32–48MByte

slot 3 - address range 48–64MByte

Value = 1 if a device is present

Value = 0 if a device is not present

$B_{[3:0]}$ Indicates whether the SDRAM in the slot is a 2- or 4-bank device

Value = 1 if a four-bank device

Value = 0 if a two-bank device

$C_{[1:0]}$ CAS latency

$C_{[1]}$	$C_{[0]}$	CAS latency
0	0	Reserved
0	1	1
1	0	2
1	1	3

Table 8-5: CAS latency

P Set priority device for the Video bus
Value = 1 if the VGA is the priority device
Value = 0 if the LCD is the priority device

W Write buffer enable
Value = 1 if the write buffer is enabled
Value = 0 if the write buffer is disabled

R Normal SDRAM controller refresh enable
Value = 1 if the SDRAM controller provides refresh control
Value = 0 if the SDRAM controller does not provide refresh

$S_{[1:0]}$ SDRAM controller Status, read-only

$S_{[0]}$	$S_{[1]}$	Status
0	0	Idle
0	1	Busy
1	0	Self Refresh
1	1	Reserved

Table 8-6: SDRAM controller status

A Auto pre-charge on ASB accesses
A = 1 auto pre-charge (default)
A = 0 no auto pre-charge

D SDRAM bus tri-state control

D = 0 the controller drives the last data onto the SDRAM data bus (default)

D = 1 the SDRAM bus is tri-stated except during writes

This bit should be cleared before the IC is programmed into a low power mode. Driving the data lines avoids floating inputs which could increase device power consumption. During normal operation the D bit should be set, to avoid data bus drive conflicts with SDRAM.

C SDRAM clock enable control

C = 0 the clock enable of all IDLE devices are de-asserted to save power (default)

C = 1 all clock enables are driven HIGH continuously

During power-up initialization, it is important that the E[3:0] and the R bits are set in the correct sequence.

The SDRAM controller powers-up with E[3:0]=0000 and R=0. This indicates that the memory interface is IDLE. Next, the software should set at least one E bit to 1 with the R bit 0. This will cause all four devices to be precharged (if present). The next operation in the initialization sequence is to auto-refresh the SDRAMs. Note that the number of refresh operations required is device-dependent. Set the R bit to 1 and all the E bits to 0000 to start the auto-refresh process. Software will have to ensure that the prescribed number of refresh cycles are completed before moving on to the next step. The final step in the sequence is to set the R bit to 1 and to set the E bits corresponding to the populated slots. This will put the SDRAM controller (and the SDRAMs) in their normal operational mode.

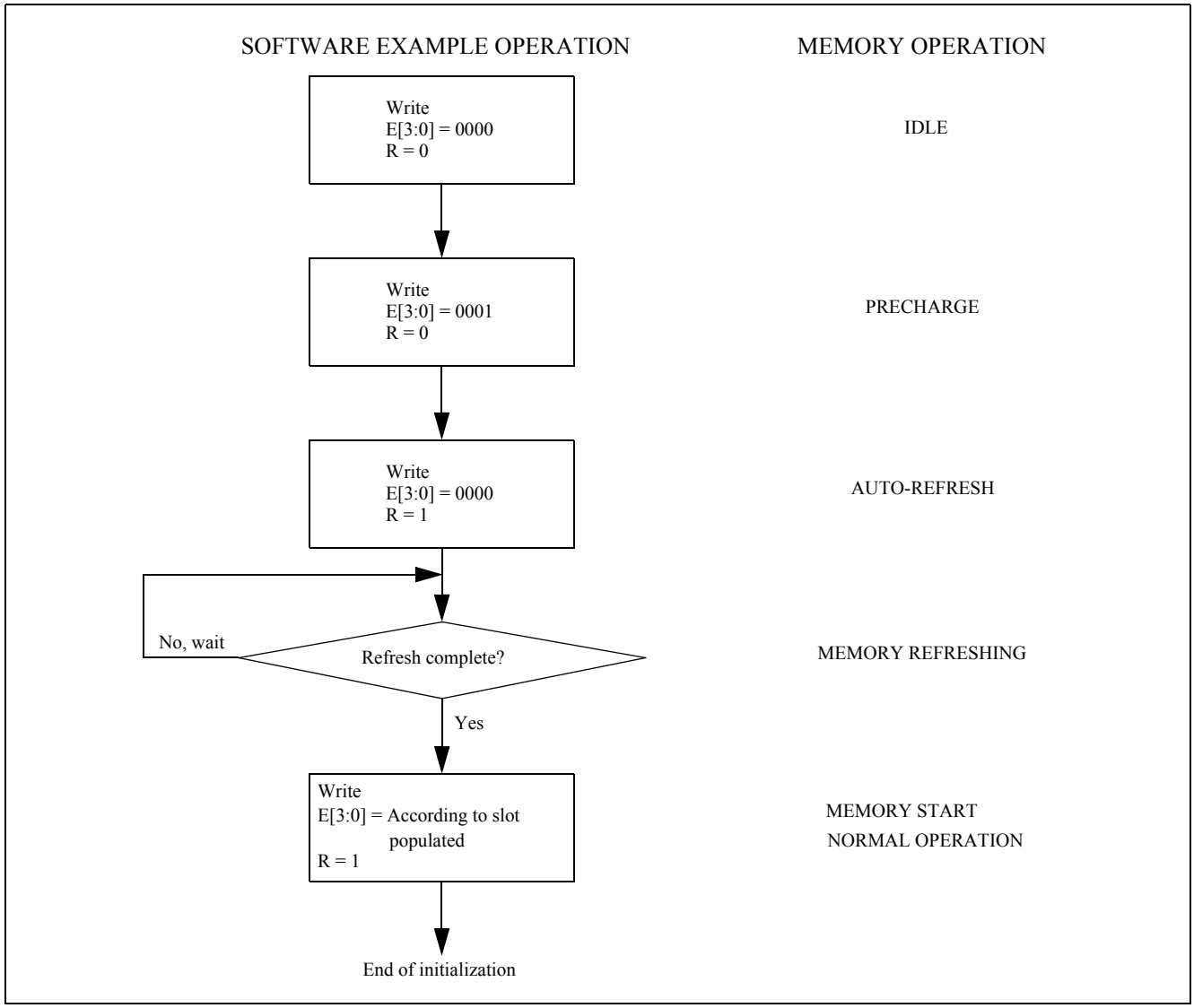


Figure 8-1: Software example/memory operation diagram

8.4.2 Refresh Timer

Address: SDRAM register base address + 4

A 16-bit read/write register that is programmed with the number of **BCLK** ticks that should be counted between SDRAM refresh cycles.

For example, for the common refresh period of 16µs, and a **BCLK** frequency of 50MHz, the following value should be programmed into it:

$16x10^{-6} * 50x10^{6} = 800$

The refresh timer defaults to a value of 128, which for a 16µs refresh period assumes a worst case (ie. slowest) clock rate of:

$$\frac{128}{16x10^{-6}} = 8 \text{ MHz}$$

The refresh register should be written to as early as possible in the system start-up procedure, and in the first few cycles if the system clock is less than 8MHz.

8.4.3 Write buffer flush timer

Address: SDRAM register base address + 8

A 3-bit read/write register that selects the time-out value for flushing the quad word merging write buffer. The times are given in the following table.

Timer value	BCLK ticks between time-outs
0	Time-out disabled
1	2
2	4
3	8
4	16
5	32
6	64
7	128

Table 8-7: Write buffer: flush time-out table

8.5 Power-up Initialization of the SDRAMs

The SDRAMs are initialized by applying power, waiting a prescribed amount of settling time (typically 100μs), performing some auto-refresh cycles (minimum 2) and then writing to the SDRAM mode register. The exact sequence is SDRAM device-dependent.

The settling time is referenced from when the SDRAM CLK starts. The processor should wait for the settling time before enabling the SDRAM controller refreshes, by setting the R bit in the SDRAM control register. The SDRAM controller automatically provides an auto refresh cycle for every refresh period programmed into the Refresh Timer when the R bit is set. The processor must wait for sufficient time to allow the manufacturer's specified number of auto-refresh cycles before writing to the SDRAM's mode register.

The SDRAM's mode register is written to via its address pins (A[13:0]). Hence, when the processor wishes to write to the mode register, it should read from the binary address (AMBA address bits [22:9]), which gives the binary pattern on A[13:0] which is to be written. The mode register of each of the SDRAMs may be written to by reading from a 64Mbyte address space from the SDRAM mode register base address. The correspondence between the AMBA address bits and the SDRAM address lines (A[13:0]) is given in the Row address mapping of ***Table 8-8: SDRAM row/column address map*** on page 8-11. Bits [25:24] of the AMBA address bus select the device to be initialized.

The SDRAM must be initialized to have the same CAS latency as is programmed into $C_{[1:0]}$ bits of the SDRAM control register, and always to have a burst length of 8.

8.6 SDRAM Memory Map

The SDRAM controller can interface with up to four SDRAMs. Three SDRAM sizes are supported—16, 64 and 128Mbits—which may be organized in either two or four banks but which must have a 16-bit data bus. A maximum of 64Mbytes of memory may be addressed by the SDRAM controller, which is subdivided into four 16Mbyte blocks, one for each of the external SDRAMs.

The mapping of the AMBA address bus to the SDRAM row and column addresses is given in ***Table 8-8: SDRAM row/column address map***. The first row of the diagram indicates the SDRAM address bit (A[13:0]); the remaining numbers indicate the AMBA address bits MBA[23:1]. Note that for 16Mbit devices, pins A[11,9] on the SDRAM should be connected to pins A[13,12] on the GMS30C7201, and the pins A[11,9] should not be connected.

SDRAM ADDR	13 (BS0)	12 (BS1)	11	10	9	8	7	6	5	4	3	2	1	0
Row 16Mbit Device	10*	9*	Note 1	20*	Note 1	19*	18*	17*	16*	15*	14*	13*	12*	11*
Col 16Mbit Device	10	9	Note 1	20	Note 1	23	8*	7*	6*	5*	4*	3*	2*	Note 2
Row 64Mbit Device	10*	9*	22*	20*	21*	19*	18*	17*	16*	15*	14*	13*	12*	11*
Col 64Mbit Device	10	9	22	20	21	23	8*	7*	6*	5*	4*	3*	2*	Note 2
Row 128Mbit Device	10*	9*	22*	20*	21*	19*	18*	17*	16*	15*	14*	13*	12*	11*
Col 128Mbit Device	10	9	22	20	21	23*	8*	7*	6*	5*	4*	3*	2*	Note 2
Mode Write	10*	9*	22*	20*	21*	19*	18*	17*	16*	15*	14*	13*	12*	Note 2
Summary	10	9	22	20	21	19/23	18/8	17/7	16/6	15/5	14/4	13/3	12/2	11

Table 8-8: SDRAM row/column address map

Notes *(1) For the 16Mbit device, SDRAM address line A11 should be connected to the GMS30C7201 pin SA[13](BS0), and the SDRAM address line A9 should be connected to the GMS30C7201 pin SA[12](BS1). The GMS30C7201 address lines A11 and A9 should not be connected.*

(2) Since all burst accesses commence on a word boundary, and SDRAM addresses are non-incrementing (the address incrementer is internal to the device), column address zero will always be driven to logic '0'.

** An asterisk denotes the address lines that are used by the SDRAM.*

The start addresses of each SDRAM is fixed to a 16Mbyte boundary. The memory management unit will be used to map the actual banks that exist into contiguous memory as seen by the ARM. Bits [25:24] of the AMBA address bus select the device to be initialized, as described in ***Table 8-9: SDRAM device selection*** on page 8-12.

A25	A24	Device selected
0	0	Device 0
0	1	Device 1
1	0	Device 2
1	1	Device 3

Table 8-9: SDRAM device selection

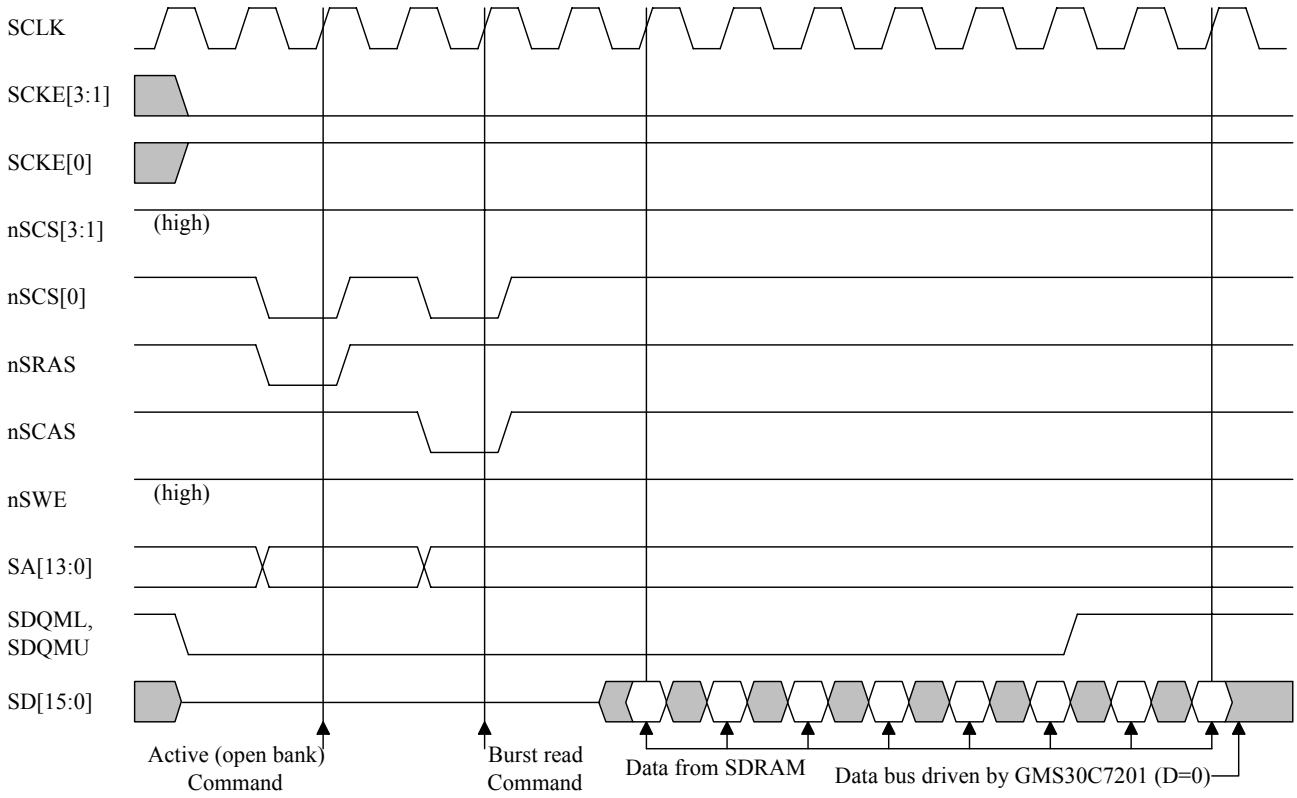


Figure 8-2: Example SDRAM burst read

Note If the D bit is cleared (in the SDRAM configuration register) the GM30C7201 starts driving the data bus immediately after the last read cycle ends. See section 8.4.1

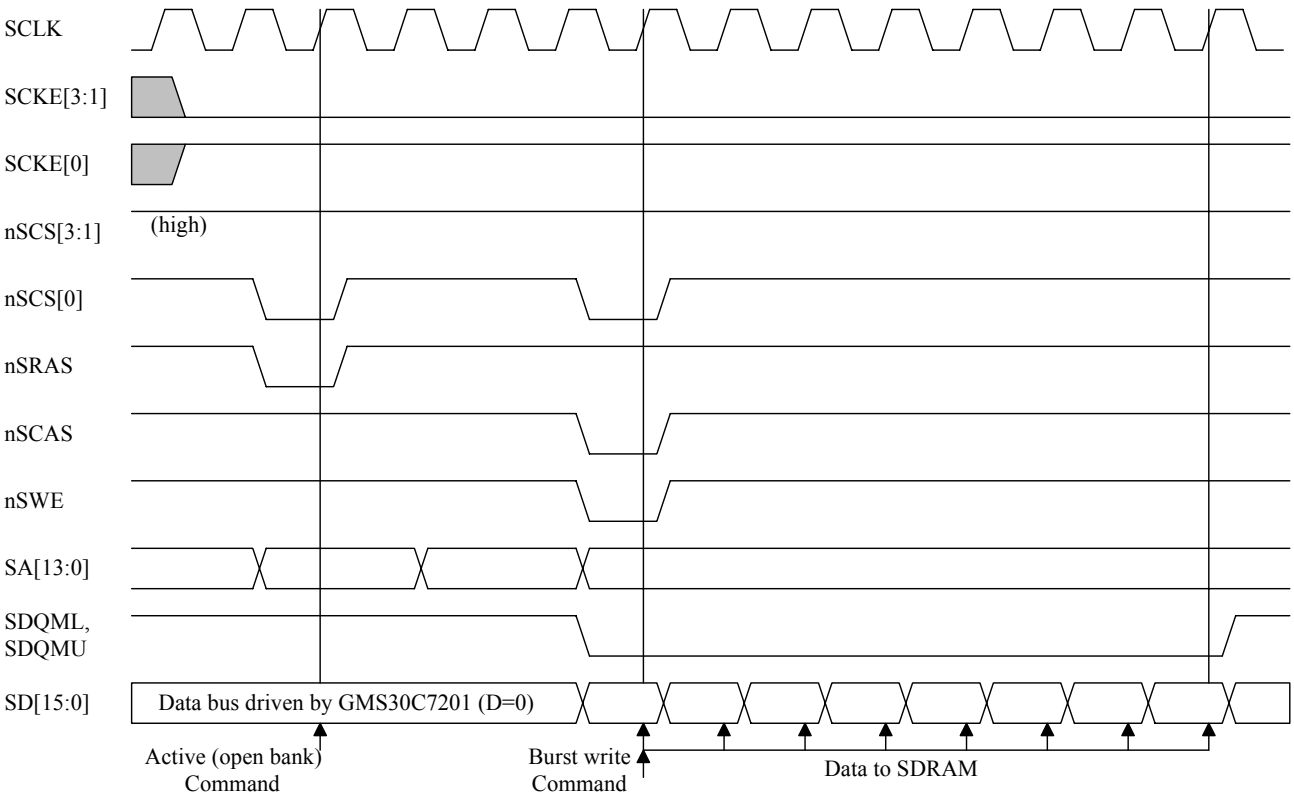


Figure 8-3: Example SDRAM burst write

Note If the D bit is cleared (in the SDRAM configuration register) the GM30C7201 drives the data lines before and after the write burst sequence. See section 8.4.1

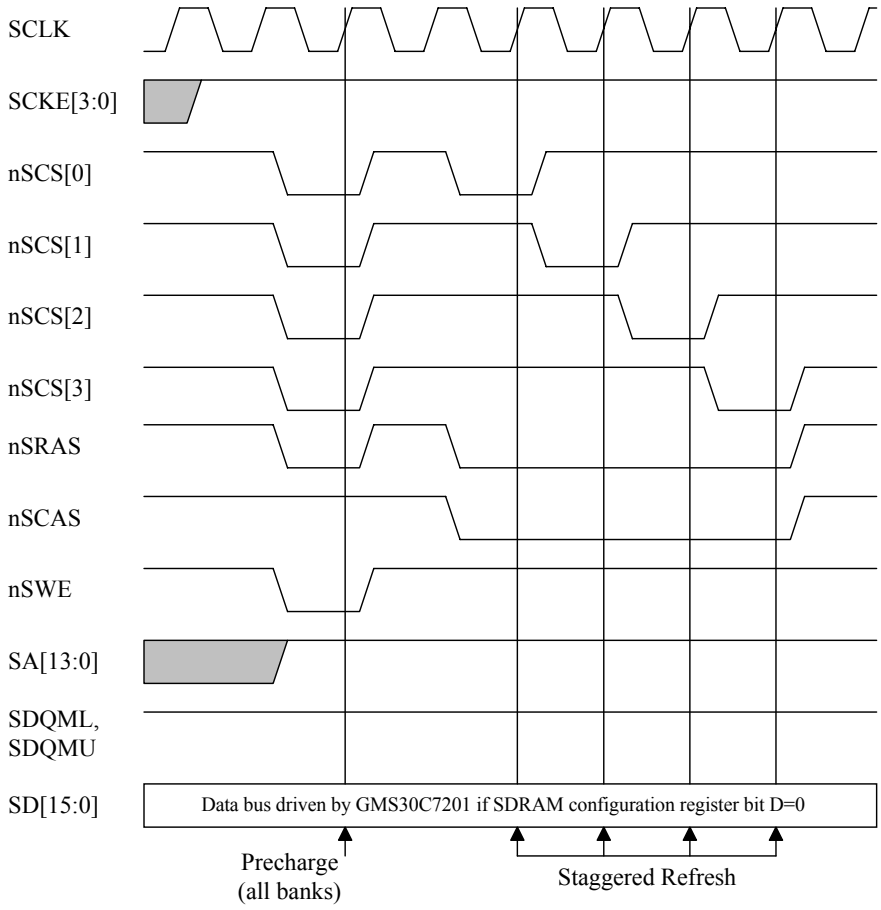


Figure 8-4: Example SDRAM refresh, all banks enabled

Note If the D bit is cleared (in the SDRAM configuration register) the GM30C7201 drives the data lines to avoid floating inputs. See section 8.4.1

8.7 AMBA Accesses and Arbitration

The SDRAM controller bridges both the AMBA Main and Video buses. On the Main bus, the SDRAM appears as a normal slave device. On the Video DMA bus, the SDRAM controller integrates the functions of the bus arbiter and address decoder.

Writes from the main bus may be merged in the quad word merging write buffer. See ***8.8 Merging Write Buffer*** on page 8-17 for more information on this feature.

Access requests from either the Main or Video buses are arbitrated by a Main/Video arbiter according to the following sequence:

Highest Priority:	LCD/VGA (modified round robin)
	Refresh request
Lowest Priority:	Main bus peripheral (PMU, ARM, Piccolo, DMA)—order determined by Main bus arbiter.

The LCD and VGA have nominally the same priority, and are arbitrated using a “biased round robin” algorithm. The “biased round robin” algorithm in this context means that if both of the video peripherals on the video bus request access at the same time, the video system indicated by the P bit in the control register gets first access, then the video system not selected by the P bit is granted access, regardless of whether the system selected by the P bit is still requesting access. As an example, if the P bit is 1, the VGA is given priority, so if both VGA and LCD request access at the same time, VGA is granted first, then LCD, even if VGA is still requesting after its first access.

LCD and VGA SDRAM accesses always occur in bursts of 16 words. Once a burst has started, data is presented by the SDRAM controller without wait states. The LCD and VGA may only read data from SDRAM, no write path is supported.

If a refresh cycle is requested, then it will have lower priority than either the VGA or LCD, but will be higher than any other accesses from the Main bus. Assuming a worst case **BCLK** frequency of 8MHz, the maximum, worst case latency that the arbitration scheme enforces is 11.5µs before a refresh cycle can take place. This is comfortably within the 16µs limit. Note that the four external SDRAM devices are refreshed on four consecutive clock cycles to reduce the peak current demand on the power source.

The arbitration of the Main bus is left to the Main bus arbiter.

Data transfers requested from the Main bus always occur as a burst of eight half-word accesses to SDRAM. Access requests from the Main bus cannot be broken into by the Main bus arbiter.

In the case where fewer than four words are actually requested by the Main bus peripheral, the excess data from the SDRAM is ignored by the SDRAM controller in the case of read operations, or masked in the case of writes.

In the case where more than four words are actually requested by the Main bus peripheral, the SDRAM controller asserts **MBLAST** to force the ASB decoder to break the burst.

In the case of word misalignment to a quad word boundary (when any of address bits [3:0] are non-zero at the start of the transfer), **MBLAST** is asserted at the next quad word boundary (bits 2 and 3 set) to force the ASB decoder to break the burst.

Sequential half word (or byte) reads are not supported. Any burst requests for byte or half word reads are broken by the SDRAM controller asserting **BLAST**.

This is not an issue, since although the GMS30C7201 includes the THUMB processor, which can generate sequential half word accesses, if the cache is enabled, these will become quad word cache line fills.

In the case of quad word misalignment, byte or half word access requests from the Main AMBA bus, the data requested by address bits MBA[3:2] is accessed from the SDRAM first.

In the case of byte or half word reads, data is replicated across the whole of the ASB data bus.

Data bus for word access:

31								23								15								7							0
d_{31}	d_{30}	d_{29}	d_{28}	d_{27}	d_{26}	d_{25}	d_{24}	d_{23}	d_{22}	d_{21}	d_{20}	d_{19}	d_{18}	d_{17}	d_{16}	d_{15}	d_{14}	d_{13}	d_{12}	d_{11}	d_{10}	d_9	d_8	d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

Data bus for half word access:

31								23								15								7							0
d_{15}	d_{14}	d_{13}	d_{12}	d_{11}	d_{10}	d_9	d_8	d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	d_{15}	d_{14}	d_{13}	d_{12}	d_{11}	d_{10}	d_9	d_8	d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

Data bus for byte access:

31								23								15								7							0
d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

8.8 Merging Write Buffer

An eight word merging write buffer is implemented in the SDRAM controller to improve write performance. The write buffer can be disabled, but its operation is completely transparent to the programmer. The eight words of the buffer are split into two quad words, the same size as all data transactions to the SDRAMs. The split into two quad words allows one quad word to be written to at the same time as the contents of the other are being transferred to SDRAM. The quad word buffer currently being written to may be accessed with non-contiguous word, half word or byte writes, which will be merged into a single quad word. The buffered quad word will be transferred to the SDRAM when:

- there is a write to an SDRAM address outside the current quad word being merged into
- there is a read to the address of the quad word being merged into
- there is a time-out on the write back timer

The two quad-words that make up the write buffer operate in “ping-pong” fashion, whereby one is initially designated the buffer for writes to go into, and the other is the buffer for write backs. When one of the three events that can cause a write-back occurs, the functions of the two buffers are swapped. Thus the buffer containing data to be written back becomes the buffer that is currently writing back, and the buffer that was the write-back buffer becomes the buffer being written to.

In the case of a write-back initiated by a read from the same address as the data in the merge buffer, the quad word in the buffer is written to SDRAM, and then the read occurs from SDRAM. The write before read is essential, because not all of the quad word in the buffer may have been updated, so its contents need to be merged with the SDRAM contents to fill any gaps where the buffer was not updated.

The write buffer flush timer forces a write back to occur after a programmable amount of time. Every time a write into the buffer occurs, the counter is re-loaded with the programmed time-out value, and starts to counts down. If a time-out occurs, then data in the write buffer is written to SDRAM.

9 Static Memory Interface

Static Memory Interface

9.1 Overview

The Static Memory Controller interfaces the AMBA Advanced System Bus (ASB) to the External Bus Interface (EBI). Six separate memory or expansion banks are provided by this block. Each bank is 64MB in size and can be programmed individually to support:

- 8-, 16- or 32-bit wide, little-endian memory
- burst mode read access support
- variable wait states (up to 16)

In addition, bus transfers can be extended using the **EXPRDY** input signal. Burst mode access allows fast sequential access within quad word boundaries. This can significantly improve bus bandwidth in reading from memory (that must support at least four word burst reads).

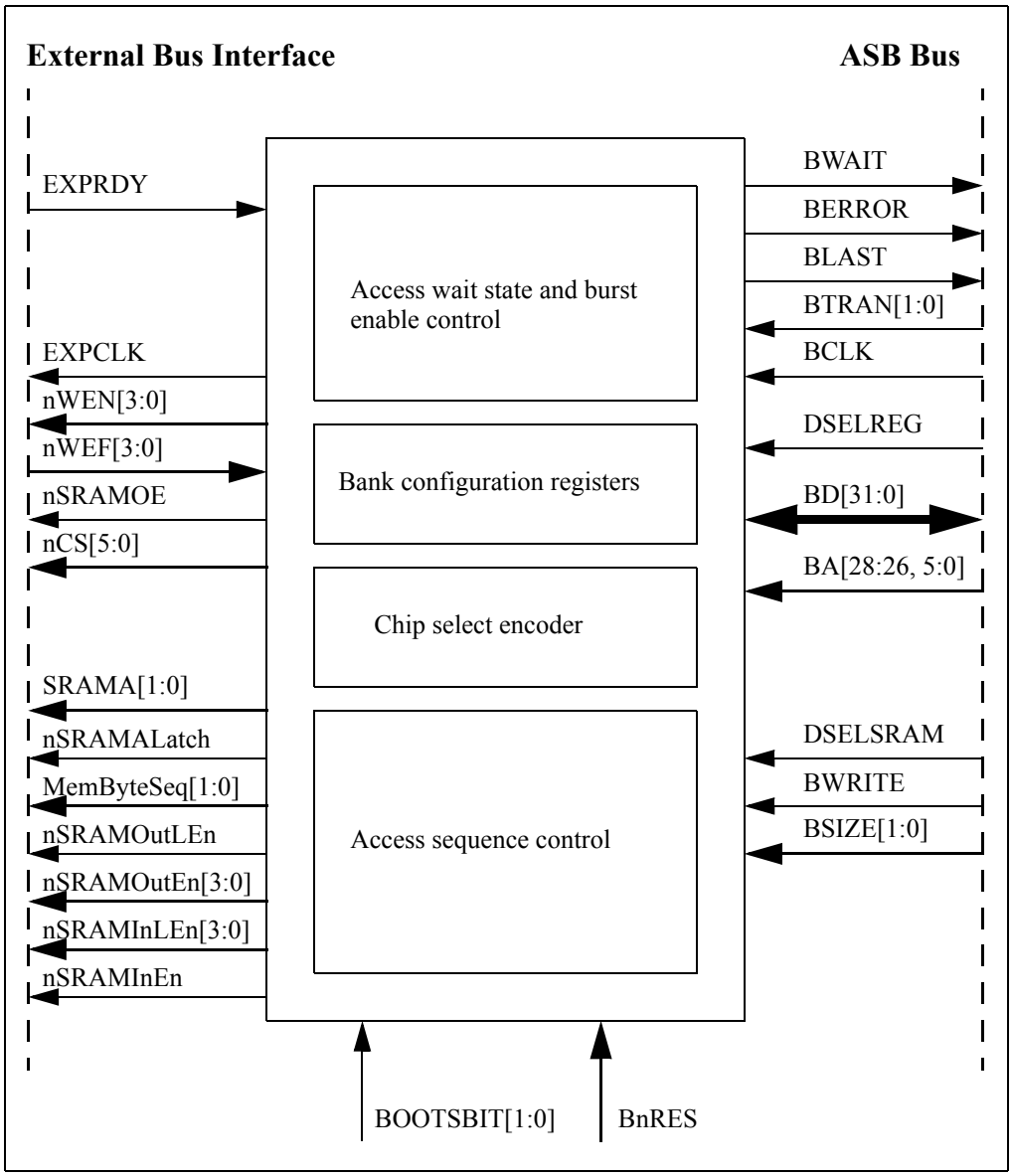


Figure 9-1: Static Memory Controller block diagram

9.2 Hardware Interface and Signal Description

The Static Memory Controller module is connected to the ASB bus. ***Table 9-1: Static Memory Controller ASB signal descriptions*** shows the internal bus interface signals to the Static Memory Controller.

Name	Type	Description
BA[28:26, 5:0]	In	System address bus.
BD[10:0]	InOut	Bidirectional system data bus.
BCLK	In	The ASB system clock
BnRES	In	AMBA asynchronous reset. This signal is negative active.
BWAIT	Out	This slave response signal is driven when the BUSC is selected, and is used to indicate if the memory has completed its current transfer.
BERROR	Out	Slave response signal.
BLAST	Out	Slave response signal.
BSIZE[1:0]	In	The signals indicate the size of the transfer, which may be byte, halfword or word.
BTRAN[1:0]	In	These signals are used to determine access type.
BWRITE	In	When LOW, Read; when HIGH, Write.
DSELSRAM	In	When HIGH, this signal indicates that the Bus controller is selected.
DSELREG	In	When HIGH, this signal indicates that one of the bank configuration registers is selected.
EXPRDY	In	Expansion channel ready. When LOW, during phase one this signal will force the current memory transfer to be extended.
nWEN[3:0]	Out	These signals are active LOW write enables for each of the memory byte lanes on the external bus.
nWEF[3:0]	In	These optional connections use PADs feedback from the external side of the nWEN[3:0] PADs. They are used to guarantee address and chip select hold time when any write enable is LOW.
nSRAMOE	Out	This is the active LOW output enable for devices on the external bus.
nCS[5:0]	Out	Active LOW chip selects.
SRAMA[1:0]	Out	These signals form the lower bits of the external address bus. They are used to control accesses to 16- or 8-bit memories when the bus requests an access size larger than the memory.
nSRAMALatch	Out	When LOW, transparent address latch enable.
BMemByteSeq[1:0]	Out	These signals control byte sequencing for the Data In and Data Out paths.

Table 9-1: Static Memory Controller ASB signal descriptions

Name	Type	Description
nSRAMOutEn[3:0]	Out	Active LOW byte lane data output driver enables.
nSRAMOutLEn	Out	When LOW, the data path output latch is transparent.
nSRAMInLEn[3:0]	Out	Each signal controls a byte latch in the Data In path.
nSRAMInEn	Out	When LOW BD[31:0] is driven by the EBI.
BOOTSBIT[1:0]	In	Configuration input. 00 - Select bank 0 as 32-bit memory 01 - Select bank 0 as 16-bit memory 10 - Select bank 0 as 8-bit memory 11 - Reserved

Table 9-1: Static Memory Controller ASB signal descriptions (Continued)

9.3 Functional Description

The Static Memory Controller has six main functions:

- memory bank select
- access sequencing
- wait states generation
- burst read control
- byte lane write control

These are described below.

9.3.1 Memory bank select

The chip select signal generation is controlled by **BA[28:26]**, **DSEL** and **BWRITE**. Refer to ***Table 9-2: Static memory bank select coding***.

DSEL	BA[28:26]	nCS[5:0]	Memory Configuration
0	XXX	111111	not selected
1	000	111110	nCS0 configuration
1	001	111101	nCS1 configuration
1	010	111011	nCS2 configuration
1	011	110111	nCS3 configuration
1	100	101111	nCS4 configuration
1	101	011111	nCS5 configuration

Table 9-2: Static memory bank select coding

9.3.2 Access sequencing

Bank configuration also determines the width of the external memory devices. When the external memory bus is narrower than the transfer initiated from the current master, the internal transfer will take several external bus transfers to complete.

9.3.3 Wait states generation

The Static Memory Controller supports wait states for read and write accesses. This is configurable between one and 16 wait states for standard memory access, and zero and 15 wait states for burst mode.

The Static Memory Controller also allows transfers to be extended indefinitely. This is done by asserting **EXPRDY** LOW. To hold the current transfer, **EXPRDY** must be asserted on the falling edge of **BCLK** before the last cycle of the accesses. The transfer cannot complete until **EXPRDY** is HIGH for at least one cycle.

9.3.4 Burst read control

This supports sequential access burst reads of up to four consecutive locations in 8-, 16- or 32-bit memories.

9.3.5 Byte lane write control

This controls **nWEN[3:0]** according to transfer width, **BA[1:0]** and the access sequencing. ***Table 9-3: nWEN coding*** shows the basic coding assuming 32-bit external memory.

BSIZE[1:0]	BA[1:0]	nWEN[3:0]
10 (word)	XX	0000
01 (half word)	1X	0011
01 (half word)	0X	1100
00 (byte)	11	0111
00 (byte)	10	1011
00 (byte)	01	1101
00 (byte)	00	1110

Table 9-3: nWEN coding

9.4 Register Description

9.4.1 Register map

Address	Name	Description
BUSCbase + 00	MEMCFGR0	Memory Configuration Register 0
BUSCbase + 04	MEMCFGR1	Memory Configuration Register 1
BUSCbase + 08	MEMCFGR2	Memory Configuration Register 2
BUSCbase + 0C	MEMCFGR3	Memory Configuration Register 3
BUSCbase + 10	MEMCFGR4	Memory Configuration Register 4
BUSCbase + 14	MEMCFGR5	Memory Configuration Register 5
BUSCbase + 18	Testreg0 (R/W)	0 Test mode bit 1 Test reset bit 3–2 External memory width select bit 31–4 Reserved
BUSCbase + 1C	Testreg1 (R)	0 nSRAMOE 4–1 nWEN[3:0] 10–5 nCS[5:0] 31–11 Reserved
BUSCbase + 20	Testreg2 (R)	1–0 MemByteseq[1:0] 2 SRAMALatch 10–3 SRAMA 31–11 Reserved
BUSCbase + 24	Testreg3 (R)	0 nSRAMOutLEn 4–1 nSRAMOutEn[3:0] 8–5 nSRAMInLEn[3:0] 9 nSRAMInEn 31–10 Reserved

Table 9-4: Static Memory Controller register map

9.4.2 Configuration register format

Each of the 8-bit fields in the memory configuration registers defines:

- the number of wait states
- the bus width
- whether **EXPCLK** is enabled during accesses
- whether the bank is connected to a burst mode ROM

This is shown in ***Figure 9-2: Byte fields in the memory configuration register***.

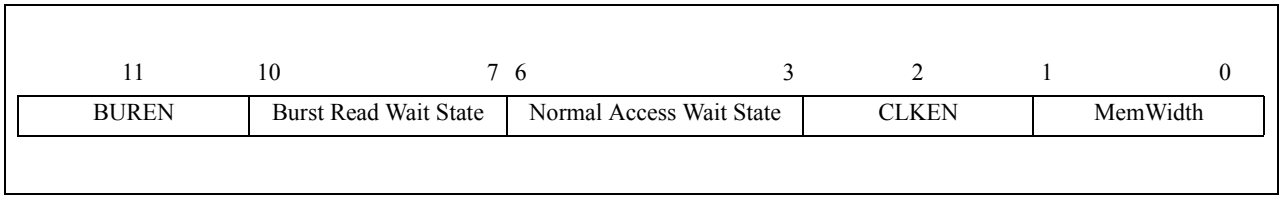


Figure 9-2: Byte fields in the memory configuration register

Table 9-5: Values of the memory width field defines the memory width.

Memory width	Description
00	32-bit memory access
01	16-bit memory access
10	8-bit memory access
11	Reserved

Table 9-5: Values of the memory width field

CLKEN — Expansion clock enable. Setting this bit enables the **EXPCLK** to be active during accesses to the specified bank. This provides a timing reference for devices that need to extend bus cycles using the **EXPRDY** input. Back-to-back sequential accesses result in a continuous clock.

Table 9-6: Values of the normal access wait states field defines the values of the Normal Access Wait States field.

Value	Number of Wait States
0000	16
0001	15
0010	14
0011	13
0100	12
0101	11
0110	10
0111	9
1000	8
1001	7
1010	6
1011	5
1100	4
1101	3
1110	2
1111	1

Table 9-6: Values of the normal access wait states field

Table 9-7: Values of the burst read wait states field defines the values of the Burst Read Wait States field.

Value	Number of Wait States
0000	15
0001	14
0010	13
0011	12
0100	11
0101	10
0110	9
0111	8
1000	7
1001	6
1010	5
1011	4
1100	3
1101	2
1110	1
1111	0

Table 9-7: Values of the burst read wait states field

BUREN — Burst enable. Setting this bit enables burst reads to take advantage of faster access times from memory devices that support burst mode.

10 PCMCIA Interface

This chapter describes the PCMCIA Interface.

PCMCIA Interface

10.1 Overview

This Adapter interfaces between the internal main system bus (AMBA - ASB) and the external PCMCIA card bus as shown in ***Figure 10-1: Overview of External Memory and Static Memory Control*** on page 10-3

Features

- Supporting two PCMCIA2.1/JEIDA4.2 compliant 68 pin card slots
- PCMCIA I/F directly connected to AMBA - ASB
- 3.3V & 5V Mixed Voltage Operation for each PCMCIA Card
- Support for Low Voltage PCMCIA Card including 3.3 V CIS Reading
- Reduced pin assignment using external buffers for Data Access
- Address and Data bus is shared by using External Bus Interface
- Enhanced Power management
- Direct memory mapping.
- 256MB address space per each slot
  - Attribute Memory: 64M,
  - Common Memory: 64M,
  - I/O: 64M,
  - Reserved: 64M
- Support for ATA PCMCIA Card (Including Flash Memory Card)
- Support for Multi-function I/O and Mixed Memory and I/O Card
- DMA operations to PCMCIA Card not supported
- Advanced Test Interface using TIC mode

Note IOIS16 and INPACK signals are not supported in this Spec.

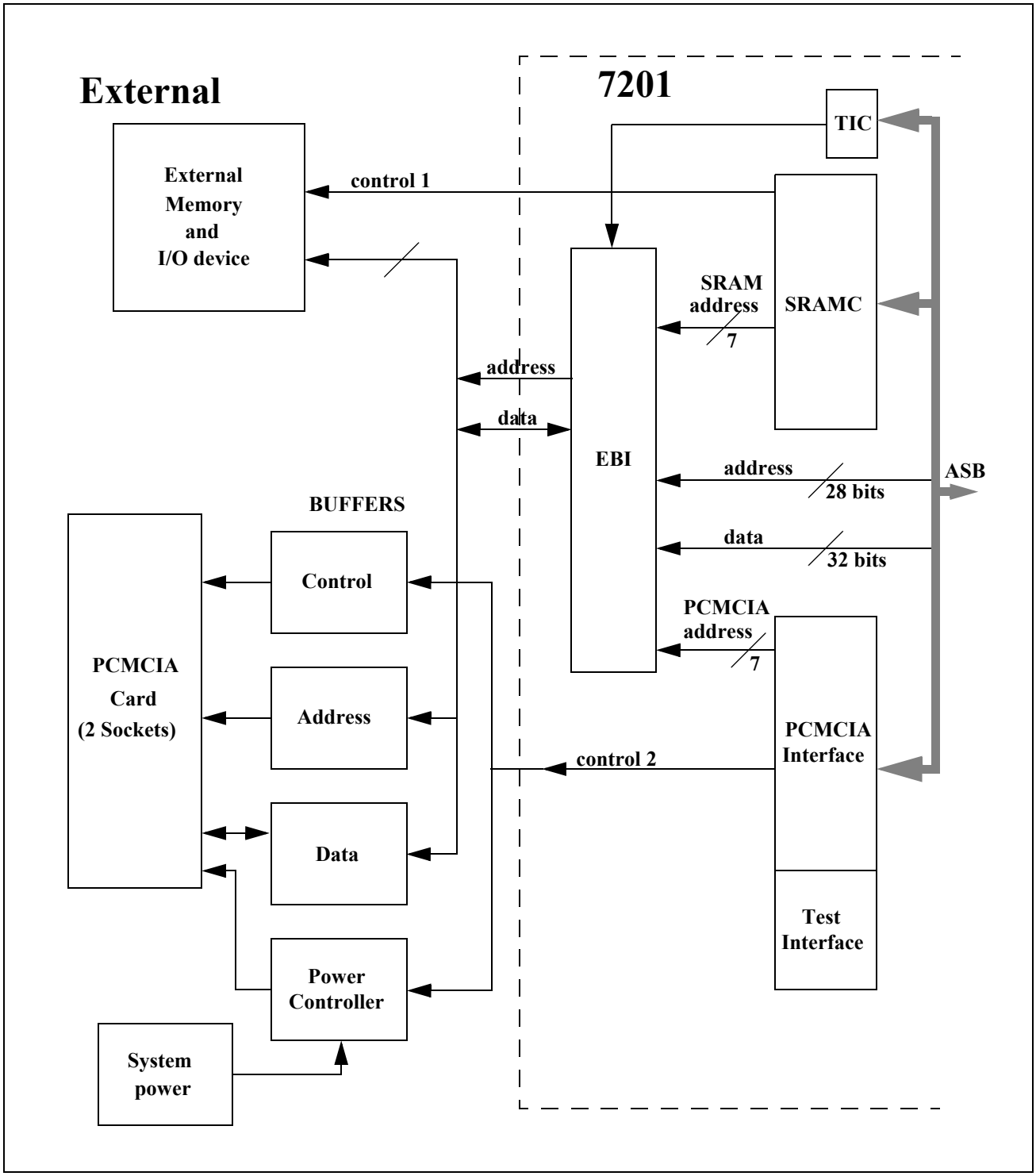


Figure 10-1: Overview of External Memory and Static Memory Control

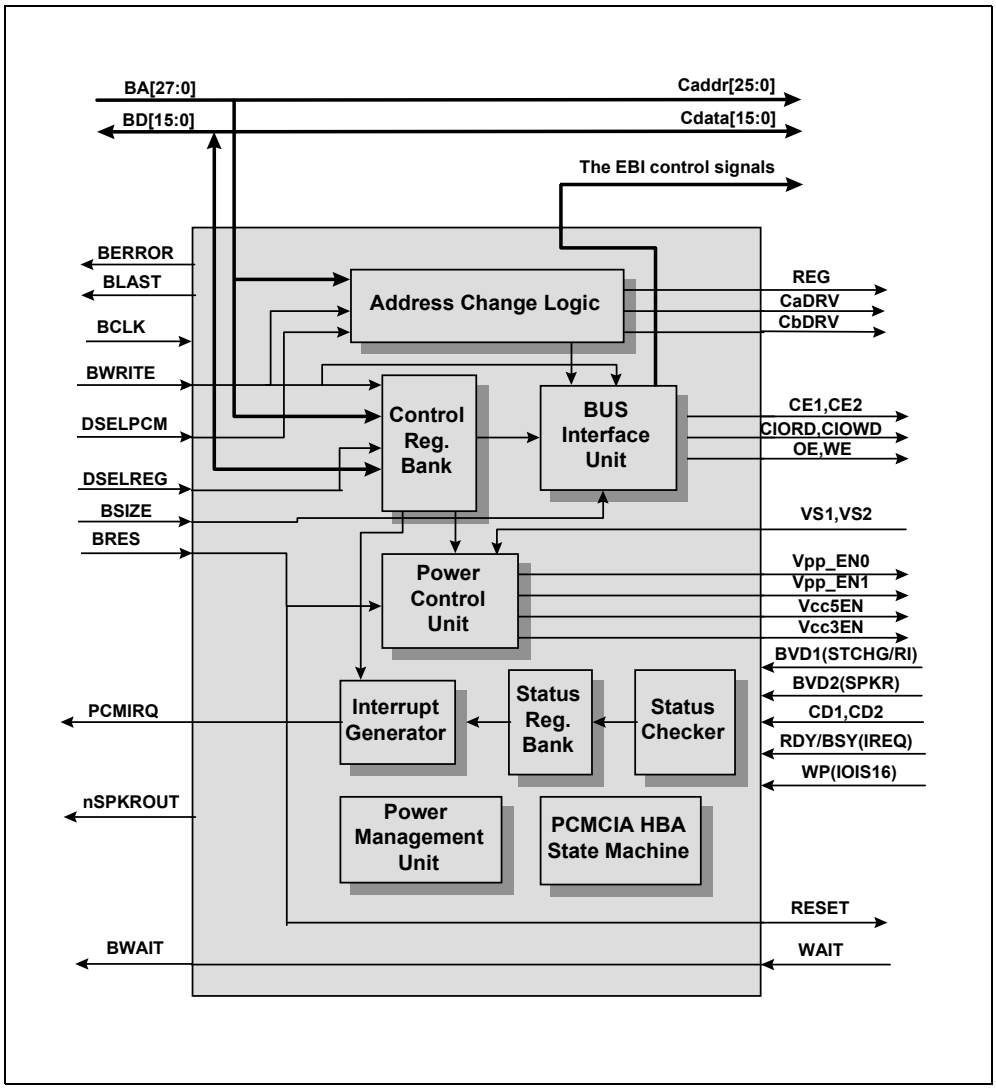


Figure 10-2: Block Diagram

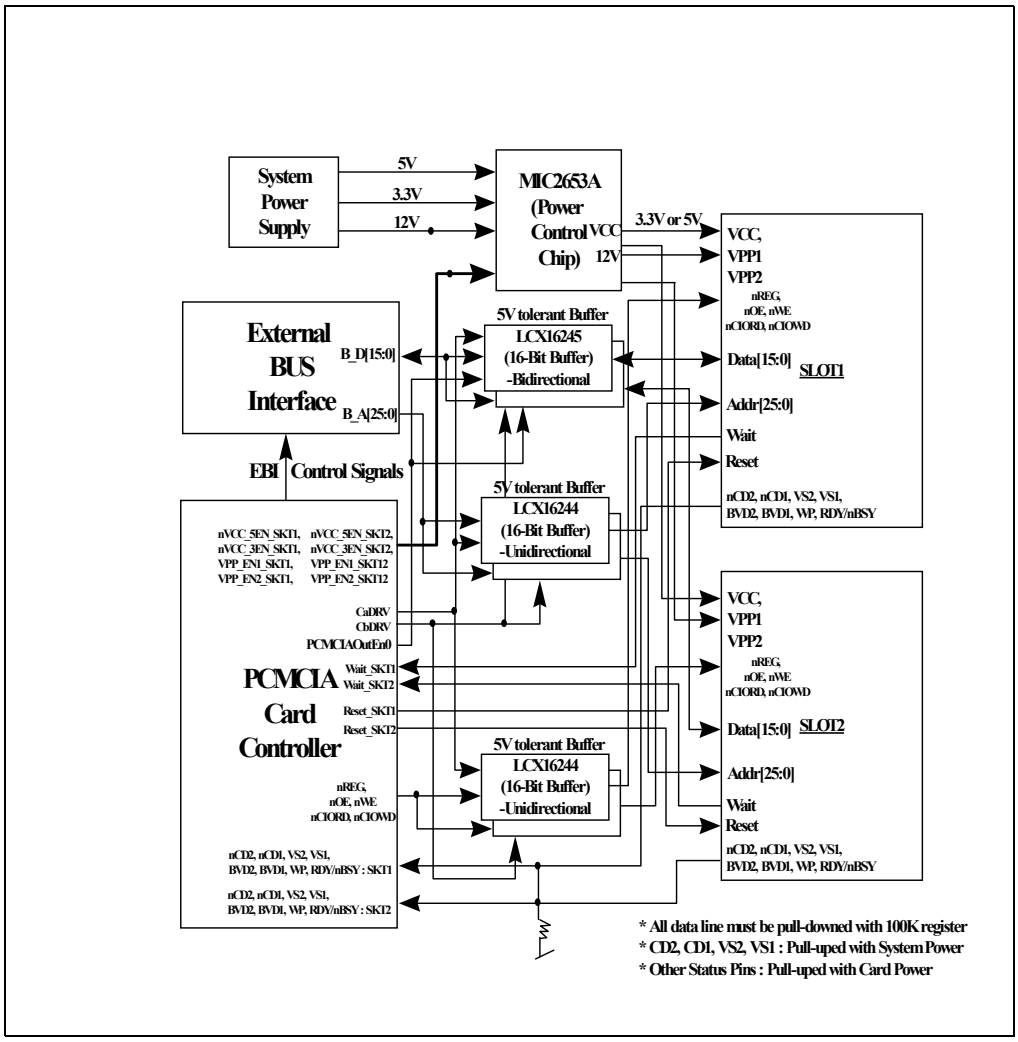


Figure 10-3: External Circuit Diagram

This diagram describes the external floorplan for PCMCIA and the interconnection from the view of system level. This diagram also describes the pin connection with the socket and the controller.

10.1.1 Hardware interface and signal description

10.1.2 Host Bus interface signal

The signals listed below go to the PCMCIA Interface on the ASB side.

Name	Type	Source	Description
BCLK	Input	Clock Controller	System Bus Clock This Clock times all bus transfers.
BA[28:21]	Input	ASB	System Bus Address

Table 10-1: Host Bus Interface Signals

Name	Type	Source	Description
BA[6:2]	InOut	ASB	Upper three bits are required to address translation and lower four bits to configure control register
BA[0]	Input	ASB	Upper three-bits are required to address translation and lower four bits to configure control register
BD[7:0]	Input	ASB	System Data Bus These bits are used to configure the control register. All configuration registers are eight bits wide.
nBRES	Input	ASB	This signal indicates the reset status of the AMBA ASB
BWRITE	Input	ASB	When HIGH, this signal indicates a write transfer and when LOW, a read.This signal has the same timing as the address bus
BERROR	Output	ASB	This slave response is driven HIGH during phase one of **BCLK** when a transfer error has occurred. When LOW the transfer is successful
BLAST	Output	ASB	This slave response is driven HIGH during phase one of **BCLK** when PCMCIA Host Bus Adapter is selected.
BWAIT	Output	ASB	This slave response is driven HIGH during phase one of **BCLK** when PCMCIA Host Bus Adapter is selected and is used to indicate if the PC Card has completed its current transfer. The signal is driven low at the end of the cycle.
BSIZE[1:0]	Input	ASB	This signal indicates the size of the transfer which may be byte, half-word or word. When HIGH, this signal indicates that one of the configuration registers is selected.
DSELPCM	Input	AMBA Decoder	This signal changes in phase two of **BCLK.** When HIGH, this signal indicates that one of the configuration registers is selected.
DSELREG	Input	AMBA Decoder	This signal changes in phase two of **BCLK.**

Table 10-1: Host Bus Interface Signals (Continued)

10.1.3 PCMCIA card slot interface

The signals listed below connect between the Interface and the PCMCIA card.

Name	Type	Description
***nCE2**	Output	Card data bus HIGH byte enable
***nCE1**	Output	Card data bus LOW byte enable
***nCIORD**	Output	Card I/O read This signal is valid for I/O PCMCIA cards
***nCIOWD**	Output	Card I/O write This signal is valid for I/O PCMCIA Card.
***nOE**	Output	Output Enable for Memory PCMCIA card.
***nWE**	Output	Write Enable for Memory PCMCIA card.
nREG	Output	When this signal is LOW, memory access is restricted to Attribute memory space.
***nCD1, nCD2**	Input	Card Detect Input 1 and 2
***VS1, VS2**	Input	When this signal is LOW, memory access is limited to Attribute memory space.
***BVD1** **(nSTSCHG/nRI)** **(I/O card)**	Input	The battery voltage detect input 1 on the memory PCMCIA Card. During normal access for I/O, this signal must be kept LOW. In I/O Card, this signal is used as Card Status Change or Ring Indicate input according to the Host Bus Adapter's configuration setting.
***BVD2** **(SPKR#)**	Input	The battery voltage detect input 2 on the memory PCMCIA Card. In I/O Card, this signal is used as Speaker input.
***RDY/nBSY** **(nIREQ)**	Input	Ready/nBusy input on the memory PCMCIA Card. In I/O Card, this signal is used as Interrupt Request signal.

Table 10-2: PCMCIA Card Slot Interface

Name	Type	Description
***WP** **(nIOIS16)**	Input	Write Protect input on the memory PCMCIA Card In I/O Card, this signal is used to indicate the I/O data size.
***nWAIT**	Output	Bus cycle wait input from PCMCIA card
***RESET**	Input	Card Reset Output
nINPACK	No Conn	Not used in this specification.

Table 10-2: PCMCIA Card Slot Interface (Continued)

*The signals marked with an asterisk are required for each socket interface.

10.1.4 Other control signals

These outputs from the GMS30C7201 do not connect to the PCMCIA card.

Name	Type	Destination	Description
nSPKROUT	Output	System Speaker	Speaker Output. Passes through nSPKR from an I/O PCMCIA Card.
PCMIRQSKT1	Output	Interrupt Controller	Interrupt Source for Socket 1
PCMIRQSKT2	Output	Interrupt Controller	Interrupt Source for Socket 2
nVcc5EN	Output	Power Logic	Power Control (5V)
nVcc3EN	Output	Power Logic	Power Control (3V)
VppEN0	Output	Power Logic	Program Power Supply Control 0
VppEN1	Output	Power Logic	Program Power Supply Control 1
CaDRV	Output	External Buffer	When HIGH, socket1 is active. This signal selects the data path.
CbDRV	Output	External Buffer	When HIGH, socket1 is active. This signal selects the data path.

Table 10-3: Other Control Signals

Note **nSPKROUT** *is supplied to the system speaker to provide a single amplitude on/off binary audio wave. In the example, the Modem Card can use this signal to inform the system user that they are being called.*

10.1.5 EBI (External Bus Interface) timing control signals

These signals come from the PCMCIA Interface and go to the External Bus Interface (EBI).

Name	Type	Destination	Description
PCMCIAEn	Output	EBI	This signal informs EBI that PCMCIA Host Bus Adapter wants to transfer data.
nPCMCIAOutEn[1:0]	Output	EBI	This signal informs EBI that current cycle is a write cycle and also indicates the write byte size.
nPCMCIAOutLEn	Output	EBI	EBI output data latch enable signal
nPCMCIAInEn	Output	EBI	This signal informs the EBI that current cycle is read cycle.
nPCMCIAInLEn[3:0]	Output	EBI	EBI Address Latch enable signal
nPCMCIALatch	Output	EBI	Address Latch enable signal
PCMCIAByteSeq[1:0]	Output	EBI	Control s the data sequence byte lane
SELECTEDBA[6:0]	Output	EBI	These bits are multiplexed with BA [25:21,1:0] at EBI

Table 10-4: External Bus Interface (EBI) Timing Control Signals

10.2 Register Description and Map

Interface Status Register (Read Only): Denotes the state of socket status

Bits	Description
7	Voltage Sense 2 input status HIGH=voltage present
6	Voltage Sense 1 input status HIGH=voltage present
5	RDY/nBSY at Memory only card Interface 0: Busy 1: Ready In I/O or Mixed Memory and I/O card interfaces this signal indicates the status of nIREQ pin of the connected card
4	Memory Write Protected at Memory only card interface 0: Not write protected 1: Write Protected In I/O or Mixed Memory and I/O card interface this signal indicates the status of nIOIS16 pin of connected card. (However, nIOIS16 pin is not supported in this specification so, this has no meaning.)
3–2	Card Detect 2 and Card Detect 1.These bits are asserted simultaneously and indicate a card is present and is fully seated in the socket.
1	BVD2 Battery Voltage Detect2 In the I/O Card, this denotes the current status of the **nSPKR.** it does not show interrupt status. See note.
0	BVD1 Battery Voltage Detect1 In the I/O Card, this denotes the current status of the **nSTSCHG/nRI.** If the Ring enable bit is set, it does not show interrupt status.See note.

Table 10-5: Interface Status Register Bits

Note *For bit1–bit0:*

00 Battery Dead

01 Battery Warning

10 Battery Dead

11 Battery Good

Card Status Change Register

Bit	Description
7	Software Card Detect Change
6	Card Detect Change
5	Interrupt request from PCMCIA card 0: No interrupt request (for pulse mode interrupts) 1: pulse mode interrupt requested (**IREQ** pin from card)
4	If Ring Indicate enable bit is 0 and **nSTSCHG/nRI** is 0, then this bit is 1. Otherwise this bit is 0.
3	Ring Indicate Change
2	Ready Change
1	Battery Warning Condition. No meaning in I/O Card
0	Battery Warning Condition. No meaning in I/O Card

Table 10-6: Card Status Change Register Bits

The Card Status Change Register contains the status for the sources of the card status change interrupt. These sources can be enabled to generate a card status change interrupt by setting the corresponding bit in the Card Status Change Interrupt Configuration Register. There are two ways to reset this register:

- Read the Card Status Change Register
- Write back in the Card Status Change Register after setting Explicit Write Back Card Status Change Acknowledge bit to 1 in the Global Control Register

Power and RESETDRV Control Register

This register controls the power.

Bit	Description
7	Host Bus Adapter SoftReset Setting this bit resets the HBA
6	Output Enable. If this bit is set to 0, the output signals **nCE2, nCE1, nCIORD, nCIOWR, nOE, nREG, RESET,** and **nWE** are tristated.
5	Auto Power Switch Enable. 1: Automatic socket power switching based on card detect is enabled.
4	Socket Power Control bit 0: **Vcc5EN** = 1, **Vcc3EN** = 1 1: Bit 1 and Bit 0 determine **Vcc5EN**, **Vcc3EN** Values

Table 10-7: Power and RESETDRV Control Register

Bit	Description
3–2	Socket Programming Control bits 11 **VppEN2** = 1 **VppEN1**=1 10 **VppEN2** = 1 **VppEN1**=0 01: **VppEN2** = 0 **VppEN1**=1 00: **VppEN2** = 0, **VppEN1**=0
1	Direct 5V/3.3V Switch enable **VS1** controls the socket power When this bit is HIGH, then pin VS1controls voltage selection. If pin **VS1** is LOW, **Vcc5EN** is active. If pin **VS1** is HIGH, **Vcc3EN** is active
0	Voltage Selection Bit. This can only be used when bit 1 above is LOW. 0: **Vcc5EN** is activated if bit 4 is set 1: **Vcc3EN** is activated if bit 4 is set

Table 10-7: Power and RESETDRV Control Register (Continued)

Card Detect and Global Control Register

The Card Detect and Global Control Register controls the operation of the Host Bus Adapter (HBA)

Bit	Description
5	PC Card Reset. Setting this bit to 1 activates the RESET signal to the PC. card. The RESET signal is active until this bit is set to 0.
4	Software Card Detect Interrupt. If Card Detect Enable bit is set to1 in the Card Change Interrupt Configuration Register, then writing 1 to S/W card detect bit in the Card Detect and General Control Register will cause a card detect and status change interrupt. This interrupt source is set when the PCMCIA Card exits from the CARD Power Down Mode
3	PC Card Type 0:Memory Card 1: I/O Card
2	IO Card Access Type 0: Accessed I/O address range has 8 bit register 1: Accessed I/O address range has 16 bit register
1	Explicit Write Back Card Status Change Acknowledge bit. Setting this bit to 1 will require an explicit write back of a ‘1’ to the Status Change Register bit which indicates an interrupt condition to acknowledge the interrupt. When this bit is 0 (default value), the Card Status Change interrupt is acknowledged by reading the Card Status Change Register, and the register bits are cleared on a read. 0: No write back is required 1: Write back is required

Bit	Description
0	Card Power Down Mode 0: Power is supplied to Socket 1: Power is not supplied to Socket. Card Power Down Mode Although the PC card is in the socket, the power is not supplied to the socket. So, there are large system power savings.Software Card Detect Interrupt is required to exit from Card Power Down Mode

Card Status Interrupt Configuration Register: Masking Register for Interrupt

Bit	Description
7	Ring Indicate / Status Changed Mask 0: **RI/nSTSCHG** interrupt masked 1: **RI/nSTSCHG** interrupt enabled
6	Card Detect Enable 1 for enable 0 for masking
5–4	PC card generated Interrupt (**nIREQ**) enable 00: PC card generated interrupt (**nIREQ**) is not enabled 01 The level-mode **IREQ** interrupt request signal is accepted An interrupt occurs when 0 is detected at **nIREQ** pin. 10: The pulse-mode **IREQ** interrupt request signal is accepted An interrupt is occurs when the falling edge is detected from **nIREQ** pin 11: The pulse-mode **IREQ** interrupt request signal is accepted An interrupt is occurs when the rising edge is detected from **IREQ** pin.
3	Ring Indicate Enable 1 for enable 0 for masking (No meaning in Memory Card)
2	Ready Enable 1for enable 0 for masking (No meaning in I/O Card)
1	Battery Warning Enable 1 for enable 0 for masking
0	Battery Dead Enable 1 for enable 0 for masking In I/O Card, if Ring Indicate Enable bit is 0, **nSTSCHG** to 0 interrupt source is enabled

Table 10-9: Card Status Interrupt Configuration Register

Memory Area Access Timing Control Register

Bit7: Reserved

Bits 6–4: Wait Control Bits

Bit	Description
6-4	Wait Control bits 000: 5 Clock Wait State 001: 6 Clock Wait State 010: 7 Clock Wait State 011: 9 Clock Wait State 100: 11 Clock Wait State 101: 12 Clock Wait State 110: 14 Clock Wait State 111: 28 Clock Wait State

Table 10-10: Bits 6-4 Wait Control Bits

bit 3–bit 2: nOE/nWE Assertion Control bits

Bit	Description
3–2	Delay times 00: 0.5 Clock Delay 01: 1.5 Clock Delay 10: 3.5 Clock Delay 11: 5.5 Clock Delay

Table 10-11: Bits3-2 Clock Delays

bit1–bit0: nOE/nWE Negation Control bits

Bit	Description
1–0	Number of Wait States 00: 0.5 Clock Delay 01: 1.5 Clock Delay 10: 3.5 Clock Delay 11: 5.5 Clock Delay

Table 10-12: Bits 1-0 Negation Control Bits

I/O Area Access timing Control Register

bit7: Reserved

bit6–bit4: Wait Control Bits

Bit	Description
6–4	Wait Control bits 000: 5 Clock Wait State 001: 6 Clock Wait State 010: 7 Clock Wait State 011: 9 Clock Wait State 100: 11 Clock Wait State 101: 12 Clock Wait State 110: 14 Clock Wait State 111: 28 Clock Wait State

Table 10-13: Bits 6–4 Wait Control Bits

bit3–bit2: nOE/nWE Assertion Control bits

Bit	Description
3–2	Delay times 00: 0.5 Clock Delay 01: 1.5 Clock Delay 10: 3.5 Clock Delay 11: 5.5 Clock Delay

Table 10-14: Bits 3-2 Assertion Control Bits

bit1–bit0: nOE/nWE Negation Control bits

Bit	Description
1–0	Number of Wait States 00: 0.5 Clock Delay 01: 1.5 Clock Delay 10: 3.5 Clock Delay 11: 5.5 Clock Delay

Table 10-15: Bits 1–0 Negation Control Bits

Default value for I/O Access => bit6–bit0 = 100 10 01

ATA Control Register: ATA Interface Control Register

Bit	Description
7	Reserved
6	Reserved
5	In ATA mode, the value of this bit appears at CA25

Table 10-16: ATA Interface Control Register

Bit	Description
4	In ATA mode, the value of this bit appears at CA24
3	In ATA mode, the value of this bit appears at CA23
2	In ATA mode, the value of this bit appears at CA22
1	In ATA mode, the value of this bit appears at CA21
0	ATA Mode bit “0”: PCMCIA Mode (I/O Mode) “1”: ATA Mode

Table 10-16: ATA Interface Control Register

Recommended Value for Access Timing Control Register

Memory Card	Recommended Value
100 ns	0 000 00 00
150 ns	0 000 01 01
200 ns	0 001 01 01
250 ns	0 010 01 01
300 ns	0 01101 01
600 ns	Read:0 110 10 10 Write:0 110 10 10
I/O Register	x 010 10 01

Table 10-17: Access Timing Control Register Values

10.2.1 Test mode registers

The following registers are used for test purposes only. Each input test register’s output bit is multiplexed with the corresponding input signal of AMBA or PCMCIA input signal to the designed PCMCIA Host Bus adapter. So, in test mode, the function of designed PCMCIA HBA can be done exactly. Output signals are also stored in the output test registers and can be read by the host in test mode.These test registers are enabled only test mode and can be reset by the command of Host.

By using this scheme of test methodology, the function of PCMCIA Host Bus Adapter can be exactly examined without the help of other system peripheral blocks because all input signals and output signals which communicate with other peripherals are derived from input test registers and stored into output test registers. Only critical system components such as the bus decoder and arbiter and Test mode controller which act as bus master are required for testing. The following test register description gives only the information about the counterpart signal of normal operation.

Test Mode Control Register

Bit	Description
7	Socket 1 0: No HBA Power Down Mode 1: HBA enters Power Down Mode. Card Status Monitoring Unit also enters Power Down. All of the module (except control register which should inform the wake up from power down) enters Power Down Mode. So, the Controller is in the minimum power consumption mode.
6	Socket 2 Power Down Mode 0: No HBA Power Down Mode 1: HBA enters Power Down Mode. Card Status Monitoring Unit also enters Power Down. All of the module (except control register which should inform the wake up from power down) enters Power Down Mode. So, the Controller is in the minimum power consumption mode.
5	Power Down Mode 0: No Power Down Mode 1: Power Down Mode Enable In this mode, only Address Change Logic enters the Power Down Mode and **BCLK** is masked. Other parts of Controller do not enter Power Down and are active, including **BCLK**.
4	Test Mode Enable Bit
3	Test Clock Bit In Test mode this bit acts on the clock.
2	Test Mode Register Reset Bit 1: Test Mode enabled
1	**nWAIT** of Socket 1
0	**nWAIT** of Socket 0

Table 10-18: Test Mode Control Register

Bus Address Input Test Register

Bit	Description
7–0	**BA[28:21]**

Table 10-19: Bus Address Input Test Register

Bus Control Input Test Register

Bit	Description
7–6	Reserved
5	**DSELREG**
4	**BA[1:0]**
3	**DSELPCM**
2	**BWRITE**
1–0	**BSIZE[1:0]**

Table 10-20: Bus Control Input Test Register

Socket 1 Status Input Test Register

Bit	Description
7–6	Inverse of **nVS2** and **nVS1**
5	**RDY/nBSY**
4	**WP**
3	Inverse of **nCD2**
2	Inverse of **nCD1**
1	**BVD2**
0	**BVD1**

Table 10-21: Socket 1 Status Input Test Register

Socket 2 Status Input Test Register

Bit	Description
7–6	Inverse of **nVS2** and **nVS1**
5	**RDY/nBSY**
4	**WP**
3	Inverse of **nCD2**
2	Inverse of **nCD1**
1	**BVD2**
0	**BVD1**

Table 10-22: Socket 2 Status Input Test Register

Slave Access Output Test Register

Bit	Description
7–4	Reserved
3	**BWAIT**
2	**BERROR**
1	**BLAST**
0	**nSPKROUT**

Table 10-23: Slave Access Output Test Register

EBI Output Test Register1

Bit	Description
7–6	Reserved
5	**PCMCIAEn**
4	**nPCMCIAOutLEn,**
3	**nPCMCIAInEn**
2	**nPCMCIALatch**
1–0	**PCMCIAByteSeq[1:0]**

Table 10-24: EBI Output Test Register1

EBI Output Test Register2

Bit	Description
7–6	Reserved
5–4	**nPCMCIAOutEn[1:0]**
3–0	**nPCMCIAOutEn[3:0]**

Table 10-25: EBI Output Test Register 2

Power Control Output Test Register

Bit	Description
7	Socket 1 Power Control Signal **nVCC5En**

Table 10-26: Power Output Control Test Register

Bit	Description
6	Socket 1 Power Control Signal **nVCC3En**
5	Socket 1 Power Control Signal **VPPEN2**
4	Socket 1 Power Control Signal **VPPEN1**
3	Socket 2 Power Control Signal **nVCC5En**
2	Socket 2 Power Control Signal **nVCC3En**
1	Socket 2 Power Control Signal **VPPEN2**
0	Socket 2 Power Control Signal **VPPEN1**

Table 10-26: Power Output Control Test Register (Continued)

Address Change Output Test Register1

Bit	Description
7	Reserved
6–0	**SELECTEDBA[6:0]**

Table 10-27: Address Change Output Test Register1

Address Change Output Test Register2 (Socket 1 Access signal)

Bit	Description
7–6	Reserved
5	**nCE2**
4	**nCE1**
3	**nCIORD**
2	**nCIOWD**
1	**nOE**
0	**nWE**

Table 10-28: Address Change Output Test Register2

Address Change Output Test Register3 (Socket 2 Access signal)

Bit	Description
7–6	Reserved
5	**nCE2**
4	**nCE1**
3	**nCIORD**
2	**nCIOWD**
1	**nOE**
0	**nWE**

Table 10-29: Address Change Output Test Register 3

Interrupt Output Test Register

Bit	Description
7–5	Reserved
6-4	**CaDRV**, **CbDRV**, **nREQ**
3-2	**RESETSKT1**, **RESETSKT2**
1-0	**PCMIRQSKT1**, **PCMIRQSKT2**

Table 10-30: Interrupt Output Test Register

10.2.2 Register map

Register Name	Socket 1 Address	Socket 2 Address	Description
Interface Status Register	BaseAddress + 0h0000	BaseAddress + 0h0020	# 8-bit R/O
Card Status Change Register	BaseAddress + 0h0004	BaseAddress + 0h0024	# 8-bit R/W
Power and RESETDRV Control Register	BaseAddress + 0h0008	BaseAddress + 0h0028	# 8-bit R/W
Card Detect and Global Control Register	BaseAddress + 0h000C	BaseAddress + 0h002C	# 8-bit R/W
Card Status Interrupt Configuration Register	BaseAddress + 0h0010	BaseAddress + 0h0030	# 8-bit R/W
Memory Area Access Timing control Register	BaseAddress + 0h0014	BaseAddress + 0h0034	# 8-bit R/W
I/O Area Access Timing Control Register	BaseAddress + 0h0018	BaseAddress + 0h0038	# 8-bit R/W
ATA Control Register	BaseAddress + 0h001C	BaseAddress + 0h003C	# 8-bit R/W

Table 10-31: Register Map Socket 1 and Socket 2 Addresses

10.2.3 Test register map

Register Name	Address	Description
Test Mode Control Register	BaseAddress + 0h0040	# 8-bit R/O
Bus Address Input Test Register	BaseAddress + 0h0044	# 8-bit R/W
Bus Control Input Test Register	BaseAddress + 0h0048	# 8-bit R/W
Socket 1 Status Output Test Register	BaseAddress + 0h004C	# 8-bit R/W
Socket 2 Status Output Test Register	BaseAddress + 0h0050	# 8-bit R/W
Slave Access Output Test Register	BaseAddress + 0h0054	# 8-bit R/W
EBI Output Test Register1	BaseAddress + 0h0058	# 8-bit R/W
EBI Output Test Register2	BaseAddress + 0h005C	# 8-bit R/W
Power Control Output Test Register	BaseAddress + 0h0060	# 8-bit R/W
Address Change Output Test Register1	BaseAddress + 0h0064	# 8-bit R/W
Address Change Output Test Register2	BaseAddress + 0h0068	# 8-bit R/W
Address Change Output Test Register3	BaseAddress + 0h006C	# 8-bit R/W
Interrupt Output Test Register	BaseAddress + 0h0070	# 8-bit R/W

Table 10-32: Test Register Map

10.3 Functional Description

10.3.1 Socket 1 and Socket 2 Access

To reduce the external pin count, common data and address signals are used for socket 1 and socket2 with two additional control signals (***CaDRV, CbDRV***). This scheme dramatically reduces the external pin count with some burdens of board level system design. An external buffer selects the exact path. Address and Data paths uses EBI for the same reason above. And to meet the wide range of address setup time and hold time of various PCMCIA card according to the access time deference, some software control method is used. **Access Timing Control Register** is used for this scheme**.** The access time of PCMCIA Card is relatively very slow when compared to the host system operating clock speed. So, the controller must provide some signal e.g, **BWAIT**, to inform the host that current access has not finished**.** Three wait conditions guarantee normal operation. See ***Figure 10-4: Memory Read Cycle With Wait***:

1 default wait until PCMCIA Card drives the wait

2 PCMCIA card driven wait

3 additional wait to meet the long address hold time - AND-ORed to inform the host of the state of current access

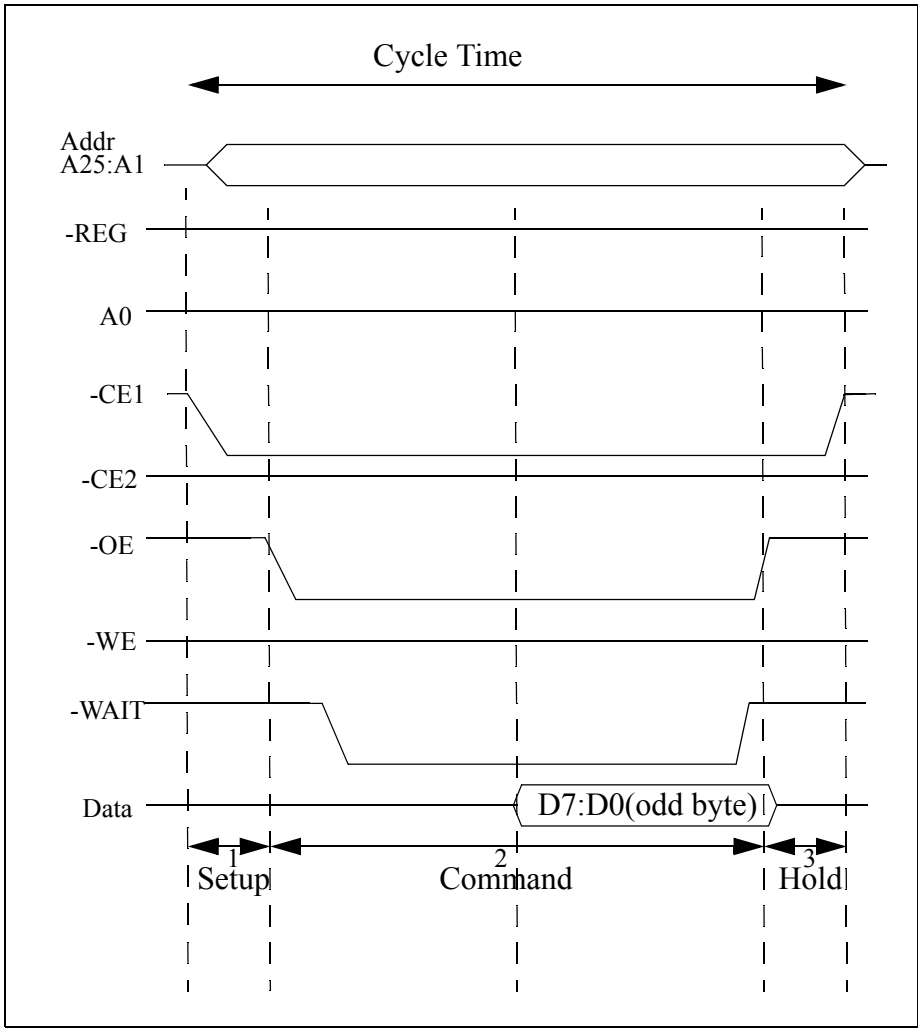


Figure 10-4: Memory Read Cycle With Wait

A total of 256 Mbytes address range is allocated for each PCMCIA Card. (So, there is no need for an address window as an ISA based PCMCIA card controller is used to access a total 64 Mbyte with very small system address range.) For the fast access support of Mixed I/O and Memory Card, each region has 64M byte address range.

Address Range	Function Area
0 - 64 Mbyte	Attribute Memory Area
64 - 128 Mbyte	Common Memory Area
64 - 192 Mbyte	I/O Area
192 - 256 Mbyte	Reserved for future use

Table 10-33: Address ranges and function areas

Also, the IOIS16 signal is not used.

Note *External Buffer Requirement:*

The external buffer must operate in both 3.3v and 5.0v to support low power cards. **CaDRV** *and* **CbDRV** *control the External buffer to pass the correct signals to the socket. LCX16244 is the 5V tolerant Buffer which has the enable pin. The data buffer must be bidirectional. This bidirectional port can be easily controlled with an additional control pin. LCX16245 is the 5V tolerant buffer which has an enable pin and a direction control pin. The PCMCIAOutLEn can be used for direction control pin of the LCX16245.*

Controller Pad Requirement:

Pads must tolerate 5v input from 5v PCMCIA Card.

10.3.2 Interrupt handling

Interrupt handling is the most important role in the PCMCIA Host Bus Controller. Interrupts are generated from two different sources, the Card Status change interrupt and a Card-generated interrupt which requires host's attention, e.g, data transmission buffer is full, so, it's requires some read operation from the host. Today, PCMCIA Cards can generate level mode or pulse mode interrupts. A different mechanism is required for Card status change interrupt handling. This interrupt source is monitored and change status is saved in relevant registers, the Interface Status Register and Card Status Change Register. Card Status Change includes the insertion or removal of the card, battery warning or dead, and so on. Only Level mode interrupt is used in this type of interrupt source. Each interrupt can be masked by setting the corresponding bit of the Card Status Interrupt Configuration Register. The host can know the exact interrupt source by reading the Interface Status Register and Card Status Change Register.

10.3.3 Enhanced power management

Three power down operations are mode supported for enhanced power management. Most power is saved by disabling the clock to inactive blocks. Power down is entered under software control.***Table 10-34: Power Down Modes*** provides a summary.

Mode	Summary
Partial Power Down	Interrupts still operate (to monitor card changes).
Full Power Down	All internal control logic disabled.

Mode	Summary
Card Power Down	Power removed from card (maximum power saving).

Table 10-34: Power Down Modes (Continued)

10.3.4 Internal register access

The internal register of the interface is accessed directly. That is, no indirect addressing method is used. The internal address decoder generates the read/write strobe for each register. These registers are reset by the **BnRES** signal when the system is power on. Software reset can also be used when a card is removed from each socket.

10.3.5 Support for compact flash memory

Compact Flash memory Card is supported by the controller. Compact Flash can be supported in either I/O or Memory interface. But no dedicated true IED interface mode is supported. ATA Interface can be used to support this type of card. In general, an external interconnection socket is required because Compact Flash uses 50 pin interface.

10.3.6 General operation flow

***Initial State*: All Registers reset to default initial value.**

Card is not inserted in the socket.

***Card Detected*: Card is inserted in socket.**

Power up sequence is performed (Fully Software Supported) Hardware support is limited to card's operating voltage level detect. CIS is read by default state. PCMCIA Spec defines the default timing and control for CIS Reading Hardware is programmed by using the CIS information (Memory only Card or Memory or IO Card).

***normal operation*: Card Access is performed.**

Access timing control is done by ATC Register and Card - driven wait signal. IOIS16 signal is not supported. So, only 16 bit register IO card and 8 bit Register IO Card is supported Address and Data line uses the EBI (External Bus Interface).

***Status Change*: Card Status is changed at both type of PCMCIA Card.**

Controller informs the status change using Interrupt Status change interrupt. This interrupt can be masked by using CSIC Register.

***Card Removal*: All Internal Register is reset.**

10.3.7 Access timing spec of attribute memory read and write

The Attribute Memory's access time is defined as 300ns at 5V VCC or 600ns at 3.3V VCC.See ***Chapter 15,*** Electrical Characteristics for timing values.

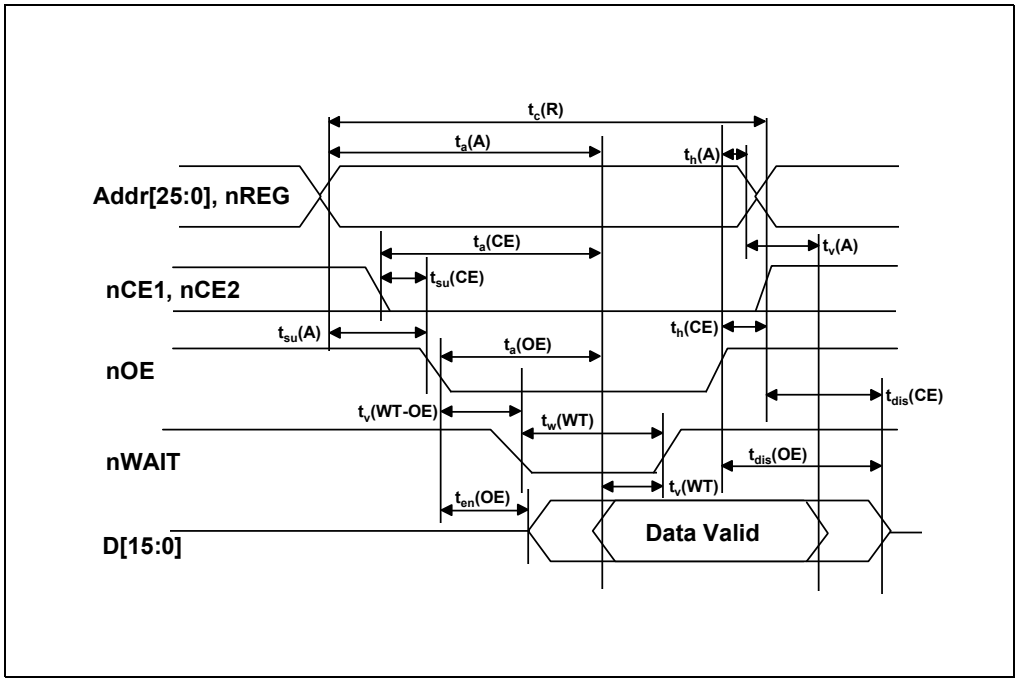


Figure 10-5: Read timing diagram

Write Timing

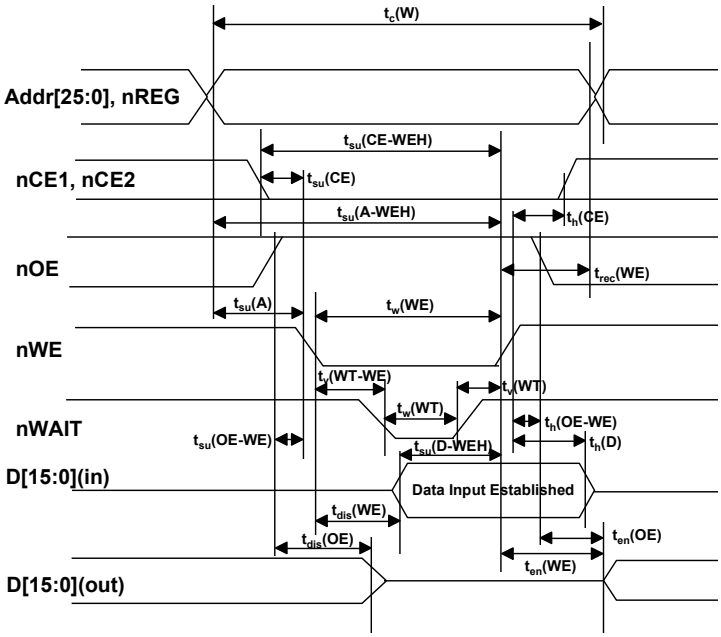


Figure 10-6: Write timing diagram

11 LCD & VGA Controllers

LCD & VGA Controllers

11.1 Overview

The VGA and LCD controllers provide the video output capabilities of the GMS30C7201. The VGA and LCD controllers can be used simultaneously, and can display either the same image, or entirely different images. If the images are the same for the two displays, and the frame rates required are also the same, then it is possible to use the same frame buffer for both controllers. If either the images are different, or different frame rates are required, then it is necessary to use two frame buffers, or one frame buffer, but with two DMA channels. In either case, the DMA bandwidth required for the two video sub-systems is the sum of the bandwidth for each display, whereas if the two images and the frame rates are the same, then only one DMA channel is used, and the DMA data is shared between the two controllers.

11.1.1 LCD features

- Single panel color and monochrome STN displays
- TFT color displays
- Resolution programmable up to 640x480
- Single panel mono STN displays with either 4- or 8-bit interfaces
- 15 grey-level mono support, 3375 color STN support
- 4bpp mono, 4 or 8bpp palettized color displays
- 12bpp color ‘true-color’ non-palettized color displays
- Programmable timing for different display panels
- 3 x 256 entry, 4-bit palette RAM
- Patented grayscale algorithm
- Little-endian operation

Note *The controller does* **not** *support dual panel STN displays.*

There is no hardware cursor support, since WinCE does not use a cursor.

11.1.2 VGA features

- Resolution programmable up to 800x600 resolution (640x480 maximum for simultaneous display mode)
- Programmable border color
- Programmable VGA timing
- On-chip video DACs for direct drive of a monitor
- Option to share video data with LCD interface
- 4bpp, 8bpp and 16bpp modes
- 5:5:5 and 5:6:5 16bpp ‘true-color’ modes
- 3 x 256 entry 8-bit palette RAM
- Patented technique for selecting 2^{16} colors from 2^{24} colors in 16bpp mode
- Little-endian operation

Note *The controller does* **not** *support interlaced displays. It supports non-interlaced monitor output only, and therefore cannot be used as a television display.*

There is no hardware cursor support, since WinCE does not use a cursor.

11.2 Video operation

A block diagram of the video system is shown in ***Figure 11-1: Video System Block Diagram***. The video system has two separate data paths. One data path is for STN LCD and for TFT LCDs, and the other data path is for the VGA controller. If the frame rate and display resolution are the same for the LCD and VGA, and if the images are the same, it is possible to share the DMA data between the two data paths, by writing DMA data into both the input FIFOs. Sharing DMA data between the data paths has some restrictions in the programming of the VGA and LCD timing information. This is explained in ***11.2.6 Sharing VGA and LCD data*** on page 11-6.

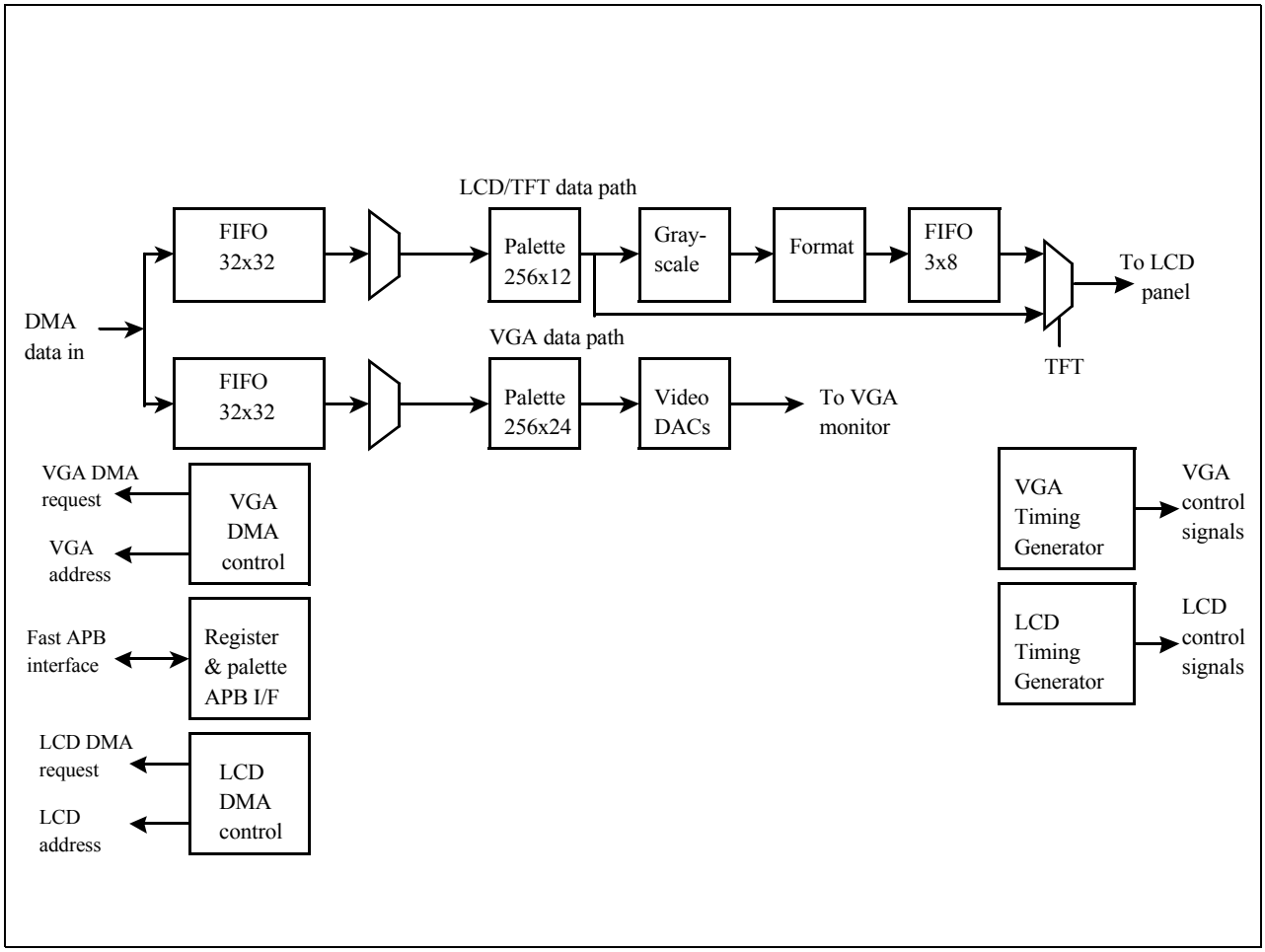


Figure 11-1: Video System Block Diagram

11.2.1 VGA data path

The VGA data path is the simpler of the two data paths. Data is received from the video ASB bus into a 32 deep by 32-bit wide asynchronous FIFO. The FIFO data is then extracted into a holding latch, and multiplexed down to a pixel at a time. This logical pixel data is then passed to the palette RAM. The palette RAM is composed of three 256 x 8-bit RAMs. There is a RAM array for each of the R, G and B color components of a pixel. The physical pixel data from the palette RAM is then passed to the three video DACs.

Palette RAM, and 16bpp mode

Logical pixels are either 4, 8 or 16 bits. In 4- and 8-bit modes, the logical pixel value is used to index into the three palette arrays to select the three color components of the physical pixel value. In 16-bit pseudo true-color mode, a patented technique is used to allow 2^{16} colors to be selected from 2^{24} possible colors. Separate color gun values are independently used to index into the three palette arrays, to select an 8-bit value for each of the color guns. By splitting the palette RAM into three separate RAM arrays, it allows 16-bit mode to generate 8-bit color gun data. The method used is an ARM patented technique, where 16bpp data is split into three over-lapping 8-bit fields that are used to index into the three RAM arrays. The red gun is indexed by bits 15:8 of the 16-bit pixel value, the blue gun is indexed by bits 7:0 of the pixel value, and the green gun is indexed by bits 11:4 of the pixel value. By programming the palette with the correct values, 5:5:5, 5:6:5, 4:8:4, and many other combinations of 16-bit data may be used. Thus:

4 bpp	16 palette entries are used for each palette array. All three palette RAMs are indexed by pixel[3:0]
8 bpp	256 palette entries are used for each palette array. All three palette RAMs are indexed by pixel[7:0]
16 bpp	256 palette entries are used for each palette array. Red array is indexed by pixel[15:8], green array is indexed by pixel[11:4], and blue array is indexed by pixel[7:0]

11.2.2 VGA control

The control logic for the VGA block consists of 3 main blocks.

- a video timing generator block

  This divides down the pixel clock from the video PLL and produces the timing control signals (**VSync**, **HSync**, whether it is video data or border color to be displayed) for the monitor and video DACs.
- a DMA bus control block

  This generates the DMA address for the SDRAM controller and generates the DMA request signal to the SDRAM controller, and controls receiving the DMA data, and writing it into the VGA FIFO.
- a register slave interface

  This resides on the fast peripheral bus and allows registers to be written from the processor. The registers and palette are fast peripheral bus slaves.

The LCD and VGA palettes have separate address spaces for reads and writes. However, there is also a combined address space for palette writes. Writing to this area causes both the LCD and the VGA palettes to be updated with the same data. When this option is used, 4 bits of each LCD color gun value are thrown away, so that the 8 bits per gun of the VGA palette data can be used to program the LCD palette. Thus the LCD gets the MS 4 bits of each VGA gun data. The LCD palette write is organized such that it used the same format for writes from the CPU as the VGA palette data, with it discarding the LS 4 bits of each gun data.

11.2.3 LCD datapath

The LCD data path is similar to the VGA data path, but it has a few additions. In TFT mode, it is similar to VGA, except that the digital RGB data is output directly to the pins of the chip, without going via a video DAC. However, in STN mode, the data must be grayscaled, and then formatted for the LCD panel. The grayscaler block converts the 4 bit per color gun data into a single bit per gun, using a patented time/space dither algorithm. In mono mode, only the B gun data is used. The output of the grayscaler is fed to the formatter, which formats the pixels in the correct order for the LCD panel type in use. (4 or 8 mono pixels per clock for mono panels, or $2^2/_3$pixels per clock for color data.) The output of the formatter in color mode is bursty, due to the $2^2/_3$pixels per clock that are output, so the formatter output goes to a small FIFO, which smooths out this burstiness, before data is output to the LCD panel at a constant rate.

11.2.4 Color/Grayscale Dithering

Entries selected from the look-up palette are sent to the color/grayscale space/timebase dither generator. Each 4-bit value is used to select one of 15 intensity levels. Note that two of the 16 dither values are identical. The table below assumes that a pixel data input to the LCD panel is active HIGH. That is, a '1' in the pixel data stream will turn the pixel on, and a '0' will turn it off. If this is not the case, the intensity order will be reversed, with "0000" being the most intense color. This polarity is LCD panel-dependent.

The gray/color intensity is controlled by turning individual pixels on and off at varying periodic rates. More intense grays/colors are produced by making the average time that the pixel is off longer than the average time that it is on. The proprietary dither algorithm is optimized to provide a range of intensity values that match the eye's visual perception of color/gray gradations, with smaller changes in intensity nearer to the mid-gray level, and greater nearer the black and the white levels. In color mode, red, green and blue components are gray-scaled simultaneously as if they were mono pixels.

The duty cycle and resultant intensity level for all 15 color/grayscale levels is summarized in ***Table 11-1: Color/grayscale intensities and modulation rates***.

Dither Value (4-bit value from palette)	Intensity (0% is white)	Modulation Rate (ratio of ON to ON+OFF pixels)
0000	0.0%	0
0001	11.1%	1/9
0010	20.0%	1/5
0011	26.7%	4/15
0100	33.3%	3/9
0101	40.0%	2/5
0110	44.4%	4/9
0111	50.0%	1/2
1000	55.6%	5/9
1001	60.0%	3/5
1010	66.6%	6/9
1011	73.3%	11/15
1100	80.0%	4/5
1101	88.9%	8/9
1110	100.0%	1
1111	100.0%	1

Table 11-1: Color/grayscale intensities and modulation rates

11.2.5 TFT mode

When TFT display mode is enabled, the timing of the pixel, line and frame clocks as well as the AC-bias pin change. The pixel clock transitions continuously in this mode as long as the LCD is enabled. The AC-bias pin functions as an output enable. When it is HIGH, the display latches data from the LCD's pins using the pixel clock. The line clock pin is used as the horizontal

synchronization signal (HSYNC), and the frame clock is used as the vertical synchronization signal (**VSync**). Pixel data is output one pixel per clock, rather than 4, 8 or $2^{2}/_{3}$pixels per clock, as it is in the passive LCD modes.

11.2.6 Sharing VGA and LCD data

Generally when the LCD and the VGA interface are running concurrently, both are operating entirely independently, and therefore two separate DMA channels are running at the same time. This means that the memory bandwidth consumption is the sum of the bandwidth required for the two DMA channels. Clearly this is not very efficient if the LCD and the VGA are displaying the same data. Therefore, there is an option to share data between the VGA and the LCD.

The timings of the VGA and the LCD must be synchronized. The LCD timing generator must be programmed so that the LCD slightly trails the VGA in outputting pixel data. Since there is a common DMA data path to the VGA and the LCD, the data is written into both FIFOs. The request is only generated when there is sufficient space in both FIFOs for the DMA data. This means that the FIFO levels of the two displays must be kept as close as possible. They cannot be kept exactly the same, because of the bursty nature of the LCD data, especially in color STN mode. However, by programming a small horizontal back porch offset from the LCD to the VGA, they can be kept broadly similar.

It is important that the free space in each of the FIFOs is kept as close as possible, because the request is only generated when there is enough space in both FIFOs, and if the FIFO levels are significantly different, then FIFO underflow could occur on one FIFO, due to the other FIFO not having reached the level at which its request is generated. When the images on the two screens are likely to be different at some stage in the future but are currently the same, the two DMA channels can be operated independently. The VGA and LCD enables “on” in the same control register, so they can be enabled simultaneously.

11.3 Video Control register

This register contains control bits for both the VGA and the LCD controllers. The reason that both VGA and LCD control bits are in the same register is to allow the simultaneous enabling of LCD and VGA when sharing DMA data.

11.3.1 LCD Power Control

LCD displays require that the LCD is running before power is applied. For this reason, the LCD's power on control is not set to "1" unless both **LcdEn** and **LcdPwr** are set to "1". Note that most LCD displays require the **LcdEn** must be set to "1" approximately 20ms before **LcdPwr** is set to "1" for powering up. Likewise, **LcdPwr** is set to "0" 20ms before **LcdEn** is set to "0" for powering down.

Bit	Name	Description
0	**LcdEn**	LCD Controller Enable 0 - LCD controller disabled 1 - LCD controller enabled
2-1	**LcdBpp**	LCD Bits Per Pixel 00 - 4bpp 01 - 8bpp 10 - 16bpp 11 - Reserved
3	**LcdBW**	LCD Monochrome 0 - Color operation enabled 1 - Monochrome operation only enabled
4	**LcdTFT**	LCD TFT 0 - Passive or STN display operation enabled 1 - Active or TFT display operation enabled
7-5	**-**	Reserved
8	**VgaEn**	VGA Controller Enable 0 - VGA controller disabled 1 - VGA controller enabled
10-9	**VgaBpp**	VGA bits per pixel 00 - 4bpp 01 - 8bpp 10 - 16bpp 11 - Reserved
11	**ShareDMA**	Share DMA Data If this bit is set, the DMA data streams for LCD and VGA are shared. The request is only generated when there is space for data in both FIFOs (both FIFO requests should be programmed to 8 words). The LCD timing generator should be slaved off the VGA timing generator, with the clock source set as the VGA clock.
12	**BGR**	0 - RGB normal video output for both LCD and VGA 1 - BGR red and blue swapped for both LCD and VGA
15-13	**-**	Reserved

Table 11-2: Video control register VideoControl

Bit	Name	Description
17-16	**VgaVComp**	Generate interrupt at: 00 - start of VSync 01 - start of BACK PORCH 10 - start of ACTIVE VIDEO 11 - start of FRONT PORCH
19-18	**LcdVComp**	Generate interrupt at: 00 - start of VSync 01 - start of BACK PORCH 10 - start of ACTIVE VIDEO 11 - start of FRONT PORCH
20	**VDE**	Video DAC Enable 0 - Video DACs disabled (powered down) 1 - Video DACs enabled
21	**LcdMono8**	Lcd monochrome data width 0 - 4 bits Lcd module 1 - 8 bits Lcd module
22	**LcdPwr**	Lcd power enable 0 - Lcd is off 1 - Lcd is on when **LcdEn**=1
23	**LcdBLE**	Lcd Backlight enable This drives “0” or “1” out to the Lcd backlight enable pin.
31-24	-	Reserved

Table 11-2: Video control register VideoControl (Continued)

11.4 LCD Timing 0 Register

LCD Timing 0 Register (LcdTiming0) contains four bit-fields that are used to control horizontal LCD timing. See ***11.6.2 Pixel Clock Divider (PCD)*** on page 11-12 for a description of the terms “PixelClock” and “LcdClk”

11.4.1 Pixels-per-line (PPL)

The pixels-per-line (PPL) bit-field is used to specify the number of pixels in each line or row on the screen. PPL is a 6-bit value that represents between 16–1024 pixels-per-line. PPL is used to count the correct number of pixel clocks that must occur before the line clock can be pulsed. Program the value required divided by 16, minus 1.

11.4.2 Horizontal Sync Pulse Width (HSW)

The 6-bit horizontal sync pulse width (HSW) field is used to specify the pulse width of the line clock in passive mode, or horizontal synchronization pulse in active mode. (Program the value required minus 1.)

11.4.3 Horizontal Front Porch (HFP)

The 8-bit Horizontal Front Porch (HFP) field is used to specify the number of pixel clock periods to insert at the end of each line or row of pixels before pulsing the line clock pin. Once a complete line of pixels is transmitted to the LCD driver, the value in HFP is used to count the number of pixel clocks to wait before pulsing the line clock. HFP generates a wait period ranging from 1–256 pixel clock cycles. (Program to value required minus one.)

11.4.4 Horizontal Back Porch (HBP)

The 8-bit Horizontal Back Porch (HBP) field is used to specify the number of pixel clock periods to insert at the beginning of each line or row of pixels. After the line clock for the previous line has been negated, the value in HBP is used to count the number of pixel clocks to wait before starting to output the first set of pixels in the next line. HBP generates a wait period ranging from 1–256 pixel clock cycles (Program to value required minus one.).

Bit	Name	Description
1-0	-	Reserved
7-2	**PPL**	Pixels-per-line Number of pixels per line, divided by 16, minus 1
15-8	**HSW**	Horizontal Sync Pulse Width Number of LcdClk clock periods to pulse the line clock at the end of each line minus 1
23-16	**HFP**	Horizontal Front Porch Number of LcdClk clock periods to add to the end of a line transmission before line clock is asserted, minus 1
31-24	**HBP**	Horizontal Back Porch Number of LcdClk clock periods to add to the beginning of a line transmission before the first set of pixels is output to the display minus 1

Table 11-3: Lcd Timing Register 0

11.4.5 VGA Timing 0 Register

The VGA Timing 0 register has the same format as the LCD Timing 0 Register.

11.5 LCD Timing 1 Register

LCD Timing 1 Register (LcdTiming1) contains four bit-fields that are used to control LCD vertical timing parameters.

11.5.1 Lines Per Screen (LPS)

The Lines Per Screen (LPS) bit-field is used to specify the number of lines or rows per LCD panel being controlled. LPS is a 10-bit value which represents between 1–1024 Lines Per Screen. The register is programmed with the number of lines per screen minus 1.

11.5.2 Vertical Sync Pulse Width (VSW)

The 6-bit vertical sync pulse width (VSW) field is used to specify the pulse width of the vertical synchronization pulse in active mode, or is used to add extra dummy line clock delays between frames in passive mode. The register is programmed with the number of lines of **VSync** minus one.

11.5.3 Vertical Front Porch (VFP)

The 8-bit Vertical Front Porch (VFP) field is used to specify the number of line clocks to insert at the end of each frame. Once a complete frame of pixels is transmitted to the LCD display, the value in VFP is used to count the number of line clock periods to wait. After the count has elapsed the **VSync** (**LcdFP**) signal is pulsed in active mode, or extra line clocks are inserted as specified by the VSW bit-field in passive mode. VFP generates from 0–255 line clock cycles.This should be zero for passive display modes, unless synchronizing to the VGA to share data.

11.5.4 Vertical Back Porch (VBP)

The 8-bit Vertical Back Porch (VBP) field is used to specify the number of line clocks to insert at the beginning of each frame. The VBP count starts just after the **VSync** signal for the previous frame has been negated for active mode, or the extra line clocks have been inserted as specified by the VSW bit-field in passive mode. After this has occurred, the value in VBP is used to count the number of line clock periods to insert before starting to output pixels in the next frame. VBP generates from 0–255 extra line clock cycles. This should be programmed to zero in passive mode, unless sensing LCD to VGA to share DMA data.

Bit	Name	Description
9-0	**LPS**	Lines Per Screen Number of lines per screen. Program to number of lines required minus 1.
15-10	**VSW**	Vertical Sync Pulse Width Number of **VSync** lines. Should be small for passive LCD, but should be long enough to re-program the video palette under interrupt control, without writing the video palette at the same time as video is being displayed. Program to the number of lines required minus one.
23-16	**VFP**	Vertical Front Porch Number of inactive lines at the end of frame, before **VSync** period. Program to zero on passive displays.
31-24	**VBP**	Vertical Back Porch Number of inactive lines at the start of a frame, after **VSync** period. Program to zero on passive displays.

Table 11-4: Lcd Timing Register 1

11.5.5 VGA Timing 1 Register

The VGA timing 1 register has the same format as the LCD Timing 1 register, except that the Vertical Front Porch and Vertical Back Porch should be programmed to the number of lines minus one.

11.6 LCD Timing 2 Register

LCD Timing 2 Register (LcdTiming2) contains seven different bit-fields that are used to control various functions associated with the timing of the LCD controller.

11.6.1 Pixel Clock Source (PCS)

This bit controls the source of the pixel clock. It can either be the video bus clock, or it can be the VGA clock. Selecting the video bus clock means that the VGA clock can be used for the 48MHz clock for FastIR when using only the LCD. Selecting the VGA clock means that the LCD and the VGA can be operated from the same clock when sharing data.

11.6.2 Pixel Clock Divider (PCD)

The 5-bit pixel clock divider (PCD) field is used to select the frequency of the **LcdCP** clock signal to the LCD panel. PCD can generate a range of **LcdCP** clock frequencies from LcdClk/ 2 to LcdClk/33, where LcdClk is the clock selected by LCS. The frequency of the pixel clock for a set PCD value can be calculated using the following equation:

$$\text{PixelClock} = \frac{\text{LcdClk}}{(\text{PCD} + 2)}$$

Note that in the case of the LCD, the pixel clock is **not** the frequency of some nominal clock rate that individual pixels are output to the LCD. It is the frequency of the **LcdCP** signal. In normal mono mode (4-bit interface), four pixels are output per **LcdCP** cycle, so the PixelClock is one quarter the nominal pixel rate. In the case of 8-bit interface mono, PixelClock is one eighth the nominal pixel rate, since 8 pixels are output per **LcdCP** cycle. In the case of color, PixelClock is 0.375 times the nominal pixel rate, because $2^2/_3$ pixels are output per **LcdCP** cycle. If the LCD and VGA are operating concurrently, and sharing DMA data, then in color mode the pixel clock should normally be $^3/_8$ the VGA clock. To achieve this, PCD should be programmed to the value 0 and the skip4 bit set to “1”. The skip4 bit produces a null clock cycle (no high phase) every fourth clock cycle.

11.6.3 AC-bias Pin Frequency (ACB)

The 5-bit AC-bias frequency (ACB) field is used to specify the number of line clock periods to count between each toggle of the AC-bias pin (LcdAC). The value programmed is the number of lines between transitions, minus 1.

Note *The ACB bit field had no effect on LcdAC in active mode. The pixel clock transitions continuously in active mode and the AC Bias line is used as an output enable signal.*

11.6.4 Invert VSync (IVS)

The Invert VSync (IVS) bit is used to invert the polarity of the **LcdFP** signal. When IVS=1, **LcdFP** is active LOW. When IVS=0, **LcdFP** is active HIGH.

11.6.5 Invert Hsync (IHS)

The Invert HSync (IHS) bit is used to invert the polarity of the **LcdLP** signal. When IHS=1, **LcdLP** is active LOW. When IHS=0, **LcdLP** is active HIGH.

11.6.6 Invert Pixel Clock (IPC)

The Invert Pixel Clock (IPC) bit is used to select which edge of the pixel clock pixel data is driven out onto the LCD's data lines. When IPC=0, data is driven onto the LCD's data lines on the rising-edge of **LcdCP**. When IPC=1, data is driven onto the LCD's data lines on the falling-edge of **LcdCP**.

11.6.7 Invert Output Enable (IEO)

The Invert Output Enable (IEO) bit is used to select the active and inactive state of the output enable signal in active display mode. In this mode, the AC-bias pin is used as an enable that signals the off-chip device when data is actively being driven out using the pixel clock. When IEO=0, the **LcdAC** pin is active HIGH. When IEO=1, the **LcdAC** pin is active LOW. In active display mode, data is driven onto the LCD's data lines on the programmed edge of **LcdCP** when **LcdAC** is in its active state.

Bit	Name	Description
4-0	**PCD**	Pixel Clock Divisor Used to specify the frequency of the pixel clock based on the LCD clock (LcdCLK) frequency. Pixel clock frequency can range from LcdCLK/2 to LcdCLK/33. Pixel Clock Frequency = LcdCLK/(PCD+2).
5	**PCS**	Pixel Clock Source 0 - Video DMA bus clock 1 - VGA clock
10-6	**ACB**	AC Bias Pin Frequency Number of line clocks to count before toggling the AC Bias pin. This pin is used to periodically invert the polarity of the power supply to prevent DC charge build-up within the display. Program to value required minus 1.
11	**IVS**	Invert **VSync** 0 - **LcdFP** pin is active HIGH and inactive LOW. 1 - **LcdFP** pin is active LOW and inactive HIGH.
12	**IHS**	Invert **Hsync** 0 - **LcdLP** pin is active HIGH and inactive LOW. 1 - **LcdLP** pin is active LOW and inactive HIGH.
13	**IPC**	Invert Pixel Clock 0 - Data is driven on the LCD's data lines on the rising-edge of **LcdCP**. 1 - Data is driven on the LCD's data lines on the falling-edge of **LcdCP**.
14	**IEO**	Invert Output Enable 0 - **LcdAC** pin is active HIGH in TFT mode 1 - **LcdAC** pin is active LOW in TFT mode.
15	**SLV**	Slave mode Slave (or genlock) LCD to VGA video. The **HSync** and **VSync** are locked to the VGA timing generator. The LCD horizontal timing must be carefully programmed if sharing DMA data
25-16	**CPL**	Clocks Per Line This field specifies the number of actual clocks to the LCD panel on each line. This the number of pixels per line divided by either 1 (TFT), 4 or 8 (for mono passive), $2^2/_3$ (for color passive), minus one.
26	**BCD**	Bypass Pixel Clock Divider

Table 11-5: LCD Controller Bit Fields

Bit	Name	Description
27	**Skip4**	Skip every fourth clock HIGH period to allow color passive LCD to run with shared VGA DMA.
31-28	-	Reserved

Table 11-5: LCD Controller Bit Fields (Continued)

11.6.8 Clocks Per Line (CPL)

This is the actual number of clocks output to the LCD panel each line, minus one. This must be programmed, in addition to the PPL field in the LCD Timing 0 Register. The number of clocks per line is the number of pixels per line divided by either one, four, eight or two-and-two-thirds for TFT mode, mono 4-bit mode, mono 8-bit, or color STN mode respectively.

11.6.9 Bypass Pixel Clock Divider

Setting this bit allows an undivided LCD clock to be output on LCD. This bit should only be set for TFT mode.

11.6.10 Skip every fourth clock pulse (Skip 4)

Set this bit to “1” when running a color passive LCD with simultaneous VGA display in Shared DMA, Slave mode. This produces an irregular clock to the LCD, where every fourth clock pulse is suppressed (the clock stays LOW for one clock period). This is necessary because two-and-two-third pixels per clock, which are sent to the LCD, is not an integer multiple. This means that three clocks will be output every four clock periods. If PCD is zero, then eight pixels will be output every eight **LcdClk** periods, since the LCD CP clock will be half the frequency of **LcdClk**.

11.7 VGA Timing 2 Register

VGA Timing 2 Register (VgaTiming2) controls various functions associated with the timing and control of the VGA controller.

11.7.1 Invert Vsync (IVS)

The Invert VSync (IVS) bit is used to invert the polarity of the **VSync** signal. When IVS=1, **VSync** is active HIGH. When IVS=0, **VSync** is active LOW.

11.7.2 Invert Hsync (IHS)

The Invert HSync (IHS) bit is used to invert the polarity of the **HSync** signal. When IVS=1, **HSync** is active HIGH. When IVS=0, **HSync** is active LOW.

11.7.3 Composite VSync (CVS)

When this bit is set, the **VSync** signal outputs a composite sync comprised of **HSync** XNOR **VSync**. If the IVS bit is set, it will invert this to produce the XOR of the syncs.

11.7.4 Composite HSync (CHS)

When this bit is set, the **HSync** pin outputs the logical AND of **VSync** and **HSync**. If IHS is set it will output the NAND of the syncs.

Bit	Name	Description
0-10	-	Reserved
11	**IVS**	Invert **VSync** 0 - **VSync** is a negative edge sync. 1 - **VSync** is a positive edge sync.
12	**IHS**	Invert **Hsync** 0 - **HSync** is a negative edge sync. 1 - **HSync** is a positive edge sync.
13	**CVS**	Composite **VSync** Output XNOR of **HSync** and **VSync** on **VSync** pin (XOR if IVS is set)
14	**CHS**	Composite **HSync** Output AND of **HSync** and **VSync** on **HSync** pin (NAND if IHS is set)
31-15	-	Reserved

Table 11-6: VGA TIming 2 Register Functions

11.8 VGA Timing 3 Register

The VGA Timing 3 Register is used to program the timing of the VGA border display. It consists of 4 8-bit fields which specify the number of pixel clocks or line clocks after the start of the porch periods that the border starts or stops. The border periods run ‘concurrently’ with porch periods, so that the porch timings for LCD and VGA are the same. (There is no border on LCD). Therefore, if programming LCD and VGA with the same timings, then the border values would be the same.

11.8.1 VGA Horizontal border start register

This register contains the number of pixel clocks after the start of the horizontal back porch period, that the border display starts, minus one.

11.8.2 VGA Horizontal border end register

This register contains the number of pixel clocks after the end of horizontal data ends that the border color will be displayed for, minus one.

11.8.3 Vertical border start register

This register contains the number of lines after the vertical back porch begins that the border display starts, minus one.

11.8.4 Vertical border end register

This register contains the number of lines after the vertical front porch begins that the border will be displayed for.

Bit	Name	Description
7-0	**HBS**	Horizontal border start
15-8	**HBE**	Horizontal border end
23-16	**VBS**	Vertical border start
31-24	**VBE**	Vertical border end

Table 11-7: Horizontal and Vertical Border Registers

11.8.5 VGA border color register

Bit	Name	Description
23-0	**BCOL**	RGB 8:8:8 Border Colour

Table 11-8: VGA Border Colour Register

11.9 LCD DMA Base Address Register

The LCD DMA base address register (LcdDBAR) is a read/write register used to specify the base address of the off-chip frame buffer for the LCD. Addresses programmed in the base address register must be aligned on sixteen-word boundaries, thus the least significant six bits (LcdDBAR[5:0]) must always be written with zeros. Only 26 bits of the register are valid (including the LS 6bits which must be zero), because LCD DMA is only allowed from SDRAM.

The 26 bit address range allows the LCD DMA to access any address within the SDRAM. The upper address lines are not needed, because these are the address lines used to select which device is accessed, but the LCD always accesses SDRAM. The user must initialize the base address register before enabling the LCD, and may also write a new value to it while the LCD is enabled to allow a new frame buffer to be used for the next frame. The user can change the state of LcdDBAR while the LCD controller is active, after the Next Frame (Next) status bit is set within the LCD's status register that generates an interrupt request. This status bit indicates that the value in the base address pointer has been transferred to the current address pointer register and that it is safe to write a new base address value. This allows double-buffered video to be implemented if required.

Bit	Name	Description
5-0	-	Reserved Should always be written zero
25-6	**LcdDBAR**	LCD DMA Channel Base Address Pointer 16-word aligned base address in SDRAM of the frame buffer within off-chip memory.
31-26	-	Reserved Should be written zero

Table 11-9: LCD DMA Base Address Registers

11.9.1 VGA DMA Base Address Register

The VGA DMA base address register is the same as the LCD base address register, but for the VGA display.

11.10 LCD DMA Channel Current Address Register

This read-only register allows the processor to read the current value of the LCD DMA channel current address register. This is not something that would normally be done, but it allows additional test observability. Its value cannot be expected to be exact, it could change at an moment. However, its contents can be read to determine the approximate line that the LCD controller is currently displaying and driving out to the display

Bit	Name	Description
5-0	-	Undefined
25-6	**LcdDCAR**	LCD DMA Channel Current Address Pointer 16-word aligned current address pointer to data in SDRAM frame buffer currently being displayed.
31-26	-	Undefined

Table 11-10: LCD DMA Channel Current Address Register

11.10.1 VGA DMA Channel Current Address Register

The VGA DMA base address register is the same as the LCD base address register, but for the VGA display.

11.11 LCD Controller Status/Mask and Interrupt Registers

The LCD controller status, mask and interrupt registers all have the same format. Each bit of the status register is a status bit that may generate an interrupt. These are masked by the corresponding bits in the mask register. The interrupt register is the logical AND of the status and mask registers, and the interrupt output from the LCD controller is the logical OR of the bits within the interrupt register.

The LCD controller status register (LCSR) contains bits that signal an under-run error for the FIFO, the DMA next base update ready status, and the DMA done status. Each of these hardware-detected events can generate an interrupt request to the interrupt controller.

11.11.1 LCD Frame Done (LDone)

The LCD Frame Done (Done) is a read-only status bit that is set after the LCD has been disabled and the frame that is active finishes being output to the LCD's data pins. It is cleared by writing the base address (**LcdDBAR**) or enabling the LCD, or, by writing "1" to the LDone bit of the Status Register. When the LCD is disabled by clearing the LCD enable bit (**LcdEn**=0) in LcdControl, the LCD allows the current frame to complete before it is disabled. After the last set of pixels is clocked out onto the LCD's data pins by the pixel clock, the LCD is disabled and Done is set.

11.11.2 LCD Next Frame (LNext)

The LCD Next Frame (LNext) is a read-only status bit that is set after the contents of the LCD DMA base address register are transferred to the LCD DMA current address register, and it is cleared when the LCD DMA base address register is written.

11.11.3 FIFO Underflow Status (LFUF)

The LCD FIFO underflow status (LFUF) status bit is set when the LCD FIFO under-runs. The status bit is "sticky", meaning it remains set after the FIFO is no longer under-running. The status bit is cleared by writing a '1' to this bit of the status register.

Bit	Name	Description
0	**LFUF**	FIFO underflow status/mask/interrupt bit
1	**LNext**	LCD Next base address update status/mask/interrupt bit This status bit is set when the base address is transferred to the current address register at the start of frame
2	**VComp**	Vertical compare interrupt
3	**LDone**	LCD Done frame status/mask/interrupt bit This status bit is set when **LcdEn** has been set to '0', after the current frame completes

Table 11-11: LCD Controller Status/Mask and Interrupt Registers

11.11.4 VComp Interrupt

This bit is set when the Lcd timing generator reaches the vertical region programmed in the Video Control Register. This bit is "sticky", meaning it remains set until it is cleared by writing a "1" to this bit of the status register.

11.11.5 VGA Status/Mask/Interrupt registers

The VGA status, mask and interrupt registers are exactly the same format as the LCD registers. The only difference is that if the VGA is disabled, it is disabled immediately, rather than waiting for the end of frame. This means there is no equivalent of LDone for the VGA

11.12 LCD Palette registers

The LCD palette registers are a set of 256 word-aligned registers that allow the LCD to be programmed. The format of the palette data is shown below. The format of the data is chosen such that it is compatible with the format of the VGA palette data. In addition to the LCD palette write address area, there is also an area where the LCD and VGA palettes may be written simultaneously.

Bit	Name	Description
3:0	-	Reserved
7:4	R	Red palette data
11:8	-	Reserved
15:12	G	Green palette data
19:16	-	Reserved
23:20	B	Blue palette data
31:24	-	Reserved

Table 11-12: LCD Palette Registers

11.12.1 VGA Palette registers

The VGA palette registers provide the same functionality as the LCD palette registers, but for the VGA. The VGA palette register entries have 24 valid bits, because there are 8 bits per color gun, rather than four for the LCD. There is a read/write area for VGA registers only, and a write-only area to write both the VGA and LCD palettes simultaneously.

Bit	Name	Description
7:0	R	Red palette data
15:8	G	Green palette data
23:16	B	Blue palette data
31:24	-	Reserved

Table 11-13: VGA Palette registers

11.13 VGA Test Register

The VGA test register contains bits that allow certain VGA signals to be output on the LCD pins for test purposes. This register should not normally be used. The register is reset to all zero, and this will result in normal operation.

Bit	Name	Description
0	-	Reserved
2-1	**THDO**	Test VGA data output 00 - LCD data bus outputs normal LCD data 01 - Undefined 10 - LCD data bus outputs LOW nibble of VGA RGB data 11 - LCD data bus outputs HIGH nibble of VGA RGB data
3	**TVDATA**	Walking one's pattern used in place of SDRAM data for the VGA controller
4	**TLDATA**	Walking one's pattern used in place of SDRAM data for the LCD controller
5	**TDAC**	0 - normal VGA DAC operation 1- test VGA DAC operation: DAC Clock and power down from pins, DAC data input from LCD data bus pins
6	**TRF**	For production test of grayscaler, never write a "1" to these registers in normal use.
7	**TCF**	For production test of grayscaler, never write a "1" to these registers in normal use.
8	**TLR**	For production test of grayscaler, never write a "1" to these registers in normal use.
9	**TCR**	For production test of grayscaler, never write a "1" to these registers in normal use.
10	**TLC**	For production test of grayscaler, never write a "1" to these registers in normal use.
11	**TCC**	For production test of grayscaler, never write a "1" to these registers in normal use.
31-12	-	Reserved

Table 11-14: VGA Test Register

11.14 Grayscaler Test Registers

The registers GSFrame State, GSRow State and GS Column State are used for the purpose of production test and **must not** be written to or read from in normal use.

11.15 Video Controller Register Locations

The VideoControl register appears in two positions in the register map. It appears in the LCD area and also in the VGA area. This register allows both controllers to be enabled simultaneously. The LCD and VGA palette registers can also be read and written in their own unique address space, or written only in a combined area.

Table 11-15: LCD register map locations shows the registers associated with the LCD controller and the physical addresses used to access them.

Name	Type	Description
VideoBase+ 0x00	VideoControl	Video Control Register
VideoBase+ 0x04	LcdStatus	LCD Status Register
VideoBase + 0x08	LcdStatusMask	LCD Status Mask Register
VideoBase + 0x0C	LcdInterrupt	LCD Interrupt Register
VideoBase+ 0x10	LcdDBAR	LCD DMA Channel Base Address Register
VideoBase + 0x14	LcdDCAR	LCD DMA Channel Current Address Register
VideoBase + 0x20	LcdTiming0	LCD Timing 0 Register
VideoBase + 0x24	LcdTiming1	LCD Timing 1 Register
VideoBase + 0x28	LcdTiming2	LCD Timing 2 Register
VideoBase+ 0x44	VgaStatus	VGA Status Register
VideoBase + 0x48	VgaStatusMask	VGA Status Mask Register
VideoBase + 0x4C	VgaInterrupt	VGA Interrupt Register
VideoBase+ 0x50	VgaDBAR	VGA DMA Channel Base Address Register
VideoBase + 0x54	VgaDCAR	VGA DMA Channel Current Address Register
VideoBase + 0x60	VgaTiming0	VGA Timing 0 Register
VideoBase + 0x64	VgaTiming1	VGA Timing 1 Register
VideoBase + 0x68	VgaTiming2	VGA Timing 2 Register
VideoBase + 0x6C	VgaTiming3	VGA Timing 3 Register (Border timing)
VideoBase + 0x70	VgaBorder	VGA Border color register
VideoBase +0x80	VgaTest	VGA Test register
VideoBase +0x84	GSFrameState	Grayscaler production test register
VideoBase +0x88	GSRowState	Grayscaler production test register
VideoBase +0x8C	GSColumnState	Grayscaler production test register
VideoBase + 0x400 - VideoBase + 0x7FC	LCDPalette	LCD Palette programming registers

Table 11-15: LCD register map locations

Name	Type	Description
VideoBase + 0x800 - VideoBase + 0xBFC	VGAPalette	VGA Palette programming registers
VideoBase + 0xC00 - VideoBase + 0xFFC	LCDVGAPalette	Combined LCD **and** VGA palette **write-only** registers

Table 11-15: LCD register map locations

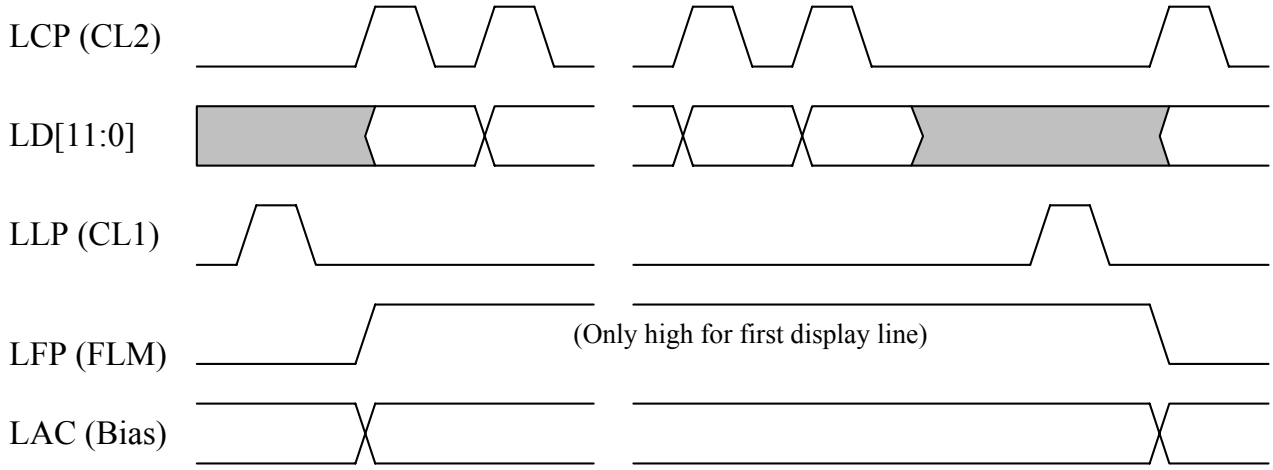


Figure 11-2: Example Mono STN LCD panel signal waveforms.

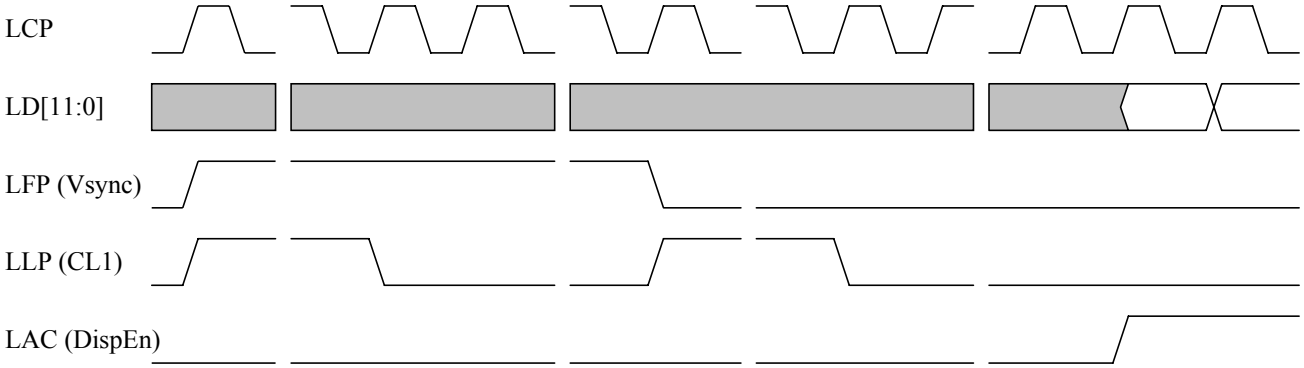


Table 11-16: Example TFT signal waveforms, start of frame.

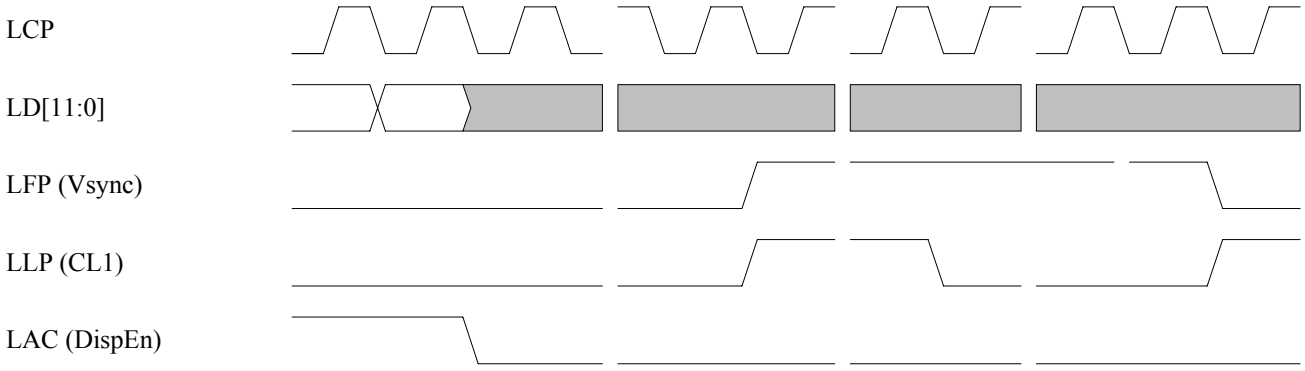


Table 11-17: Example TFT signal waveforms, end of last line.

12 Fast AMBA Peripherals

Fast AMBA Peripherals

12.1 Introduction

This chapter describes the peripherals that are connected to the 30MHz internal peripheral bus. The LCD and VGA control registers are connected to this bus. A summary of LCD and VGA features appears in ***11.1 Overview*** on page 11-2. These peripherals are supported by the DMA controller, which transfers data to/from SDRAM.

Further details are available in ***Chapter 3, Architecture Overview***.

12.2 Peripheral DMA Controller

The 7201 has an on-chip DMA controller (DMAC) that can transfer data on up to three channels. The following sections describe the Direct Memory Access unit (DMAC).

Overview

Register Descriptions

DMAC Operation

Examples of Use

12.2.1 Overview

This chip includes a three-channel direct memory access controller (DMAC). The DMAC can be used in place of the CPU to perform high-speed transfers between peripheral devices and memory space. Note the DMA controller can only transfer data to and from SDRAM. Transfers to addresses other than SDRAM will produce unpredictable results.

Features

The DMAC has the following features.

- Three Channels
- 4 Gbytes of address space on the architecture
- Max Transfer rate: 910MB/s
- Max Transfer number: 16384
- Address mode: Single address is supported.
- Channel function: Transfer mode is different in each channel.

  Channel 0

  This channel has a source address reload function, which is used by sound interface controller. The memory space of sound I/O device consists of double buffer. The sound interface uses exception bus mode and word access. Exception bus mode: when the request is active, DMAC serves only one time operation.

  Channel 1

  This channel is used by the Infrared Communication Port(ICP). The channel uses exception bus mode and word access.

  Channel 2

  This channel is used by the Universal Serial Bus(USB). The channel uses burst bus mode and word access.

- Channel priority level : Selectable fixed mode
- Interrupt request : An interrupt request can be generated to the CPU after transfers end by the specified counts.

12.2.2 Block Diagram

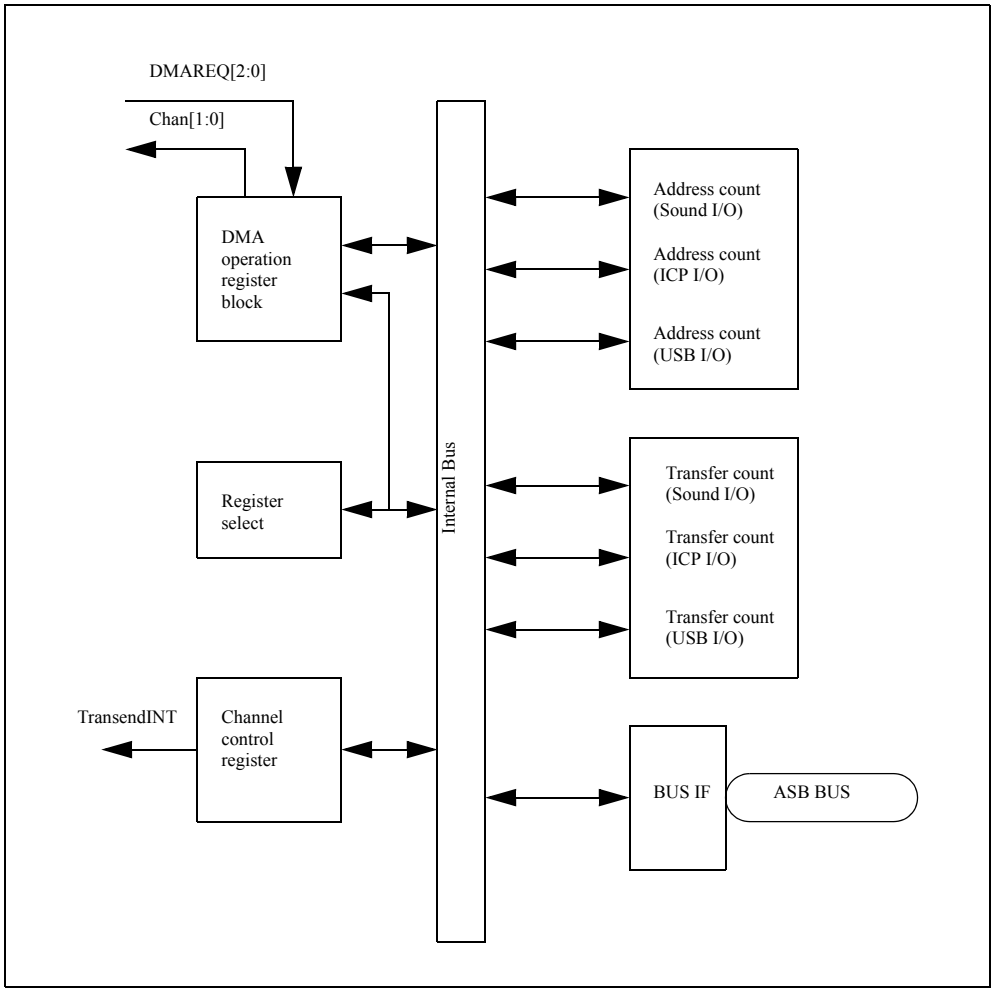


Figure 12-1: DMAC Block Diagram

12.2.3 Signal Description

The DMAC module is connected to the ASB.

Name	Type	Source/ Destination	Description
BCLK	In	Clock controller	System (bus) clock. This clock times all bus transfers. The clock has two distinct phases - phase 1 in which BCLK is LOW, and phase 2 in which BCLK is HIGH.
BnRES	In	Reset Controller	These signals indicate the reset status of the bus
BA[31:0]	InOut	ASB bus	ASB address. Output for DMAC operation.Input for Register access.

Table 12-1: ASB Signal Description

Name	Type	Source/ Destination	Description
BD[31:0]	InOut	ASB bus	This is part of the bidirectional system data bus.
AREQ	Out	Arbiter	Request signal for ASB Bus mastership.
AGNT	In	Arbiter	Bus Grant signal from ASB arbiter.
Granted	Out	APB	This signal informs APB of the granted Bus state.
BERROR	InOut	ASB bus	ASB error signal.
BLAST	InOut	ASB bus	ASB break burst signal from Slave (SDRAM Controller).
BLOCK	Out	ASB bus	ASB locked transfer signal
BPROT[1:0]	Out	ASB bus	ASB master protection information.
BSIZE[1:0]	Out	ASB bus	ASB transaction size signal
BTRAN[1:0]	Out	ASB bus	ASB transaction type signal.
BWAIT	InOut	ASB bus	ASB wait transfer signal. Input for DMA cycle stretch. Out for Register access.
BWRITE	InOut	ASB bus	ASB transfer direction signal
DSEL	In	Decoder	Register select signal
nDMREQ[2:0]	In	I/O device	DMA transfer request signal from the I/O device
Chan[1:0]	Out	APB	DMA channel select signal to APB
TransendINT	Out	CPU	DMA transfer end interrupt signal to CPU
DMATest	Out	Arbiter	This signal informs DMA Test mode during TIC test.

Table 12-1: ASB Signal Description (Continued)

12.2.4Register Configuration

Table 12-2: DMAC register summary summarizes the DMAC registers.

Address	Name	Description
DMACBase + 0x00	Ch0_start_address_buffer0	# 32-bit R/W
DMACBase + 0x04	Ch0_start_address_buffer1	# 32-bit R/W
DMACBase + 0x08	Ch0_transfer_num_buffer0	# 32-bit R/W
DMACBAse + 0x0C	Ch0_transfer_num_buffer1	# 32-bit R/W
DMACBase + 0x10	Ch0_control	# 32-bit R/W
DMACBase + 0x14	Ch1_start_address	# 32-bit R/W
DMACBase + 0x18	Ch1_transfer_number	# 32-bit R/W
DMACBase + 0x1C	Ch1_control	# 32-bit R/W
DMACBase + 0x20	Ch2_start_address	# 32-bit R/W
DMACBase + 0x24	Ch2_transfer_number	# 32-bit R/W
DMACBase + 0x28	Ch2_control	# 32-bit R/W
DMACBase + 0x2C	Interrupt_flag	# 32-bit R/O
DMACBase + 0x30	test_register0	# 32-bit R/W
DMACBase + 0x34	test_register1	# 32-bit R/O
DMACBase + 0x38	test_register2	# 32-bit R/O
DMACBase + 0x3C	DMA_operation	# 32-bit R/W

Table 12-2: DMAC register summary

The address register, transfer number register and channel control register of DMA channel 0.

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x00	R/W	ADR0	[31–0]	0x0	DMAC 0 Source Address (Buffer 0 Address) This channel transfers data from External Memory to the Sound Interface block

Table 12-3: DMAC0 Registers

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x04	R/W	ASR	[31:0]	0x00...00	DMAC 0 Sound Address (Buffer 1 Address) This channel transfers data from External Memory to Sound Interface block.This value will be automatically reloaded.
DMA Base + 0x08	R/W R	TNR0	[13:0] [31:14]	0x3FFF 0x00000	DMAC0 Transfer Number (Maxmium Transfer Number of Buffer0 is 0x4000 word) Reserved bits
DMA Base + 0x0C	R/W R	TSR	[13:0] [31:14]	0x3FFF 0x00000	DMAC0 Sound Transfer Number (Maximum Transfer Number of Buffer1 is 0x4000 word) This value will be automatically reloaded. Reserved bits
DMA Base + 0x10	 R/W R	 CCR0	[0] [1] [2] [31:3]	0 0 0 0x00	DMEN (DMAC0 enable bit) MASK0(Buffer 0 transfer end interrupt mask bit) MASK1(Buffer 1 transfer end interrupt mask bit) Reserved bits

Table 12-3: DMAC0 Registers (Continued)

12.2.5 DMAC1 Registers

This section gives the address register and channel control registers of DMA channel 1.

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x14	R/W	ADR1	[31:0]	0x00...00	This value is start address of ICP RX buffer.
DMA Base + 0x18	R/W	TNR1	[13:0] [31:14]	3FFF 0x00000	Maximum Transfer Number of DMAC1 is 0x4000 word Reserved bits
DMA Base + 0x1C	R/W	CCR1	[0] [1] [2] [31:3]	0 0 0 0x0000	DMEN1 (DMAC1 enable bit) MODSEL When LOW, transfer from memory to I/O. When HIGH, transfer from I/O to memory. Reserved bits MASK (The mask bit of transfer end interrupt for ICP)

Table 12-4: DMAC1 Registers

12.2.6 DMAC 2 Registers

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x20	R/W	ADR2	[31:0]	00x00	This value is the start address of DMA Channel for USB Controller
DMA Base + 0x24	R/W	TNR2	[13:0] [31:14]	3FFF 0x0000	Maximum Transfer Number of DMAC2 is 0x4000 word Reserved bits
DMA Base + 0x28	R/W	CCR2	[0] [1] [2] [31:3]	0 0 0 0x0000	DMEN MODSEL When LOW, transfer from memory to I/O. When HIGH, transfer from I/O to memory. MASK Reserved bits

Table 12-5: DMAC2 Registers

12.2.7 Interrupt flag registers

The interrupt flag registers of DMAC

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x2C	R/W	FLAGR	[0]	0	FLAG0 (Buffer 0 transfer end flag bit for Channel 0) Cleared by writing 1.
			[1]	0	FLAG1 (Buffer 1 transfer end flag bit for Channel 0) Cleared by writing 1.
			[2]	0	FLAG2 (Transfer end flag bit for Channel 1) Cleared by writing 1.
			[3]	0	FLAG3 (Transfer end flag bit for Channel 2) Cleared by writing 1.
	R		[31:4]	0x000000	Reserved bits

Table 12-6: Flag Register

12.2.8 Test register 0 of DMAC

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x30	R/W	TESTR0	[0]	0	Test internal mode
			[1]	0	Test external mode
			[2]	0	Test request of channel 0
			[3]	0	Test request of channel 1
			[4]	0	Test request of channel 2
			[5]	0	Carryin bit for testing the address counter of channel 0
			[6]	0	Carryin bit for testing the address counter of channel 1
			[7]	0	Carryin bit for testing the address counter of channel 2
			[8]	0	Carryin bit for testing the transfer counter of Channel 0
			[9]	0	Carryin bit for testing the transfer counter of Channel 1
			[10]	0	Carryin bit for testing the transfer counter of Channel 2
	R		[31:11]	0x000000	Reserved bit

Table 12-7: Test Register 0

12.2.9 DMAC (Read only) Test Register1

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x34	R/W	TESTR1	[31:0]	0	When in test mode, this register latches the address of DMAC.

Table 12-8: DMAC Test Register 1

12.2.10DMAC (Read only) Test Register2

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x38	R/W R	TESTR2	[0] [1] [3:2] [5:4] [6] [8:7] [31:9]	0 0 0 0 0 0 0x000000	BWRITE DMAC request signal BSIZE[1:0} BTRAN[1:0] TransendINT (the transfer-end interrupt of DMAC) ChSel[1:0] Reserved bits

Table 12-9: Test Register 2

12.2.11DMAC Operation Register

Address	R/W	Name	Bit	Initialized to	Contents
DMA Base + 0x3C	R/W R	DMAOR	[0] [1] [2] [31:3]	0 0 0 0x0000	DMAEN (DMAC master enable bit) PRMD0 (Priority mode bit 0) PRMD1 (Priority mode bit 1) Reserved bits

Table 12-10: Test Register 2

12.2.12Register Descriptions

DMA address registers (ADR0–2, ASR)

DMA address registers are 32 bits read/write registers that specify the address of a DMA transfer. Initial value is 0x00000000.

DMA Transfer Number register (TNR0–2, TSR)

DMA Transfer number registers are 32 bits read/write registers that specify the DMA transfer number.

Initial value is 0x3FFF.

DMA Channel Control register (CCR0–2)

DMA Channel Control registers are 32 bits read/write register that specifies operation mode in each channel.

Channel Control Register 0(CCR0)

This register is channel control register of the interface controller for sound peripheral device.

Bit 0 (DMEN: Enables channel operation)

DMEN	Description
0	Disables channel operation (initial value)
1	Enables channel operation

Table 12-11: Bit 0

Bit 1 (MASK0: The mask bit of transfer end interrupt for buffer 0)

MASK0	Description
0	Interrupt request is not generated even if data transfer ends by the specified count (initial value)
1	Interrupt request is generated if data transfer ends by the specified count

Table 12-12: Bit 1

Bit 2 (MASK1: The mask bit of transfer end interrupt for buffer 1)

MASK 1	Description
0	Interrupt request is not generated even if data transfer ends by the specified count (initial value)
1	Interrupt request is generated if data transfer ends by the specified count

Table 12-13: Bit 2 Mask 1

Channel Control Register 1(CCR1)

This register is channel control register of Infrared Communication Port(ICP) controller.

Bit 0 (DMEN: Enables channel 1 operation)

DMEN	Description
0	Disables channel 1 operation (initial value)
1	Enables channel 1 operation

Table 12-14: Bit 0

Bit 1 (MODSEL: Mode select bit)

MODSEL	Description
0	DMAC transfers data from memory to I/O buffer
1	DMAC transfers data from I/O buffer to memory

Table 12-15: Bit 1

Bit 2 (MASK: The mask bit of transfer end interrupt for receive buffer)

MASK	Description
0	Interrupt request is not generated even if data transfer ends by the specified count (initial value)
1	Interrupt request is generated if data transfer ends by the specified count

Table 12-16: Bit 2

Channel Control Register 2(CCR2)

This register is the channel control register of the Universal Serial Bus interface controller.

Bit 0 (DMEN: Enables channel 2 operation)

DMEN	Description
0	Disables channel 2 operation (initial value)
1	Enables channel 2 operation

Table 12-17: Bit 0

Bit 1 (MODSEL: Mode select bit)

MODSEL	Description
0	DMAC transfers data from memory to I/O buffer
1	DMAC transfers data from I/O buffer to memory

Table 12-18: Bit 1

Bit 2 (MASK: The mask bit of transfer end interrupt for receive buffer)

MASK	Description
0	Interrupt request is not generated even if data transfer ends by the specified count (initial value)
1	Interrupt request is generated if data transfer ends by the specified count

Table 12-19: Bit 2

DMA Test Register

The DMA Test Register is a 32 bits read/write register that tests the DMA Master. Initial value: 0x00.

DMA Operation Register (DMAOR)

The DMA Operation Register is a 32 bits read/write register that controls the DMA Master. Bit 0 (DMAEN: Enables or disables DMA transfers on all channels).

DMAEN	Description
0	Disable DMA transfer on all channels (initial value)
1	Enable DMA transfer on all channels

Table 12-20: DMA Operation Register (DMAOR)

Bit1 (PRMD0–1): Select the priority level between channels when there are transfer requests for multiple channels simultaneously.

PRMD1	PRMD0	Description
0	0	Ch1 > Ch2 > Ch0 (initial value)
0	1	Ch1 > Ch0 > Ch2
1	0	Ch2 > Ch1 > Ch0
1	1	Ch0 > Ch1 > Ch2

Table 12-21: Bit 1 (PRMD0-1)

12.2.13DMAC Operation

When there is a DMA transfer request, the DMAC starts the transfer according to the predetermined channel priority order. If the transfer end conditions are satisfied, it ends the transfer.

12.2.14DMA Transfer Flow

After the DMA address register (ADR, ASR), DMA transfer number register (TNR), DMA channel control register (CCR), and DMA operation register (DMAOR) are set, the DMAC transfers data according to the following procedure.

- See if the DMEN bit of CCR and the DMAEN of DMAOR are enabled.
- When a transfer request comes and transfer condition is enabled, the DMAC transfers data according to bus size, address mode and bus mode.
- When the specified number of transfer have been competed (TNR = count value), the transfer ends normally. If the MASK bit of the CCR is set to 1 at this time, the DMA transfer end interrupt is sent to the CPU.

DMA Channel Priority

When the DMAC receives simultaneous transfer requests, it selects a channel according or a predetermined priority order. The priority order is selected by the priority order select bits, PRMD0 and PRMD1, in the DMA operation register.

DMA bus mode

Burst mode

Once the bus mastership is obtained, the transfer is performed continuously until the transfer end condition is satisfied. however, when the nDMREQ pin is driven high, the bus passes to the other bus master after current cycle ends. DMA request is nDMREQ level detection.

nDMREQ

CPU

CPU

DMA

DMA

DMA

DMA

CPU

Exception mode

In the exception mode, the bus mastership is given to another bus master after a one-transfer-unit

DMA transfer.

The DMA request should be disabled by I/O device module.

DMA request is nDMREQ level detection.

nDMREQ

CPU

DMA

CPU

CPU

DMA

CPU

CPU

12.2.15Examples of Use

DMA transfer between Sound I/O device and Memory.

In the following example, data is transferred from Memory to Sound I/O device.
The transfer is performed by DMAC channel 0.

Transfer conditions	Register	Setting value
Transfer source: Memory	ADR0 or ASR	Memory address value
The number of transfer:10	TNR0	0x000A
Source address: Increment		
Interrupt enable	CCR0	0x07
Bus mode: exception		
Transfer size: word		
Channel priority order: CH 1> CH 2 > CH 0	DMAOR	0x01

Table 12-22: DMA transfer between Sound I/O device and Memory

DMA transfer between the transmit buffer of ICP and Memory

The transfer is performed by DMAC channel 2.

Transfer conditions	Register	Setting value
Transfer destination: Memory	ADR2	Memory address value
The number of transfer:10	TNR0	0x000A
HSSP mode and enables interrupt	CCR0	0x016
Bus mode: burst		
Transfer size: word		
Channel priority order: CH 2 > CH 1 >CH 0	DMAOR	0x05

Table 12-23: DMA transfer between the transmit buffer of ICP and Memory

12.3 Medium and Fast Infrared Module

12.3.1 Overview

The ARM 7201 Infrared Interface Module implements in hardware the physical layer of an infrared serial port, compliant with Version 1.1 of the Infrared Data Association (IrDA) standard. Communication speeds of up to 4Mbit/s are supported. When combined with analogue transducer components, it provides a complete interface between infra-red media and an AMBA-compliant peripheral bus (APB).

The module comprises three separate encoder/decoder units for implementing three different combinations of modulation scheme and data encoding system defined by the IrDA standard. These are called the Slow-, Medium- and Fast-infrared encoder/decoders.

SIr, MIr and FIr

The Slow Encoder/Decoder (SIr) is used to modulate and demodulate serial data only, using the Hewlett-Packard Serial Infrared standard (HP-SIR) for bit encoding. Serial transmit data from a UART (external to the Ir Interface Module) is modulated using return-to-zero (RTZ) encoding to produce an output to drive the IR transmitter LED, while data received from the IR detector is converted into a serial bit stream to drive the UART serial input. The SIr supports data rates of up to 115.2kbit/s.

The Medium Speed Encoder/Decoder (MIr) encodes/decodes peripheral bus data according to a modified HDLC standard, using flag characters, bit-stuffing and a 16-bit CRC checker. RTZ modulation and demodulation of the encoded data stream takes place in the same way as for the SIr. Two signal bit rates are specified: 0.576Mbit/s and 1.152Mbit/s. MIr data and control bytes are memory mapped via the ARM 7201 fast APB.

The Fast Speed Encoder/Decoder (FIr) operates at a fixed bit rate of 4Mbit/s. Modulation/ demodulation is by a phase shift key scheme called pulse position modulation (4PPM) that uses one of four signalling symbols to represent each pair of data bits. Data encoding uses a packet format that prefixes bit- and symbol-synchronization flags to data and appends a 32-bit CRC and stop flag to the end of each packet. The start and stop flags use signalling symbols that are not used to encode data and hence bit-stuffing of data is not required in this mode. The FIr also interfaces to the ARM 7201 fast APB.

Only one of the Encoder/Decoder modules can be enabled to transmit and receive data from the IrDA transducers at one time. Selection of an Ir sub-module is by means of the IrEnable register.

Accordingly the MIr and FIr sub-modules can be regarded by programmers as independent entities which are operated using common control and data registers, but which report status data via separate read registers.

Detailed descriptions of the MIr and FIr are given in the following sections. The SIr however has no data or control register associated with it, and interfaces directly to the UART serial streams. Accordingly with the exception of the Ir Enable register, it has no presence on the memory map nor an interface to the APB.

12.3.2 Common Register Description

The Infrared Interface module uses a single mode register, IrEnable, to select which infrared interface module is selected. The Medium and Fast modules share a common control and data interface while maintaining separate status registers. The common registers used by both the FIr and MIr blocks are described below.

12.3.3 IrEnable Register

The register IrEnable selects which of the three Ir sub-modules (SIr, MIr and FIr) is used to operate the IrDA interface. Only one of the three may be active at any one time. The reset value for this register is zero which disables all three encoder/decoder modules. The bottom two bits of this register select the encoder decoder module according to the tabulated values listed below.

IrEnable Value (EN1,EN0)	Encoder Selected	Data Rate (Mbit/s)	Modulation Scheme	Data Interface
00	None	-	-	-
01	SIr	0–0.1152	NRZ + HP-SIR	Serial port
10	MIr	0.576 or 1.152	NRZ + HDLC	Fast APB
11	FIr	4.0	4PPM	Fast APB

Table 12-24: Ir Interface Mode Selection

This register may also be used to enable a hardware loopback mode for the sub-module selected.

Loop Back Mode (LBM)

The loop back mode (LBM) bit is used to enable and disable the ability of the Ir transmit output to be fed back into the receive logic for diagnostic purposes. When LBM=0, the selected Ir module operates normally. The transmit and receive data paths are independent and communicate via their respective pins. When LBM=1, the output of the transmit serial shifter is directly connected to the input of the receive serial shifter internally.

Note that even though the IrDA standard only permits half-duplex operation, this implementation does not restrict the user from transmitting and receiving data at the same time; both are fully independent units.

Transmitter Disabled Bits (FD/MD)

Two read-only status bits are provided within this register: FD and MD. When set these bits indicate that the FIr/MIr transmit module has completed transmission of the current frame and that it is safe to disable the module using bits EN1 and EN0. This feature simplifies clean switching between IrDA formats.

IrEnable Register:

Figure 12-2: Ir Enable register bits shows the register bits in Ir Enable.

Address: 0h80011000 IrEnable Read/Write

Bit	7	6	5	4	3	2	1	0
	Res.	Res.	Res.	FD	MD	LBM	EN1	EN0
Reset	0	0	0	1	1	0	0	0

Figure 12-2: Ir Enable register bits

12.3.4 Ir Control Register

The Ir Control Register (IrCon) contains seven different bit-fields which control various functions for the MIr and FIr.

Baud Rate Divisor (BRD)

The 1-bit baud rate divisor (BRD) field is used to select the baud or bit rate of the MIr. Two different baud rates can be selected:0.576Mbit/s and 1.152Mbit/s. The baud rate generator uses the 48MHz clock generated by the on-chip PLL, divided down to the current data rate as defined by the BRD bit. The receive baud clock is synchronized with the data steam each time a positive edge transition is detected on the receive data line.

Thus:

- when BRD=0, MIr operates at 0.576Mbit/s
- when BRD=1, MIr operates at 1.152Mbit/s

This bit has no effect when the FIr is selected.

Transmit buffer Underrun Select (TUS)

The transmit buffer underrun select (TUS) bit is used both to select what action to take as a result of a transmit buffer underrun, as well as mask or enable the transmit buffer underrun interrupt.

When TUS=0, transmit buffer underruns are used to signal the transmit logic that the end of the frame has been reached. When the transmit buffer experiences an underrun, the CRC value which is calculated continuously on outgoing data is loaded to the serial shifter and transmitted, followed by the stop flag and SIP pulse. Also when TUS=0, the transmit buffer interrupt is masked and the state of the transmit buffer underrun (TUR) status bit is ignored by the interrupt controller.

When TUS=1, transmit buffer underruns are used to signal the transmit logic that the end of the frame has not yet been reached and that the rate in which data is supplied to the transmit buffer is not sufficient. When the transmit buffer experiences an underrun zeroes are continuously output by the transmitter to signal an abort condition until data is once again available within the transmit buffer, and the CRC value is discarded. Additionally, when TUS=1, the transmit buffer underrun interrupt is enabled, and whenever TUR is set (one) an interrupt request is made to the interrupt controller. To change the state of this bit during operation, the user should fill the transmit buffer to ensure TUS is not written at the same time the transmit buffer underruns. Note that programming TUS=0 does not affect the current state of TUR or the transmit buffer logic's ability to set and clear TUR, it only blocks the generation of the interrupt request.

TUS is useful for ensuring that frames are not prematurely ended due to an unexpected transmit buffer underrun. At the start of a frame the user may configure TUS=1 such that any underrun signals an abort to the off-chip receiver. Just before the end of the frame the user may then configure TUS=0, allowing the remaining data to be output by the transmit logic. The buffer then underruns, causing the CRC, stop flag, and SIP to be transmitted.

Transmit Enable (TXE)

The transmit enable (TXE) bit is used to enable and disable the MIr/FIr transmit sub-module selected using the IrEnable register. When TXE=0, the transmit logic is disabled and its clocks are turned off to conserve power. When TXE=1, the transmitter logic is enabled for IrDA transmission. It is required that the user first program all other control bits before setting TXE. If the TXE bit is cleared to zero while the is actively transmitting data, transmission is stopped immediately, all data within the transmit buffer and serial output shifter is cleared. Clearing TXE to zero ensures the transmitter is disabled. Note that TXE is ignored by the SIr (which is always enabled whenever selected by the IrEnable register).

Also note that even though the IrDA standard only permits half-duplex operation, this implementation does not restrict the user from transmitting and receiving data at the same time; both are fully independent units. This function is particularly useful when using loop back mode, described above.

Receive Enable (RXE)

The receive enable (RXE) bit is used to enable or disable MIr/FIr receive operation. When RXE=0, the receive logic is disabled and its clocks are turned off to conserve power. When RXE=1, the receiver logic is enabled for IrDA reception. It is required that the user first program all other control bits before setting RXE. If the RXE bit is cleared to zero while the Ir interface is actively receiving data, reception is stopped immediately, all data within the receive buffer and serial input shifter is cleared. Clearing RXE to zero ensures the selected Ir receiver is disabled. Note that RXE is ignored by the SIr (which is always enabled whenever selected by the IrEnable register).

Also note that even though the IrDA standard only permits half-duplex operation, the FIr does not restrict the user from transmitting and receiving data at the same time; both are fully independent units. This function is particularly useful when using the FIr in loop back mode.

Receive buffer Interrupt Mask (RIM)

The receive buffer interrupt mask (RIM) bit is used to mask or enable the receive buffer service request interrupt. When RIM=0, the interrupt is masked, and the state of the receive buffer service request (RFS) bit within MIr/FIr status register 0 is ignored by the interrupt controller. When RIM=1, the interrupt is enabled, and whenever RFS is set (one) an interrupt request is made to the interrupt controller. Note that programming RIM=0 does not affect the current state of RFS or the receive buffer logic's ability to set and clear RFS, it only blocks the generation of the interrupt request.

Also note that RIM does not affect generation of the receive buffer DMA request which is asserted whenever both RFS is set and the receiver buffer error/end flag (EIF) is clear.

Transmit buffer Interrupt Mask (TIM)

The transmit buffer interrupt mask (TIM) bit is used to mask or enable the transmit buffer service request interrupt. When TIM=0, the interrupt is masked and the state of the transmit buffer service request (TFS) bit within MIr/FIr status register 0 is ignored by the interrupt controller. When TIM=1, the interrupt is enabled, and whenever TFS is set (one) an interrupt request is made to the interrupt controller. Note that programming TIM=0 does not affect the current state of TFS or the transmit buffer logic's ability to set and clear TFS, it only blocks the generation of the interrupt request.

TIM does not affect generation of the transmit buffer DMA request which is asserted whenever TFS is set.

Address Match Enable (AME)

The address match enable (AME) bit is used to enable or disable the receive logic from comparing the address programmed in the address match value (AMV) bit-field, to the address of all incoming frames. When AME is set (equals one), data is stored in the receive buffer only for those frames which have addresses that match AMV, and for any frame which contains the broadcast address (0hFF). For frames in which the address does not match, the data and CRC are ignored, and the receiver resumes hunting for another data packet. When AME is clear (zero), address values are not compared and the data in every frame is stored in the receive buffer.

Figure 12-3: Location of bits within Ir Control Register shows the location of the bits within the Ir Control Register. All bits are cleared (set to zero) following a reset of the ARM 7201. Note that the currently selected Ir interface (MIr or FIr) must be disabled by clearing the enable bits in this register (RXE=TXE=0) before selecting a different interface using the IrEnable register. The other bits in this register may be written while the interface is enabled to allow various modes to be changed during operation.

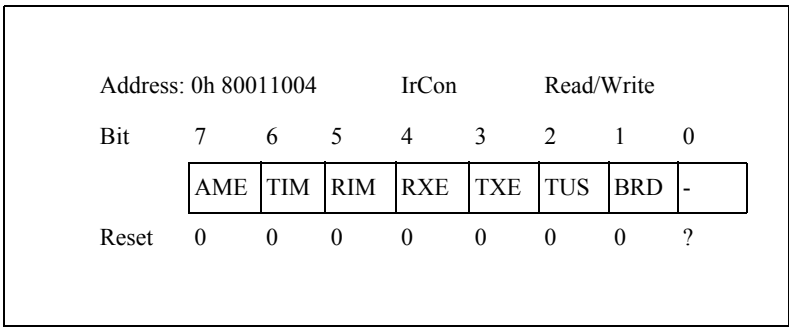


Figure 12-3: Location of bits within Ir Control Register

Bit	Name	Description
0		Unused
1	BRD	MIr bit rate select 0 - MIr data rate is 0.576 Mbit/s 1- Mir data rate is 1.152 MBit/s
2	TUS	Transmit buffer Underrun Select 0 - Transmit buffer underrun causes CRC, stop flag, and SIP to be transmitted, and masks interrupt generation (TUR ignored) 1 - Transmit buffer underrun causes an abort to be transmitted, and generates an interrupt (state of TUR sent to interrupt controller)
3	TXE	Transmit Enable 0 - Ir transmit logic disabled 1 - Ir transmit logic enabled
4	RXE	Receive Enable 0 - Ir receive logic disabled 1 - Ir receive logic enabled
5	RIM	Receive buffer Interrupt Mask 0 - Receive buffer half-full or more condition does not generate an interrupt (RFS bit ignored) 1 - Receive buffer half-full or more condition generates an interrupt (state of RFS sent to interrupt controller)
6	TIM	Transmit buffer Interrupt Mask 0 - Transmit buffer half-full or less condition does not generate an interrupt (TFS bit ignored) 1 - Transmit buffer half-full or less condition generates an interrupt (state of TFS sent to interrupt controller)
7	AME	Address Match Enable 0 - Disable receiver address match function, store data from all incoming frames in receive buffer 1 - Enable receiver address match function, do not buffer data unless address recognized or incoming address contains all ones (0hFF)

Table 12-25: Ir Control Register

12.3.5 Ir Address Match Value Register

The Ir Address Match Value Register (IrAmv) contains the 8-bit address match value field which is used by the MIr and FIr to selectively filter out unwanted received frames.

Address Match Value (AMV)

The 8-bit address match value (AMV) field is programmed with an address value which is used to selectively store only the data within receive frames which have the same address value. The address match enable (AME) bit in IrCon must be set to enable this function. For incoming frames which have the same address value as the AMV field, the frame's address, control and data is stored in the receive buffer. For those that do not, the remainder of the frame is ignored, and the receive logic searches for the beginning of the next frame in the received data stream.

A broadcast address exists which is always matched by the address match logic regardless of the value programmed in AMV. When address matching is enabled, any time a frame is received with an address containing all ones (FFh), the value programmed in AMV is ignored and the frame data is automatically stored in the receive buffer.

The address value is contained within the first byte of data in a frame following the flag. AMV can be written at any time, and is used for comparison for the next frame which occurs following its update.

Figure 12-4: Address match value field in the IrAmv Register shows the address match value field within Ir Address Match Value Register. The reset state of AMV is unknown and must be initialized before enabling the Ir interface. Note that IrAmv may be written while an Ir interface is enabled to allow the address match value to be changed during active receive operation.

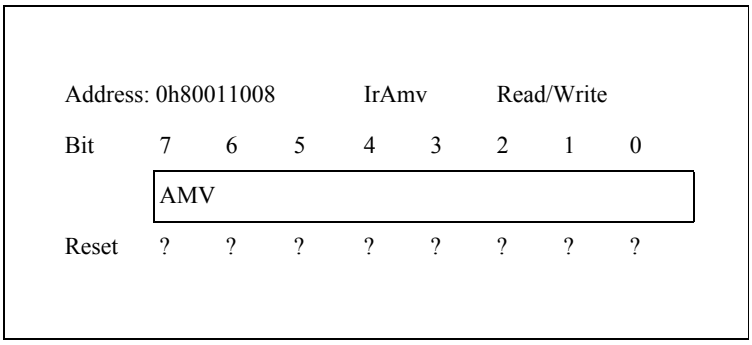


Figure 12-4: Address match value field in the IrAmv Register

Bit	Name	Description
7–0	AMV	Address Match Value 8-bit value used by receiver logic to compare to address of incoming frames. If address matches store frame address, control and data in receive buffer; if address does not match, ignore frame and search for preamble. Note: an address of 0hFF (all 1) in the incoming frame automatically generates a match (AMV is ignored).

Table 12-26: IrAmv Register

12.3.6 Ir Data Register

The Ir data register (IrData) is a 32-bit register corresponding to the transmit and receive buffers used by the MIr and FIr interfaces.

Receive Data FIFO

When IrData is read, the lower 32 bits of the bottom entry of the 37-bit two-stage receiver buffer are accessed. Bits 33-36 are used as tags to indicate various conditions which occur during reception of each piece of data. The tag bits are transferred down to the buffer along with the data word which encountered the condition. Bit 32 of the buffer is automatically transferred to the end of frame (EOF) flag, bit 33 to the CRC error (CRE) flag, and bit 34 to the receiver overrun (ROR) flag, all within MIr/FIr status register 1. Bits 35 and 36 indicate whether the received data word contains less than four valid data bytes, as occurs on the last word of a received packet that is not an integer multiple of four bytes long. The user can read these flags to determine if the value at the bottom of the buffer represents the last word within the frame and/or encountered an error during reception. After checking the flags, the buffer value can then be read.

The end/error in FIFO (EIF) status bit is set within status register 0 whenever one or more of the tag bits (32-36) are set within the receive buffer. When EIF is set, an interrupt is generated and the receive buffer DMA request is disabled so that the user can manually empty the buffer, checking the end of frame, CRC error, and overrun error flags in status register 1 first before removing each data value from the buffer. After the buffer is flushed, the user can re-enable DMA servicing by clearing the EIF bit.

Transmit Data FIFO

When IrData is written, the transmit buffer is accessed. Data is removed from the buffer one piece at a time by the transmit logic. Unlike the receive data FIFO, the transmit data FIFO may only contain32-bit words. In order to transmit a frame containing a non-integer number of words (multiple of four bytes) the Ir Data Tail Register must be used to store the final one, two or three bytes of the frame (see ***12.3.7 Ir Data Tail Register*** on page 12-24).

Figure 12-5: Bit locations within the Ir Data Register shows the bit locations corresponding to the data field, end of frame bit, as well as the cyclical redundancy check and receiver overrun error bits within the Ir Data Register. Note that both buffers are cleared when the ARM 7201 is reset. Additionally the transmit buffer is cleared whenever the TXE bit is written with a zero in IrCon, and the receive buffer is cleared whenever the RXE bit is written with a zero.

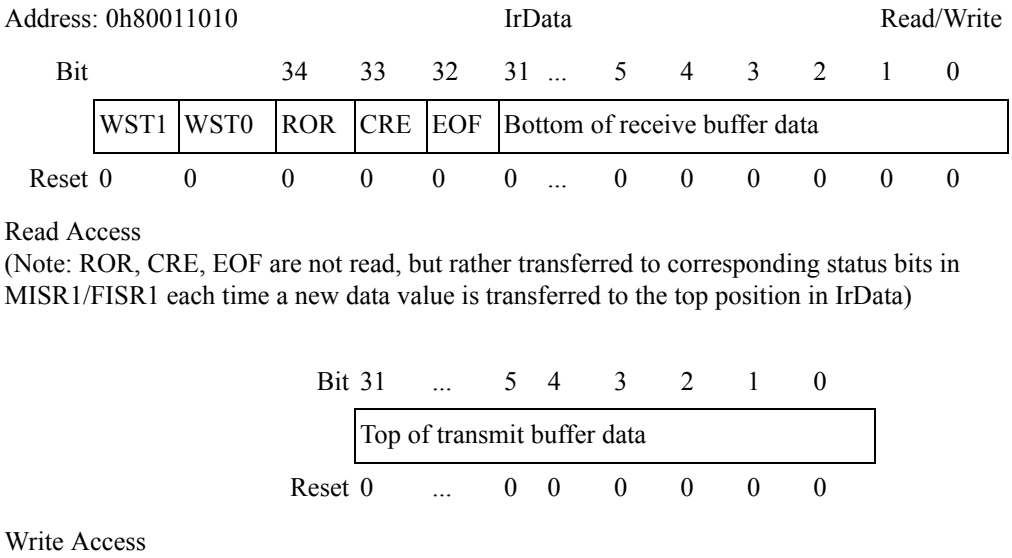


Figure 12-5: Bit locations within the Ir Data Register

12.3.7 Ir Data Tail Register

The Ir Data Tail Register (IrDataTail) is a 24-bit write-only register used for transmitting frames whose payload data is not an integral multiple of four bytes long.

IrDataTail may be written to using one of three addresses. Bits two and three of the address determine how many bytes within the word written are significant, i.e. are intended for transmission. If none of these addresses is written to the register remains marked as empty and payload data will be read by transmit logic from the 32-bit FIFO only. The status of this register does not affect the TFS flag, nor does it cause interrupts or DMA requests to be generated.

Address	Bytes to be transmitted	Bits
0h80011014	least significant byte in word only	bits 0–7
0h80011018	least significant two bytes in word only	bits 0–15
0h8001101C	least significant three bytes in data word	bits 0–23

Table 12-27: Data Tail Register addresses and bytes to be transmitted

Data is removed from this register by the transmit logic once the main transmit FIFO (IrData) is empty. Provided that the transmit FIFO does not underrun prematurely, data from this register will form the last data bytes in a frame. The transmit logic will only begin to terminate a frame -- using either a CRC or abort sequence -- once both the transmit FIFO and the IrDataTail register are empty.

Figure 12-6: Bit locations within the Ir Data Tail Register The tail register is cleared whenever it is read by the transmit logic or the TXE bit in IrCon is cleared.

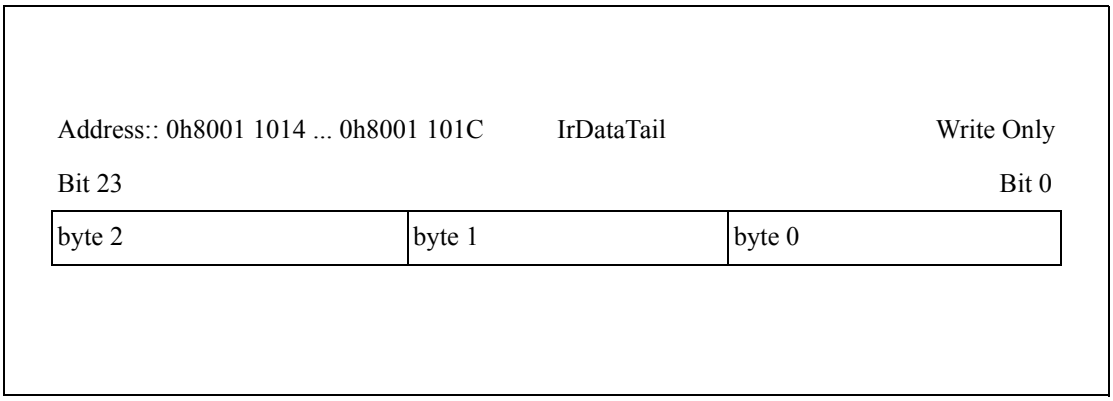


Figure 12-6: Bit locations within the Ir Data Tail Register

12.4 General Configuration

This section gives a programmer's guide to operating the IrDA interface.

12.4.1 Select Ir Mode (SIr/MIr/FIr)

The IrEnable register selects which of the three Ir sub-modules is used to operate the IrDA interface. Only one of the three may be active at any one time. The reset value for this register is zero which disables all three encoder/decode modules. The bottom two bits of this register select the encoder decoder module according to the tabulated values listed in ***Table 12-28: Bit values to select Ir module***

IrEnable EN1	IrEnable EN0	Encoder selected
0	0	None
0	1	SIr
1	0	MIr
1	1	FIr

Table 12-28: Bit values to select Ir module

12.4.2 SIr Operation

SIr does not use the data transfer mechanism implemented for MIr and FIr described in this section. After selecting SIr mode, all data transfer operations are made through UART 2 as if connection is through a serial cable without handshake lines. For details of this operation see of ***Chapter 13, 13.2.2 Features*** where operation of the UART is explained.

12.4.3 Select Data Rate

The data rates for MIr and FIr are as follows:

MIr

Clear BRD bit in IrCon for 0.576 Mbit/s

Set BRD bit in IrCon for 1.152 Mbit/s

FIr

Fixed at 4 Mbit/s

12.5 Transmitting Data

This section deals with the initialization and transmit processes.

12.5.1 Initialization

The principal transfer of data from memory to the active IrDA encoder in MIr and FIr modes is by DMA. DMA transfers data in 4-byte words into the transmit FIFO when requested by the Ir block. As data frames are not necessarily a multiple of four bytes in length, there is a mechanism built into the Ir peripheral for handling any bytes left over at the end of the data frame. This uses a register called IrDataTail. Before starting Ir transfer, pushing the first two words of data into the transmit FIFO will reduce system overheads.The Ir peripheral can be configured to provide an additional level of error checking using the TUS bit in the IrCon register. This monitors the outgoing data flow and transmits an abort signal to the far end receiver if the transmit FIFO becomes empty before the end of the frame is reached.

12.5.2 The transmit process

This section describes the transmission process in detail.

Check that the previous transmission is complete

Ensure that the Ir block is not currently receiving or transmitting data by reading the RSY (if half-duplex communications are in operation) and TBY bits in IrMISR1 or IrFISR1 (depending on whether MIr or FIr mode is in operation). If either are set, the start of transmission is postponed.

Select Transmit Underrun Action

Set the TUS bit within the IrCon register to ensure that if the Ir encoder is starved of data it will signal an abort condition to the far-end receiver.

Pre-loading the Transmit FIFO

Copy the first two full words of data into the transmit FIFO by writing them into the IrData register.The Ir encode block can hold up to 11 bytes of data (two words in the FIFO plus up to three bytes in the IrDataTail register). If this is sufficient to hold the complete data packet to be transmitted, there will be no DMA activity and the IrCon TUS bit should be cleared. This will cause the Ir encoder to send the CRC and end frame flag correctly.

Setting the DMA Buffer Address, Transfer Length and Mode

DMA channel 1 is used to transfer data to and from the Ir block. The register details can be found in ***12.2.5 DMAC1 Registers*** on page 12-8.

1. Store the word-aligned address of the source data (not including the data already pre-loaded into the transmit FIFO) in DMAADR1.
2. Store the Transfer Length (number of words to be transferred by DMA to the Ir FIFO) in DMATNR1.
3. Clear DMACCR1.MODSEL (selects transfer from memory to IO).
4. Set DMACCR1.mask. This enables end of transfer interrupt.
5. Set DMACCR1.DMEN1. This enables DMA transfer.
6. Ensure that the master enable bit in the DMAC Operation Register is set.

Sending packets which are not a multiple of 4 bytes in length

The FIFO is 32 bits wide. Loading the FIFO with less than 32 bits would cause extraneous zero-bits to be transmitted. The IrDataTail register is a mechanism used to pre-load the last 1, 2 or 3 bytes of a frame. When the DMA transfer is complete and the transmit FIFO is empty, any bytes

stored in the IrDataTail register are transmitted before the Ir encoder sends the CRC and end of frame flags.There are three distinct addresses to write the end of frame data to. The addresses are given in ***Table 12-29: Addresses for end of frame data*** below. This allows a single word write to specify the data to be transmitted and the number of trailing bytes to send.

Bytes to transmit	Address to write to
1	0x80011014
2	0x80011018
3	0x8001101C

Table 12-29: Addresses for end of frame data

If there is a single byte to transmit, write to address 0x80011014, for two bytes write to 0x80011018 and if there are three trailing bytes write to 0x8001101C.

Starting Transmission

Set IrCon.TXE Transmit enable bit.

12.5.3 End of Frame

End of Buffer Interrupt from DMA

DMAFLAGR.FLAG1 will be set when the number of words specified when initializing the DMA transfer length. This can be checked in an Interrupt Service Routine and cleared by writing a 1 to this bit.When this interrupt has occurred, all remaining data to transmit is either in the FIFO or in the IrDataTail register. At this point, clearing the TUS bit in the IrCon register will ensure that when transmission is complete the CRC and end of frame flags are transmitted.

Disable Transmit Circuitry

To save power, after completion of the frame transmission, clear the Transmit Enable (TXE) bit in the IrEnable register.

12.5.4 Error conditions

Transmit buffer Underrun

This is only signalled if IrCon register bit TUS is set to 1 as described in ***Select Transmit Underrun Action*** on page 12-26.

12.6 Receiving Data

Any data error or reception of the end of frame will cause an interrupt to be generated which may be masked with the IrCon.RIM bit (Receive Interrupt Mask).

12.6.1 Initialization

Address Matching

To use Address Match filtering, set the local 8-bit address in the Address Match Value Register, and set the Address Match Enable bit in the IrCon register.

Set up DMA for Receive

1. Store the destination address (word aligned) for incoming data in DMAADDR1.
2. Store the maximum buffer size in words in DMATNR1.
3. Set DMACCR1.MODSEL (selects transfer from IO to memory)
4. Set DMACCR1.MASK (enables end of transfer interrupt)
5. Set DMACCR1.DMEN1 (enables DMA transfer)
6. Ensure that the master enable bit in the DMAC Operation Register is set.

Enable Ir Receive

Set the Receive Enable bit (RXE) in IrEnable

12.6.2 End of Frame

End of Frame Interrupt

An interrupt is generated when a complete frame has been received or when an error in the received data has been detected. To detect if the interrupt is due to reception of a complete frame or due to an error condition, different status registers are used depending on whether MIr or FIr is in operation. In this description, SR0 and SR1 are referred to. In MIr mode, these would be IrMSR0 and IrMSR1. In FIr these would be IrFSR0 and IrFSR1.

Note Bit positions for flags in IrMSR1 and IrFSR1 are *not* identical.

Completing Received Data Transfer

If SR0.RFS (Receive buffer service request flag) is set, then there is valid data in the Ir FIFO which has not been moved by DMA into the external buffer. SR0.WST0 & SR0.WST1 indicate how many valid bytes there are in the top entry of the FIFO. If both bits are zero then there are four valid bytes in the top entry of the FIFO.

To append the data to the data already transferred by DMA, the address of the next word in the DMA buffer must be calculated. The address can be found by subtracting the current value in DMATNR1 (transfer count) from the original value to get the number of words already transferred. This value can then be added to DMAADR1 to give the destination for the word read from IrDATA. In general, there will be 1 to 4 bytes in the receive FIFO, but in exceptional circumstances it is possible that there is another entry in the Ir FIFO, so SR0.RFS should be checked and if necessary the next entry (one to four bytes, indicated by SR0.WST0 & SR0.WST0) appended to the input buffer.

12.6.3 Error conditions

Input Buffer Overrun

A DMA interrupt for the Ir channel during Ir receive indicates that the incoming data has exceeded the buffer length programmed into the DMA.

Receiver Abort Detected

SR0.RAB indicates, when set, that the far-end transmitter sent an abort signal during frame transmission.

Receiver Overrun

SR1.ROR indicates that new data was received while the Ir receiver FIFO was full. This could occur when DMA has been stopped or in the unlikely event of the interrupt service routine not transferring the last two words from the FIFO at the end of a frame.

CRC Error

If the CRC for the received data does not match the CRC value contained in the incoming data stream, SR1.CRC will be set.

Frame Error (FIr only)

IrFSR0.FRE indicates that a framing error has been detected.

These error flags are cleared by reading IrData, at which point the error flags associated with the next word in the FIFO (if present) are transferred into the status registers.

12.7 Special Conditions

This section deals with actions associated with changing Ir operating modes.

12.7.1 Early Termination of Transmission

Clearing IrCon.TXE (transmit enable bit) stops any transmission in progress and clears all data within the FIFO, transmit buffer and serial output shifter.

12.7.2 Early Termination of Reception

Clearing IrCon.RXE receive enable bit stops reception immediately. All data within the receive buffer, serial input shifter and FIFO is cleared.

12.7.3 Changing IrDA Mode

Poll FD or MD bit in IrEnable register until end of transmission is indicated.The new mode can then be set as described in ***12.4 General Configuration***.

12.7.4 Loopback Mode

For test purposes, data will be looped back from the output of the transmit serial shifter into the input of the receive serial shifter when IrEnable.LBM is set.

12.8 Medium Speed Infra-red Port (MIr)

The MIr comprises a dedicated serial port and RTZ modulator/demodulator supporting the Infrared Data Association (IrDA) standard for transmission/reception at 0.576 and 1.152Mb/s.

Frames contain an 8-bit address, an optional control field, a data field of any size that is a multiple of 8-bits and a 16-bit CRC-CCITT. The start/stop flag and CRC generation/checking is handled automatically. Data can be selectively saved in the receive buffer by programming an address with which to compare against all incoming frames. Interrupts are signalled when CRC checks performed on received data indicate an error, when a receiver abort occurs, when the transmit buffer underruns during an active frame and is aborted, when the receive buffer overruns and data is lost.

12.8.1 MIr (Medium-speed Infrared) Operation

Following reset, the MIr is disabled. Reset also causes the transmit and receive buffers and tail register to be flushed (buffers marked as empty). Before enabling the MIr, the user must first clear any writable or “sticky” status bits that are set by writing a one to each bit. (A sticky bit is a readable status bit that may be cleared by writing a one to its location.) Next, the desired mode of operation is programmed in the control register. At this point the user may “prime” the transmit buffer by writing the first data word for transmission and any tail bytes, or the buffer can remain empty and either programmed I/O or the DMA may be used to service it after the MIr is enabled. Once the MIr is enabled, transmission/reception of data can begin on the transmit and receive pins.

Bit Encoding

The MIr bit encoding uses an RZI modulation scheme where a ‘0’ is represented by a light pulse. For both 0.576 and 1.152Mb/s data rates, the optical pulse duration is normally $^1/_4$ of a bit duration. For example, if the data frame (in the order of transmission) is 11010010, then ***Figure 12-7: RZ1/NRZ bit encoding example*** represents the signal that is actually transmitted.

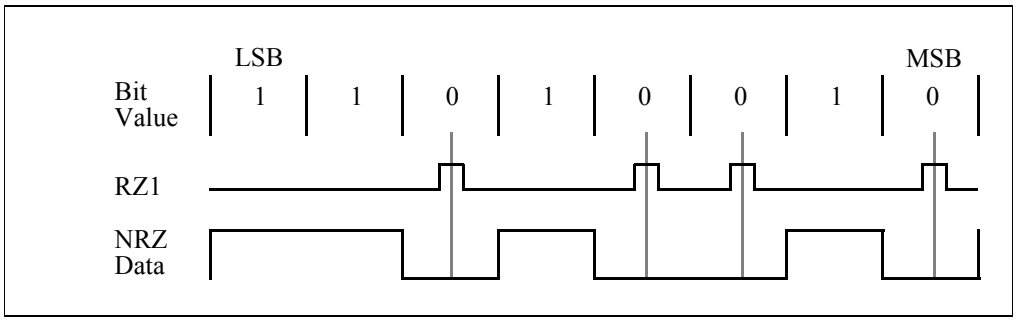


Figure 12-7: RZ1/NRZ bit encoding example

Frame Format

MIr uses a flag (reserved bit pattern) to denote the beginning and end of a frame of information and to synchronize frame transmission. A double flag is used to indicate the start of a frame, and a single flag the end. The flag contains eight bits, which start and end with a zero and contain six sequential ones in the middle (01111110). This sequence of six ones is unique because all data between the start and stop flag is prohibited from having more than five consecutive ones. Data that violates this rule is altered before transmission by automatically inserting a zero after five consecutive ones are detected in the transmitted bit stream. This technique is commonly referred to as “bit stuffing” and is transparent to the user. The information field within a MIr frame is placed between the start and stop flags, consisting of an 8-bit address, an optional 8-bit control field, a data field containing any multiple of 8-bits and a 16-bit cyclical redundancy

check (CRC-CCITT). Note that each byte within the address, control and data fields is transmitted and received LSB first, ending with the byte's MSB. However, the CRC is transmitted and received MSB first.The MIr frame format is outlined below in ***Figure 12-8: MIr frame format***.

8 bits	8 bits	8 bits	8 bits (optional)	Any Multiple of 8 bits	16 bits	8 bits
Start Flag 0111 1110	Start Flag 0111 1110	Address	Control	Data	CRC-CCITT	Stop Flag 0111 1110

Figure 12-8: MIr frame format

Address Field

The 8-bit address field is used by a transmitter to target a select group of receivers when multiple stations are connected using the infrared link. The address allows up to 255 stations to be uniquely addressed (00000000 to 11111110). The global address (11111111) is used to broadcast messages to all stations. The serial port contains an 8-bit register that is used to program a unique address for broadcast recognition as well as a control bit to enable/disable the address match function. Note that the address of received frames is stored in the receive buffer along with normal data, and that it is transmitted and received starting with its LSB and ending with its MSB.

Control Field

The MIr control field is typically 8-bits, but can be any length. The serial port does not provide any hardware decode support for the control byte, but instead treats all bytes between the address and the CRC as data. Thus any control bits appear as data to the programmer. Note that the control field is transmitted and received starting with its LSB and ending with its MSB.

Data Field

The data field can be any length that is a multiple of 8 bits, including zero. The user determines the data field length according to the application requirements and transmission characteristics of the target system. Usually a length is selected which maximizes the amount of data that can be transmitted per frame, while allowing the CRC checker to be able to consistently detect all errors during transmission. All data fields must be a multiple of 8 bits. If a data field that is not a multiple of 8 bits is received, an abort is signalled and the end of frame tag is set within the receive buffer. Also note that each byte within the data field is transmitted and received starting with its LSB and ending with its MSB.

CRC Field

MIr uses the established CCITT cyclical redundancy check (CRC) to detect bit errors that occur during transmission. A 16-bit CRC-CCITT is computed using the address, control and data fields, and is included in each frame. A separate CRC generator is implemented in both the transmit and receive logic. The transmitter calculates a CRC while data is actively transmitted, and places the 16-bit value at the end of each frame before the stop flag is transmitted. The receiver calculates a CRC for each received data frame, and compares the calculated CRC to the expected CRC value contained within the end of each received frame. If the calculated value does not match the expected value, an interrupt is signalled. The CRC computation logic is

preset to all ones before reception/transmission of each frame. Note that the CRC is transmitted and received starting with its MSB and ending with its LSB. The CRC uses the four-term polynomial:

$$CRC(x) = (X^{16} + X^{12} + X^5 + 1)$$

Baud Rate Generation

The baud or bit rate is derived by dividing down the 48MHz clock generated by the on-chip PLL. The clock is first divided down by 10 and five-twelfths, then either 1 (BRD=1) or 2 (BRD=0), and then by a fixed value of four, generating the transmit clock for 1.152Mb/s and 0.576Mb/s data rates, respectively. The receive clock is generated by the receiver Digital Phase Locked Loop (DPLL). The DPLL uses a sample clock that is undivided. A sample rate counter (incremented at the sample clock rate) is used to generate a receive clock at the nominal data rate (sample clock divided by 41 and two-thirds). The sample rate counter is reset on the detection of each positive-going data transition (indicating the RZI encoding of a '0') to ensure that synchronization with the incoming data stream is maintained.

Receive Operation

Once the MIr receiver is enabled it enters hunt mode, searching the incoming data stream for the flag (01111110). The flag serves to achieve bit synchronization, denotes the beginning of a frame, and delineates the boundaries of individual bytes of data. The end of the flag denotes the beginning of the address byte. Once the flag is found, the receiver is synchronized to incoming data and hunt mode is exited.

After each bit is decoded, a serial shifter is used to receive the incoming data a byte at a time. Once the flag is recognized, each subsequent byte of data is decoded and placed within a two-byte temporary buffer. A temporary buffer is used to prevent the CRC from being placed within the receive buffer. When the temporary buffer is filled, data values are pushed out one by one to the receive buffer. The first byte of a frame is the address. If receiver address matching is enabled, the received address is compared to the address programmed in the address match value field in a control register. If the two values are equal or if the incoming address contains all ones, all subsequent data bytes including the address byte are stored in the receive buffer. If the values do not match, the receive logic does not store any data in the receive buffer, ignores the remainder of the frame and begins to search for the stop flag. The second byte of the frame can contain an optional control field that must be decoded in software (There is no hardware support within the MIr). Use of a control byte is determined by the user.

When the receive buffer is filled, a DMA request is signalled. If the data is not removed soon enough and the buffer is completely filled, an overrun error is generated when the receive logic attempts to place additional data into the full buffer. Once the buffer is full, all subsequent received data are lost while the buffer contents remain intact.

Frames can contain any amount of data in multiples of 8-bits. Although the MIr protocol does not limit frame size, in practice they tend to be implemented in numbers ranging from hundreds to a couple of thousand bytes.

The receive logic continuously searches for the stop flag at the end of the frame. Once it is recognized, the last byte that was placed within the receive buffer is flagged as the last byte of the frame, and the two bytes remaining within the temporary buffer are removed and used as the 16-bit CRC value for the frame. Instead of placing this in the receive buffer, the receive logic compares it to the CRC-CCITT value which is continuously calculated using the incoming data stream. If they do not match, the last byte that was placed within the receive buffer is also flagged with a CRC error. The CRC value is not placed in the receive buffer.

The MIr protocol permits back to back frames to be received. When this occurs, three flags separate back to back frames.

Most commercial IrDA transceivers can generate an abort (7 to 13 ones) when their transmit buffer underruns. The receive logic contains a counter that increments each time a one is decoded before entering the serial shifter, and is reset any time a zero is decoded. When seven

or more ones are detected, a receiver abort occurs. Note that data is moved from the serial shifter to the temporary buffer a byte at a time, and seven consecutive ones may bridge two bytes. For this reason, after an abort is detected, the remaining data in the serial shifter is discarded along with the most recent byte of data placed in the temporary buffer. After this data is discarded, the oldest byte of data in the temporary buffer is placed in the receive buffer, the EOF tag is set within the top entry of the buffer (next to the byte transferred from the temporary buffer), the receiver abort interrupt is signalled, and the receiver logic enters hunt mode until it recognizes the next flag.

If the user disables the receiver during operation, reception of the current data byte is stopped immediately, the serial shifter and receive buffer are cleared, and all clocks used by the receive logic are automatically shut off to conserve power.

Transmit Operation

The user may either “prime” the transmit buffer by filling it with data or allow service requests to cause the DMA to fill the buffer once the MIr transmitter is enabled. Once enabled, the transmit logic issues a service request if its buffer is empty. A Serial Infrared Interaction Pulse (SIP) is transmitted in order to guarantee non-disruptive co-existence with slower (up to 115.2Kb/s) systems, for example another GMS30C7201 device attempting to use its SIr. This is followed by continuous transmission of flags until valid data resides within the buffer. Once a byte of data resides at the bottom of the transmit buffer, it is transferred to the serial shifter, is encoded and shifted out onto the transmit pin clocked by the programmed baud rate clock. Note that the flag and CRC value are automatically transmitted and need not be placed in the transmit buffer.

When the transmit buffer is emptied, an interrupt and/or DMA service request is signalled. If new data is not supplied soon enough, the buffer is completely emptied and the transmit logic attempts to take additional data from the empty buffer, one of two actions can be taken as programmed by the user. An underrun can either signal the normal completion of a frame or an unexpected termination of a frame in progress.

When normal frame completion is selected and an underrun occurs, the transmit logic transmits the 16-bit CRC value calculated during the transmission of all data within the frame (including the address and control bytes), followed by a flag to denote the end of the frame. The transmitter then transmits an SIP, followed by a continuous transmission of flags until data is once again available within the buffer. Once data is available, the transmitter begins transmission of the next frame.

When unexpected frame termination is selected and an underrun occurs, the transmit logic outputs an abort and interrupts the CPU. An abort continues to be transmitted until data is once again available in the transmit buffer. The MIr then transmits an SIP, followed by a double flag and starts the new frame. The off-chip receiver may choose to ignore the abort and continue to receive data, or to signal the serial port to retry transmission of the aborted frame.

If the user disables the transmitter during operation, transmission of the current data byte is stopped immediately, the serial shifter and transmit buffer are cleared, and all clocks used by the transmit logic are automatically shut off to conserve power.

CPU and DMA Register Access Sizes

Bit positioning, byte ordering and addressing of the MIr are described in terms of little endian ordering. All MIr control and status registers are 8-bits wide and are located in the least significant byte of individual words. Transmit and receive data buffers are 32 bits wide, with the first byte to be transmitted/ or received located in the least significant byte position. The ARM peripheral bus does not support byte or half-word operations. All reads and writes of the MIr by the CPU should be word wide. Separate DMA requests exist for the transmit and the receive buffer. If the DMA controller is used to service the transmit and/or receive buffers, the user must ensure the DMA is properly configured to perform single word-wide accesses. Burst mode DMA is not supported by the peripheral. Refer to ***Table 12-34: Ir Interface Block Registers and their Physical Addresses*** on page 12-51 for a summary of the MIr serial port’s registers.

12.8.2 MIr Register Definitions

The MIr uses the control and data registers described in the previous section. These are shared with the FIr interface and can only be used with the MIr when the MIr is selected using the IrEnable register. In addition to the shared registers there are two status registers specific to the MIr.

The status registers contain bits that signal CRC, overrun, underrun and receiver abort errors as well as the transmit buffer service request, receive buffer service request and end of frame conditions. Detection of end of frame, underrun and receiver abort errors signal interrupt requests to the interrupt controller. The status registers also contains flags for transmitter busy, receiver synchronized, receive buffer not empty, transmit buffer not full and receive transition detect (No interrupt is generated).

12.8.3 MIr Status Register 0

MIr status register 0 (MISR0) contains bits that signal the transmit buffer service request, receive buffer service request, receiver abort, transmit buffer underrun and the end/error in receive buffer condition. Detection of receiver abort, transmit buffer underrun and the end/error in receive buffer condition signal an interrupt request to the interrupt controller.

Bits that cause an interrupt signal the interrupt request as long as the bit is set. Once the bit is cleared, the interrupt is cleared. Read/write bits are called status bits, read-only bits are called flags. Status bits are referred to as “sticky” (once set by hardware, they must be cleared by software). Writing a one to a sticky status bit clears it, writing a zero has no effect. Read-only flags are set and cleared by hardware, writes have no effect.

End/Error in buffer Status (EIF)(read-only)

The end/error in buffer flag (EIF) is a read-only bit that is set when any tag bits (32-36) are set in either entry of the receive buffer, and is cleared when no error bits are set within the buffer. When EIF is set an interrupt is signalled and DMA requests to empty the receive buffer are disabled until EIF is cleared. Once all set tag bits are cleared from the receive buffer, EIF is automatically cleared, which in turn clears the interrupt and re-enables the receive buffer DMA request.

Transmit Underrun Status (TUR) (read/write)

The transmit underrun status bit (TUR) is set when the transmit logic attempts to fetch data from the transmit buffer while it and the tail register are empty. When an underrun occurs, the transmitter takes one of the following two actions. When the transmit underrun select bit is clear (TUS=0) the transmitter ends the frame by shifting out the CRC which is calculated continuously on outgoing data, followed by a flag. When TUS=1, the transmitter is forced to transmit an abort and continues to transmit ones until valid data is again available within the buffer. Once data resides in the transmit buffer, a new data frame is initiated by transmitting an SIP and a start flag followed by the transmission of data from the buffer. When the TUR bit is set, an interrupt request is made. Note that underruns are not generated when the MIr transmitter is first enabled and is in the idle state (continuously transmits flags).

Receiver Abort Status (RAB)(read/write)

The receiver abort status bit (RAB) is set for two different cases:

- when an abort is detected during receipt of an incoming frame
- if the stop flag is not received on a byte boundary.

An abort is signalled when seven or more consecutive ones are detected in the incoming data stream. It is also generated when the end flag is received and it is not on a byte boundary, which indicates that the address, control and data fields did not add up to an even multiple of 8-bits. When an abort is detected, the current data byte within the serial shifter is discarded, the least recent byte (the oldest of the two bytes) of data in the temporary buffer is moved to the receive

buffer (the other byte is discarded), and the EOF tag is set in the buffer entry that corresponds to the last piece of data that was received before the frame was aborted. The receiver then enters hunt mode, searching for a flag. When the RAB bit is set, an interrupt request is made.

Transmit buffer Service Request Flag (TFS)(read-only)

The transmit buffer service request flag (TFS) is a read-only bit that is set when the transmit buffer is not full and requires service to prevent an underrun. The state of TFS is also sent to the DMA controller, and may be used to signal a DMA service request. After the DMA or CPU fills the buffer, the TFS flag (and the service request) is automatically cleared.

Receive buffer Service Request Flag (RFS) (read-only)

The receive buffer service request flag (RFS) is a read-only bit that is set when the receive buffer contains valid data. The state of RFS is also sent to the DMA controller, and may be used to signal a DMA service request. After the DMA or CPU empties the buffer, the RFS flag (and the service request) is automatically cleared.

Figure 12-9: Bit locations in MIr status register 0 shows the bit locations corresponding to the status and flag bits within MIr status register 0. Note that the reset state of all writable status bits is unknown and must be cleared (by writing a one to them) before enabling the MIr. Also note that writes to reserved bits are ignored and reads return zeros.

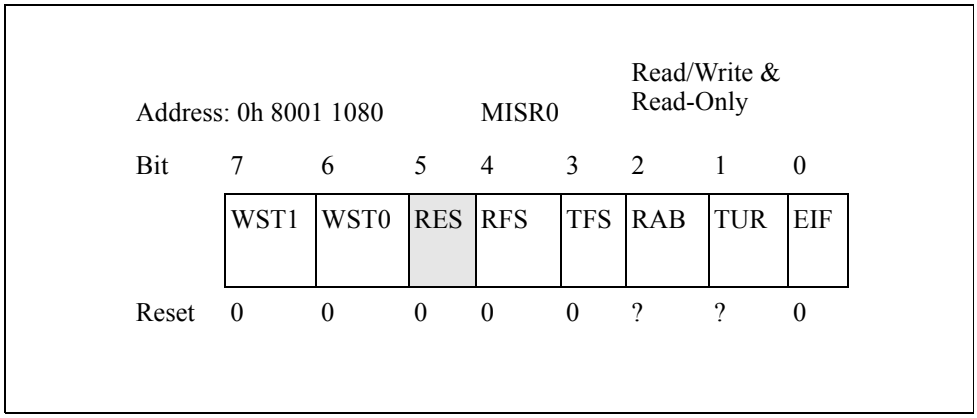


Figure 12-9: Bit locations in MIr status register 0

Bit	Name	Description
0	EIF	**Error in buffer (read-only)** 0 - Bits 32-36 are clear within each valid entry of the receive buffer, receive buffer DMA service requests are enabled 1 - One or more tag bits (32-36) are set within one or more entries in the receive buffer, request interrupt, disable receive buffer DMA service requests
1	TUR	**Transmit buffer Underrun** 0 - Transmit buffer has not experienced an underrun 1 - Transmit logic attempted to fetch data from transmit buffer while it and the tail register were empty, interrupt request signalled

Table 12-30: MIr status register 0

Bit	Name	Description
2	RAB	**Receiver Abort** 0 - No abort has been detected for the incoming frame 1 - Abort detected during receipt of incoming frame, seven or more ones detected on receive pin, EOF bit set in receive buffer next to last piece of "good" data received before the abort, interrupt requested
3	TFS	**Transmit buffer Service Request (read-only)** 0 - Transmit buffer is full or the transmitter disabled 1 - Transmit buffer is not full and the transmitter is enabled, DMA service request signalled
4	RFS	**Receive buffer Service Request (read-only)** 0 - Receive buffer is empty or the receiver disabled 1 - Receive buffer is not empty and the receiver is enabled. DMA service request is signalled unless the receive buffer contains either an error or the final byte in a frame (Indicated by EIF set).
5	-	Reserved
7-6	WST	**Receive word width status** 00 - All four bytes in receive buffer are valid 01 - Least significant byte valid only 10 - Least significant two bytes valid only 11 - Least significant three bytes valid only

Table 12-30: MIr status register 0 (Continued)

12.8.4 MIr status register 1 (MISR1)

MIr status register 1 (MISR1) contains flags and status bits that indicate when the receiver is synchronized, the transmitter is active, that the transmit buffer is not full, that the receive buffer is not empty, a transition has been detected on the receive line, and when an end of frame, CRC error, or underrun error has occurred. All bits within MISR1 are non-interrupting.

Receiver Synchronized Flag (RSY) (read-only)

The receiver synchronized (RSY) flag is a read-only bit that is set when the receiver is synchronized with the incoming data stream, and is cleared when the receiver logic is in hunt mode (looking for a flag to achieve bit and frame synchronization), or the receiver is disabled (RXE=0).

Transmitter Busy Flag (TBY) (read-only)

The transmitter busy (TBY) flag is a read-only bit that is set when the transmitter is actively transmitting a frame (address, control, data, CRC, start or stop flag), and is cleared when the transmitter is idle (transmitting flags that are not part of a frame), or the transmitter is disabled (TXE=0).

Receive Transition Detect Status (RTD) (read/write)

The receive transition detect (RTD) status bit is set whenever the receiver is enabled (RXE=1), and a positive edge transition is detected on the **RXD1** pin.

End of Frame Flag (EOF) (read-only)

The end of frame flag (EOF) is set when the last byte of data within a frame (including aborted frames) resides within the bottom entry of the receive buffer.

The receive buffer contains three tag bits (32, 33 and 34) that are not directly readable. The 32nd bit is set at the top of the buffer whenever the last byte within a frame is moved from the receive serial shifter to the receive buffer. This tag travels along with the last data value to the buffer.

CRC Error Status (CRE) (read-only)

The CRC error flag (CRE) is set when the CRC value calculated by the receive logic does not match the CRC value contained within the incoming serial data stream.

Whenever a CRC error is detected, the 33rd bit is set within the top entry of the receive buffer corresponding to the last byte of data within the frame

Receiver Overrun Status (ROR) (read-only)

The receiver overrun flag (ROR) is set when the receive logic attempts to place data into the receive buffer after it has been filled.

The 34th bit is set within the top entry of the receive buffer whenever an overrun occurs. This tag travels along with the last “good” data value before the overflow occurred.

Figure 12-10: bits within MIr status register 1 shows the location of the flag and status bits within MIr status register 1. The bits within this register do not produce interrupt requests. Note that the reset value of RTD is unknown and must be cleared if set following a reset of the ARM 7201. The remainder of FIDR is read-only (writes are ignored).

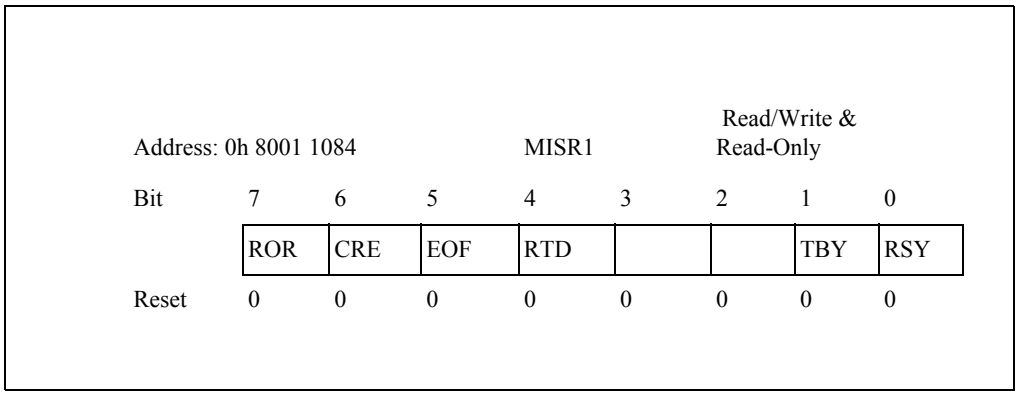


Figure 12-10: bits within MIr status register 1

Bit	Name	Description
0	RSY	**Receiver Synchronized Flag (read-only)** 0 - Receiver is in hunt mode or is disabled 1 - Receiver logic is synchronized with the incoming data
1	TBY	**Transmitter Busy Flag (read-only)** 0 - Transmitter is idle (continuous flags), or disabled, or an abort is being transmitted 1 - Transmit logic is currently transmitting a frame (address, control, data, CRC, or start/stop flag)
2	-	unused
3	-	unused

Table 12-31: MIr status register 1

Bit	Name	Description
4	RTD	**Receive Transition Detect** 0 - No transition detected on receiver pin since the last time s/w cleared this bit 1 - Rising edge detected on receiver pin
5	EOF	**End of Frame (read-only)** 0 - Current frame has not completed 1 - The value in the receive buffer contains the last byte of data within the frame
6	CRE	**CRC Error (read-only)** 0 - No CRC check errors encountered in the receipt of data 1 - CRC calculated on the incoming data does not match CRC value contained within the received frame
7	ROR	**Receive buffer Overrun (read-only)** 0 - Receive buffer has not experienced an overrun 1 - Receive logic attempted to place data into receive buffer while it was full, the next data value in the buffer is the last piece of “good” data before the buffer was overrun

Table 12-31: MIr status register 1 (Continued)

12.9 Fast Infrared Port (FIr)

The Fast Infrared port (FIr) operates at half-duplex and provides direct connection to commercially available Infrared Data Association (IrDA) compliant LED transceivers. The FIr supports the 4.0Mbps IrDA standard, using four pulse position modulation (4PPM) and a specialized serial packet protocol developed expressly for IrDA transmission.

High Speed FIr Operation

Before enabling the FIr for high-speed operation, the user must first clear any writable or "sticky" status bits which are set by writing a one to each bit. Next, the desired mode of operation is programmed in the control registers. At this point the user may "prime" the FIr's transmit buffer by writing the first data word, or the buffer can remain empty and either programmed I/O or the DMA may be used to service it after the FIr is enabled. Once the FIr is enabled, transmission/reception of data can begin on the transmit and receive pins. The FIr's frame formats are outlined below.

4PPM Modulation

Four position pulse modulation (4PPM) is used for the high-speed transmission rate of 4.0Mbps. Payload data is divided into data bit pairs (DBPs) for encoding with LSBs transmitted first. Each DBP is represented by one of four symbols (DDs) comprising a single 125ms pulse within a 500ms symbol period. The 125ms quarters of a symbol are known as "chips". The resulting signal waveform for the four data DDs is shown in ***Figure 12-11: 4PPM Modulation Encoding***. ***Figure 12-12: 4PPM Modulation Example*** on page 12-41 shows an example of 4PPM modulation of the byte, 0b10110001 which is constructed using four DBPs. Note that bits within each DBPare not reordered, but the least significant DBP is transmitted first.

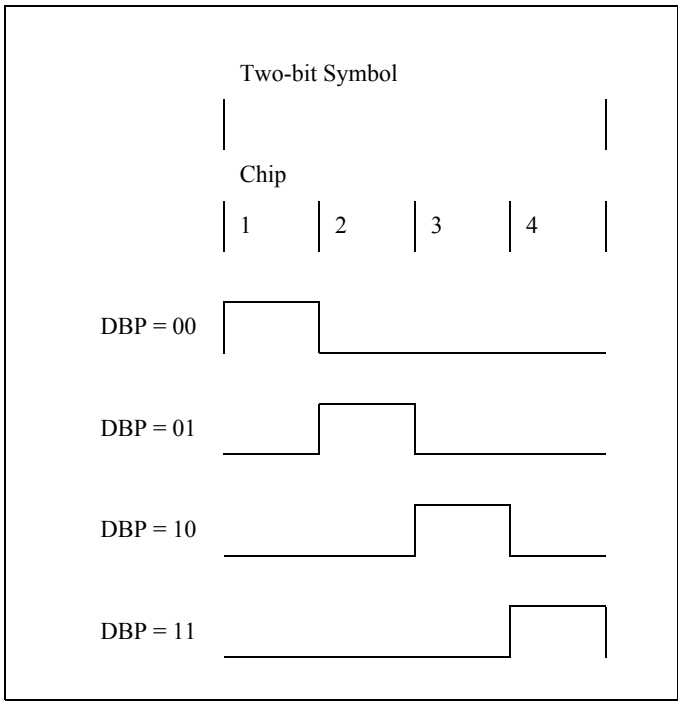


Figure 12-11: 4PPM Modulation Encoding

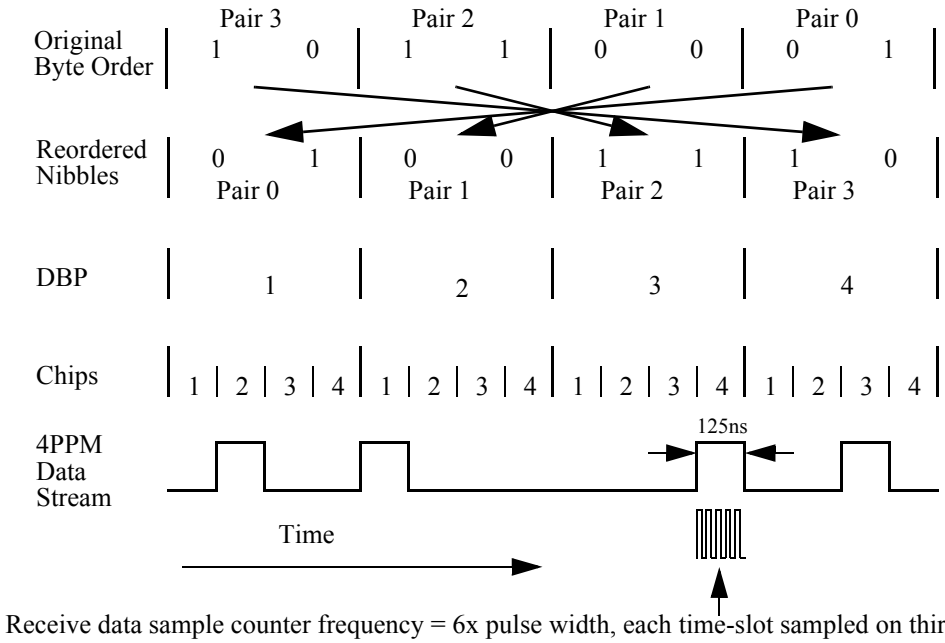


Figure 12-12: 4PPM Modulation Example

4.0Mbps FIr Frame Format

When the 4.0Mbps transmission rate is used, the high-speed serial/parallel (FIr) interface within the FIr is used along with the 4PPM bit encoding. The high-speed frame format shown in ***Figure 12-13: High Speed Serial Frame Format for IrDA Transmission (4.0Mbps)*** on page 12-42 is similar to the SDLC format with several minor modifications: the start/stop flags and CRC are twice as long, and instead of one start flag, a preamble and start flag of differing length are used.

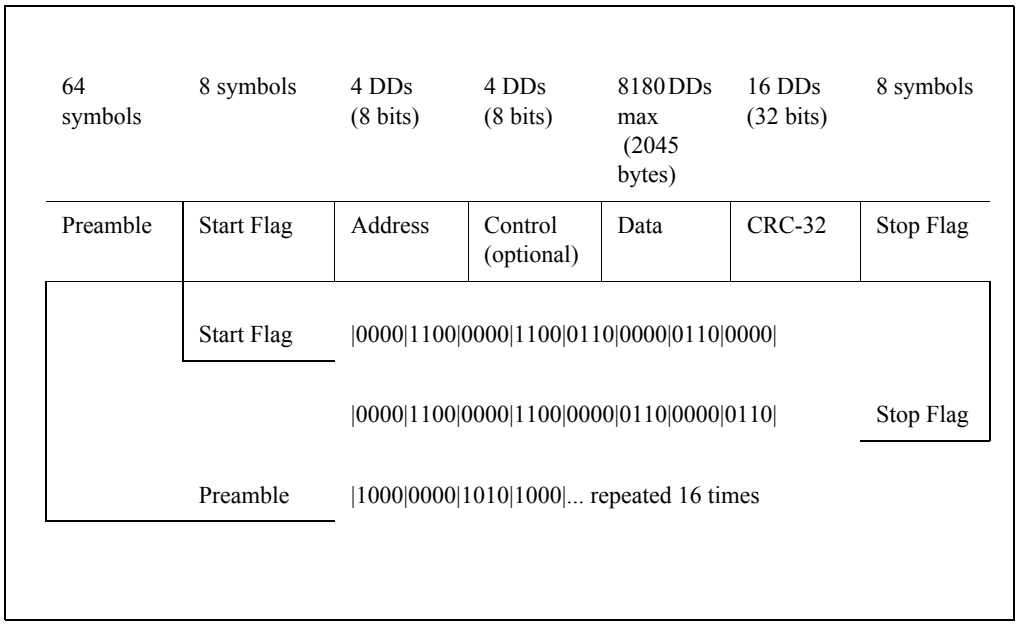


Figure 12-13: High Speed Serial Frame Format for IrDA Transmission (4.0Mbps)

The preamble, start, and stop flags are a mixture of symbols which contain either 0, 1, or 2 pulses within the four time-slots. Symbols with 0 and 2 pulses are used to construct flags since they represent invalid data bit pairings (one pulse required per symbol to represent one of four bit pairs). The preamble contains sixteen repeated transmissions of the four symbols: 1000 0000 1010 1000, the start flag contains one transmission of eight symbols: 0000 1100 0000 1100 0110 0000 0110 0000, and the stop flag contains one transmission of eight symbols: 0000 1100 0000 1100 0000 0110 0000 0110. The address, control, data, and CRC-32 all use the standard 4PPM DDs described above.

Address Field

The 8-bit address field is used by a transmitter to target a select group of receivers when multiple stations are connected to the same set of serial lines. The address allows up to 255 stations to be uniquely addressed (00000000 to 11111110). The global address (11111111) is use to broadcast messages to all stations. Serial port 1 contains an 8-bit register which is used to program a unique address for broadcast recognition as well as a control bit to enable/disable the address match function. Note that the address of received frames is stored in the receive buffer along with normal data, and that it is transmitted and received starting with its LSB and ending with its MSB.

Control Field

The IPC control field is 8-bits and is optional (as defined by the user). The FIr does not provide any hardware decode support for the control byte, but instead treats all bytes between the address and the CRC as data. Note that the control field is transmitted and received starting with its LSB and ending with its MSB.

Data Field

The data field can be any length which is a multiple of 8-bits, from 0 to 2045 bytes. The user determines the data field length according to the application requirements and transmission characteristics of the target system. Usually a length is selected which maximizes the amount of data which can be transmitted per frame, while allowing the CRC checker to be able to consistently detect all errors during transmission. Note that the serial port does not contain any

hardware which restricts the maximum amount of data transmitted or received. It is up to the user to maintain these limits. If a data field which is not a multiple of 8-bits is received an abort is signalled. Also note that each byte within the data field is transmitted and received starting with its LSB and ending with its MSB.

CRC Field

The FIr uses the established 32-bit cyclical redundancy check (CRC-32) to detect bit errors which occur during transmission. A 32-bit CRC is computed using the address, control, and data fields, and is included in each frame. A separate CRC generator is implemented in both the transmit and receive logic. The transmitter calculates a CRC while data is actively transmitted byte shifting each byte transmitted through its serial shifter LSB first, then places the inverse of the resultant 32-bit value at the end of each frame before the flag is transmitted. In a similar manner, the receiver also calculates a CRC for each received data frame, and compares the calculated CRC to the expected CRC value contained within the end of each received frame. If the calculated value does not match the expected value, an interrupt is signalled. The CRC computation logic is preset to all ones before reception/transmission of each frame and the result is inverted before it used for comparison or transmission. Note that unlike the address, control, and data fields, the 32-bit inverted CRC value is transmitted and received from least significant byte to most significant, and within each byte the least significant nibble is encoded/decoded first. The cyclical redundancy checker uses the 32 term polynomial:

$$:) = (x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} +$$

Baud Rate Generation

The baud rate is derived by dividing down a fixed 48MHz clock generated by one of the two on-chip PLLs by six. The 8MHz baud (time-slot) clock for the receive logic is synchronized with the 4PPM data stream each time a transition is detected on the receive data line using a digital PLL. To encode a 4.0Mbps data stream, the required “symbol” frequency is 2.0MHz, with four chips per symbol at a frequency of 8.0MHz. Receive data is sampled half way through each time-slot period by counting three out of the six 48MHz clock periods which make up each chip. Refer to ***Figure 12-12: 4PPM Modulation Example*** on page 12-41. The symbols are synchronized during preamble reception. Recall that the preamble consists of four symbols repeated sixteen times. This repeating pattern is used to identify the first time-slot or beginning of a symbol, and resets the two-bit chip counter logic, such that the 4PPM data is properly decoded.

Receive Operation

The IrDA standard specifies that all transmission occurs at half-duplex. This restriction forces the user to enable one direction at a given time; either the transmit or receive logic, but not both. However, the FIr’s hardware does not impose such a restriction.The user may enable both the transmitter and receiver at the same time. Although forbidden by the IrDA standard, this feature is particularly useful when using the FIr’s loop back mode, which internally connects the output of the transmit serial shifter to the input of the receive serial shifter.

After the FIr is enabled for 4.0Mbps transmission, the receiver logic begins by selecting an arbitrary symbol boundary, receives four incoming 4PPM symbols from the input pin using a serial shifter, and latches and decodes the symbols one at a time. If the symbols do not decode to the correct preamble, the chip counter’s clock is forced to skip one 8MHz period, effectively delaying the chip count by one. This process is repeated until the preamble is recognized, signifying that the chip counter is synchronized. The preamble may be repeated as few as sixteen times, or may be continuously repeated to indicate an idle receive line.

At any time after the transmission of sixteen preambles, the start flag may be received. The start flag is eight symbols long. If any portion of the start flag does not match the standard encoding, the receive logic signals a framing error and the receive logic once again begins to look for the frame preamble.

Once the correct start flag is recognized, each subsequent grouping of four DDs is decoded into a data byte, placed within a five byte temporary buffer which is used to prevent the CRC from being placed within the receive buffer. When the temporary buffer is filled, data values are pushed out one by one to the receive buffer. The first data byte of a frame is the address. If receiver address matching is enabled, the received address is compared to the address programmed in the address match value field in one of the control registers. If the two values are equal or if the incoming address contains all ones, all subsequent data bytes including the address byte are stored in the receive buffer. If the values do not match, the receiver logic does not store any data in the receive buffer, ignores the remainder of the frame, and begins to search for the next preamble. The second data byte of the frame can contain an optional control field as defined by the user and must be decoded in software (There is no hardware support within the FIr).

Frames can contain any amount of data in multiples of 8-bits up to a maximum of 2047 bytes (including the address and control byte). The FIr does not limit frame size, thus it is the responsibility of the user to check that the size of each incoming frame does not exceed the IrDA protocol's maximum allowed frame size.

When the receive buffer is filled, an interrupt or DMA transfer is signalled. If the data is not removed quickly enough, an overrun error is signalled when the receive logic attempts to place additional data into the full buffer. Once the buffer is full, all subsequent data bytes received are lost while all buffer contents remain intact.

If any two sequential symbols within the data field do not contain pulses (are 0000), the frame is aborted, the least recent or oldest byte within the temporary buffer is moved to the receive buffer (the remaining four buffer entries are discarded), the end of frame (EOF) tag is set within the same buffer entry where the last "good" byte of data resides, and the receiver logic begins to search for the preamble. An abort also occurs if any data symbol containing 0011, 1010, 0101, or 1001 occurs (invalid symbols which do not occur in the stop flag).

The receive logic continuously searches for the 8-symbol stop flag. Once it is recognized, the last byte which was placed within the receive buffer is flagged as the last byte of the frame and the data in the temporary buffer is removed and used as the 32-bit CRC value for the frame. Instead of placing this in the receive buffer, the receive logic compares it to the CRC-32 value which is continuously calculated using the incoming data stream. If they do not match, the last byte which was placed within the receive buffer is also tagged with a CRC error. The CRC value is not placed in the receive buffer.

If the user disables the FIr's receiver during operation, reception of the current data byte is stopped immediately, the serial shifter and receive buffer are cleared, and all clocks used by the receive logic are automatically shut off to conserve power.

Transmit Operation

Before enabling the FIr for transmission, the user may either "prime" the transmit buffer by filling it with data or allow service requests to cause the CPU or DMA to fill the buffer once the FIr is enabled. Once enabled, the transmit logic issues a service request if its buffer is empty. For each frame output, a minimum of sixteen preambles are transmitted. If data is not available after the sixteenth preamble, additional preambles are output until a byte of valid data resides within the bottom of the transmit buffer. The preambles are then followed by the start flag and then the data from the transmit buffer. Four symbols (8 bits) are encoded at a time and then loaded into a serial shift register. The contents are shifted out onto the transmit pin clocked by the 8MHz baud clock. Note that the preamble, start and stop flags, and CRC value is automatically transmitted, and need not be placed in the transmit buffer.

When the transmit buffer is emptied, an interrupt and/or DMA service request is signalled. If new data is not supplied quickly enough, and the transmit logic attempts to take additional data from the empty buffer, one of two actions can be taken as programmed by the user. An underrun can either signal the normal completion of a frame or an unexpected termination of a frame in progress.

When normal frame completion is selected and an underrun occurs, the transmit logic transmits the 32-bit CRC value calculated during the transmission of all data within the frame (including the address and control bytes), followed by the stop flag to denote the end of the frame. The transmitter then continuously transmits preambles until data is once again available within the buffer. Once data is available, the transmitter begins transmission of the next frame.

When unexpected frame termination is selected and an underrun occurs, the transmit logic outputs an abort and interrupts the CPU. An abort continues to be transmitted until data is once again available in the transmit buffer. The FIr then transmits 16 preambles, a start flag, and starts the new frame. The remote receiver may choose to ignore the abort and continue to receive data, or to signal the FIr to retry transmission of the aborted frame.

At the end of each frame transmitted, the FIr outputs a pulse called the serial infrared interaction pulse (SIP). A SIP is required at least every 500ms to keep slower speed devices (115.2Kbps and slower) from colliding with the higher speed transmission. The SIP simulates a start bit which causes all low speed devices to stay off the bus for at least another 500ms. Transmission of the SIP pulse causes the transmit pin to be forced high for a duration of 1.625us and low for 7.375us (total SIP period = 9.0us). After the 9.0us elapses, the preamble is then transmitted continuously to indicate to the remote receiver that the FIr's transmitter is in the idle state. The preamble continues to be transmitted until new data is available within the transmit buffer, or the FIr's transmitter is disabled. Note that it is the responsibility of the user to ensure that a frame completes once every 500ms such that a SIP pulse is produced keeping all low speed devices from interrupting transmission. Because most IrDA compatible devices produce a SIP after each frame transmitted, the user may only need to ensure that a frame is either transmitted or received by the FIr every 500ms. Note that frame length does not represent a significant portion of the 500ms time frame in which a SIP must be produced. At 4.0Mbps, the longest frame allowed is 16,568 bits, which takes just over 4ms to transmit. Also note that the FIr issues a SIP when the transmitter is first enabled, to ensure all low speed devices are silenced before transmitting it's first frame.

If the user disables the FIr's transmitter during operation, transmission of the current data byte is stopped immediately, the serial shifter and transmit buffer are cleared and all clocks used by the transmit logic are automatically shut off to conserve power.

CPU and DMA Register Access Sizes

Bit positioning, byte ordering, and addressing of the FIr is describes in terms of little endian ordering. All FIr control and status registers are 8 bits wide and are located in the least significant byte of individual words. Data transfers are up to 32 bits wide. If the DMA controller is used to service the transmit and/or receive buffers, the user must ensure the DMA is properly configured to perform single word-wide accesses.

DMA burst mode access is not supported.

FIr Register Definitions

The Fir uses the control and data registers described in ***12.3 Medium and Fast Infrared Module*** on page 12-17. In addition there are two status registers specific to the FIr

The status registers contain bits which signal CRC, overrun, underrun, framing, and receiver abort errors as well as the transmit buffer service request, receive buffer service request, and end of frame conditions. Each of these hardware detected events signal an interrupt request to the interrupt controller. The status registers also contain flags for transmitter busy, receiver synchronized, receive buffer not empty, and transmit buffer not full (no interrupt generated).

The status registers contain bits which signal CRC, overrun, underrun, framing, and receiver abort errors as well as the transmit buffer service request, receive buffer service request, and end of frame conditions. Each of these hardware detected events signal an interrupt request to the interrupt controller. The status registers also contains flags for transmitter busy, receiver synchronized, receive buffer not empty, and transmit buffer not full (no interrupt generated).

12.9.1 FIr Status Register 0

FIr status register 0 (FISR0) contains bits which signal the transmit buffer service request, receive buffer service request, receiver abort, transmit buffer underrun, framing error, and the end/error in receive buffer condition. Each of these hardware detected events signal an interrupt request to the interrupt controller.

Bits which cause an interrupt signal the interrupt request as long as the bit is set. Once the bit is cleared, the interrupt is cleared. Read/write bits are called status bits, read-only bits are called flags. Status bits are referred to as “sticky” (once set by hardware, must be cleared by software). Writing a one to a sticky status bit clears it, writing a zero has no effect. Read-only flags are set and cleared by hardware, writes have no effect. Additionally some bits which cause interrupts have corresponding mask bits in the control registers and are indicated in the section headings below.

End/Error in buffer Status (EIF) (read/write, non-maskable interrupt)

The end/error in buffer status bit (EIF) is set when any tag bits (32 through 36) are set in the receive buffer. When EIF is set an interrupt is signalled and DMA requests to empty the receive buffer are disabled until EIF is cleared

Transmit Underrun Status (TUR) (read/write, maskable interrupt)

The transmit underrun status bit (TUR) is set when the transmit logic attempts to fetch data from the transmit buffer after it has been completely emptied. When an underrun occurs, the transmitter takes one of two actions. When the transmit underrun select bit is clear (TUS=0) the transmitter ends the frame by shifting out the CRC which is calculated continuously on outgoing data, followed by a stop flag and SIP pulse. When TUS=1, the transmitter is forced to transmit an abort and continues to transmit symbols containing all zeros (0000) until valid data is again available within the buffer. Once data resides within the bottom entry of the transmit buffer, a new data frame is initiated by transmitting sixteen preambles and a start flag followed by the transmission of data from the buffer. When the TUR bit is set an interrupt request is made unless it is masked. When TUS=0 the interrupt is masked, when TUS=1 it is enabled. Note that underruns are not generated when the FIr transmitter is first enabled and is in the idle state (continuously transmits flags).

Receiver Abort Status (RAB) (read/write, non-maskable interrupt)

The receiver abort status bit (RAB) is set when an abort is detected during receipt of an incoming frame. An abort is signalled when two or more symbols which do not contain any pulses (0000) or symbols containing 0011, 1001, 1010, or 0101(invalid symbols which are not contained within the stop flag) are detected after a valid start flag has been detected but before a complete stop flag has been received (i.e. an incorrect chip in the stop flag generates an abort as well). When an abort is received, the EOF tag is set in the buffer entry which corresponds to the last piece of data which was received before the frame was aborted. The receiver then enters hunt mode, searching for the preamble.

Transmit buffer Service Request Flag (TFS) (read-only, maskable interrupt)

The transmit buffer service request flag (TFS) is a read-only bit which is set when the transmit buffer is not full and requires service. When the TFS bit is set, an interrupt request is made unless the transmit buffer interrupt request mask (TIM) bit is cleared. The state of TFS is also sent to the DMA controller, and may be used to signal a DMA service request. Note that TIM has no effect on the generation of the DMA service request. After the DMA or CPU fills the buffer, the TFS flag (and the service request and/or interrupt) is automatically cleared.

Receive buffer Service Request Flag (RFS) (read-only, maskable interrupt)

The receive buffer service request flag (RFS) is a read-only bit which is set when the receive buffer is not empty and requires service. When the RFS bit is set, an interrupt request is made unless the receive buffer interrupt request mask (RIM) bit is cleared. The state of RFS is also sent to the DMA controller, and may be used to signal a DMA service request. Note that RIM has no effect on the generation of the DMA service request. After the DMA or CPU fills the buffer, the RFS flag (and the service request and/or interrupt) is automatically cleared.

Framing Error Status (FRE) (read/write, non-maskable interrupt)

The framing error status (FRE) bit is set when a frame alignment error is detected by the receive logic. A frame alignment error is detected on received data when a preamble is followed by something other than another preamble or a start flag.

Figure 12-14: FIr status register 0 bit locations shows the bit locations corresponding to the status and flag bits within FIr status register 0. Note that the reset state of all writable status bits is unknown and must be cleared (by writing a one to them) before enabling the FIr. Also note that writes to reserved bits are ignored and reads return zeros.

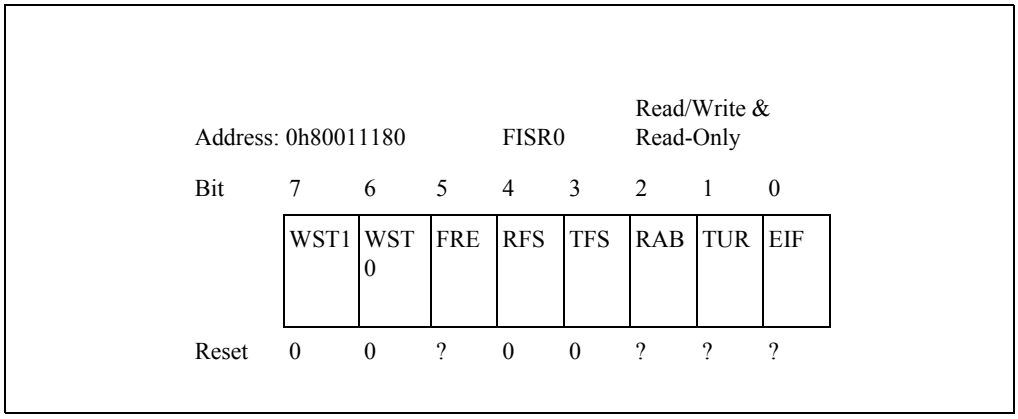


Figure 12-14: FIr status register 0 bit locations

Bit	Name	Description
0	EIF	Error in buffer 0 - Bits 32-36 are not set within either entry of the receive buffer 1 - One or more tag bits (32-36) are set within one or more entries of the receive buffer, request interrupt, disable receive buffer DMA service requests Note: once EIF is cleared, receive buffer DMA service requests are re-enabled
1	TUR	Transmit buffer Underrun 0 - Transmit buffer has not experienced an underrun 1 - Transmit logic attempted to fetch data from transmit buffer while it and the tail register were empty, interrupt request signalled if not masked (if TUS=1)
2	RAB	Receiver Abort 0 - No abort has been detected for the incoming frame 1 - Abort detected during receipt of incoming frame, two or more symbols containing no pulses (0000) detected on receive pin, EOF bit set in receive buffer next to last piece of “good” data received before the abort, interrupt requested

Table 12-32: FIr Status Register 0

Bit	Name	Description
3	TFS	Transmit buffer Service Request (read-only) 0 - Transmit buffer is full or the transmitter disabled 1 - Transmit buffer is not full and the transmitter is enabled. A DMA service request is signalled, interrupt request is signalled if not masked (if TIM=1)
4	RFS	Receive buffer Service Request (read-only) 0 - Receive buffer is empty or receiver disabled 1 - Receive buffer is not empty and receiver operation is enabled, DMA service request signalled, interrupt request signalled if not masked (if RIM=1)
5	FRE	Framing Error 0 - No framing errors encountered in the receipt of this data 1 - Framing error occurred, preamble followed by something other than another preamble or start flag, request interrupt
7–6	WST	Width Status 00 - All four bytes in receive buffer are valid 01 - Least significant byte valid only 10 - Least significant two bytes valid only 11 - Least significant three bytes valid only

Table 12-32: FIr Status Register 0 (Continued)

FIr Status Register 1

FIr status register 1 (FISR1) contains flags that indicate when the receiver is synchronized, the transmitter is active, that the transmit buffer is not full, that the receive buffer is not empty, and when an end of frame, CRC error, or underrun error has occurred. All bits within FISR1 are read-only and non-interrupting.

Receiver Synchronized Flag (RSY) (read-only, non-interrupting)

The receiver synchronized (RSY) flag is a read-only bit which is set when the receiver is synchronized with the incoming data stream, and is cleared when the receiver logic is in hunt mode (looking for the preamble to achieve byte and frame synchronization), or the receiver is disabled (RXE=0). This bit does not request an interrupt.

Transmitter Busy Flag (TBY) (read-only, non-interrupting)

The transmitter busy (TBY) flag is a read-only bit which is set when the transmitter is actively transmitting a frame (address, control, data, CRC, start or stop flag), and is cleared when the transmitter is idle (transmitting preambles), or the transmitter is disabled (TXE=0). This bit does not request an interrupt.

End of Frame Flag (EOF) (read-only, non-interrupting)

The end of frame flag (EOF) is set when the last byte of data within a frame (including aborted frames) resides within the receive buffer.

The receive buffer contains five tag bits (32 - 36) which are not directly readable. The 32nd bit is set at the top of the buffer whenever the last byte within a frame is moved from the receive serial shifter to the receive buffer. Each time a data value is transferred to the buffer, the state of the tag bit is moved to the EOF bit in the status register. Whenever EOF, EIF is set within FISR0, an interrupt is signalled, and the receive buffer DMA request is disabled.

CRC Error Status (CRE) (read-only, non-interrupting)

The CRC error flag (CRE) is set when the CRC value calculated by the receive logic does not match the CRC value contained within the incoming serial data stream.

Whenever a CRC error is detected, the 33rd bit is set within the receive buffer. Each time a data value is transferred to the buffer, the state of the tag bit is moved to the CRE bit in the status register, indicating whether or not the frame has encountered a CRC error. Whenever CRE is set, EIF is set within FISR0, an interrupt is signalled, and the receive buffer DMA request is disabled.

Receiver Overrun Status (ROR) **(read-only, non-interrupting)**

The receiver overrun flag (ROR) is set when the receive logic attempts to place data into the receive buffer after it has been completely filled.

The 34th bit is set within the top entry of the receive buffer whenever an overrun occurs. Each time a data value is transferred to the buffer, the state of the tag bit is moved to the ROR bit in the status register, indicating that the next value in the buffer is the last “good” piece of data before the overflow occurred. Whenever ROR is set, EIF is set within FISR0, an interrupt is signalled, and the receive buffer DMA request is disabled.

Figure 12-15: FIr status register 1 bits shows the location of the flags within FIr status register 1. The bits within this register are read-only and do not produce interrupt requests. Note that writes to bits 7,3 and 2 are ignored and reads return zero.

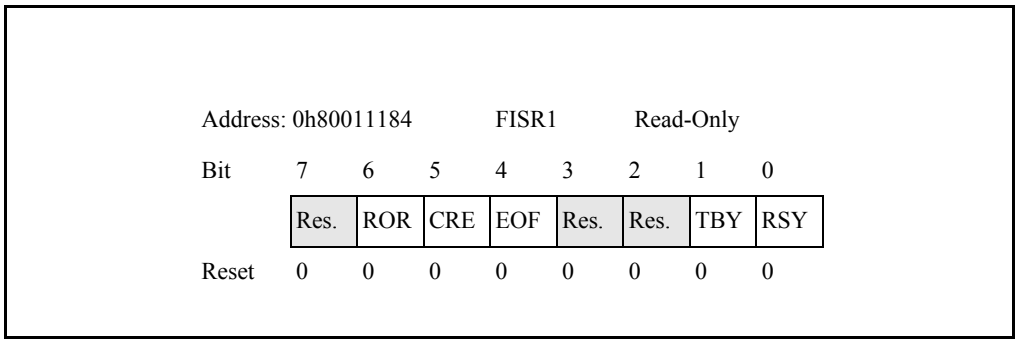


Figure 12-15: FIr status register 1 bits

Bit	Name	Description
0	RSY	Receiver Synchronized Flag (read-only) 0 - Receiver is in hunt more or is disabled 1 - Receiver logic is synchronized with the incoming data (no interrupt generated)
1	TBY	Receiver Synchronized Flag (read-only) 0 - Receiver is in hunt more or is disabled 1 - Receiver logic is synchronized with the incoming data (no interrupt generated)
2	-	Unused
3	-	Unused
4	EOF	End of Frame (read-only) 0 - Current frame has not completed 1 - The word in the receive buffer contains the last byte of data within the frame

Table 12-33: FIr Status Register 1

Bit	Name	Description
5	CRE	CRC Error (read-only) 0 - No CRC check errors encountered in the receipt of data 1 - CRC calculated on the incoming data does not match CRC value contained within the received frame
6	ROR	Receive buffer Overrun (read-only) 0 - Receive buffer has not experienced an overrun 1 - Receive logic attempted to place data into receive buffer while it was full, the next data value in the buffer is the last piece of "good" data before the buffer was overrun
7	-	Reserved

Table 12-33: FIr Status Register (Continued) 1

12.9.2 Interface Register Locations

The Ir Interface module registers occupy a 4k block of addresses within the fast APB peripheral area of the ARM 7201 memory map. The table below gives addresses of individual registers within that block relative to the Ir interface base address, specified elsewhere.

Address	Name	Description
0h 80011000	IrEnable	SIr/MIr/FIr Selector and Enable Register
0h 80011004	IrCon	Ir Interface Control register
0h 80011008	IrAmv	Ir Address Match Value Register
0h 80011010	IrData	Ir transmit and receive data FIFOs
0h 80011014 - 0h 8001101C	IrDataTail	Ir transmit Data Tail Register
0h 80011020 - 0h 8001107C	-	Reserved
0h 80011080	MISR0	MIr Status Register 0
0h 80011084	MISR1	MIr Status Register 1
0h 80011088 - 0h 8001117C	-	Reserved
0h 80011180	FISR0	FIr Status Register 0
0h 80011184	FISR1	FIr Status Register 1
0h 80011188 - 0h 80011FFF	-	Reserved

Table 12-34: Ir Interface Block Registers and their Physical Addresses

12.10 Universal Serial Bus

This section describes the implementation-specific options of USB protocol for a device controller. It is assumed that the user has a knowledge of the USB standard. This USB Device Controller(USBD) is chapter 9(of USB specification) compliant, and supports standard device requests issued by the host. The user should refer to *the Universal Serial Bus Specification revision 1.0* for a full understanding of the USB protocol and its operation. (The USB specification 1.0 can be accessed via the world wide web at: http://www.usb.org). The USBD is a universal serial bus device controller (slave, not hub or host controller) which supports three endpoints and can operate half-duplex at a baud rate of 12 Mbps. Endpoint 0,by default is only used to communicate control transactions to configure the USBD after it is reset or physically connected to an active USB host or hub. Endpoint 0's responsibilities include connection, address assignment, endpoint configuration and bus numeration.

The USBD is configured by the connected host which can get a device descriptor stored in USBD's internal ROM via endpoint 0. The USBD uses two separate 32 x 8 bit FIFOs to buffer receiving and transmitting data to/from the host. The FIFOs can be accessed by the DMAC (Direct Memory Controller), with service requests being signaled when either FIFO is full/ empty. The external pins dedicated to this interface are **UVPO, UVP, UVMO, UVM, URCVIN, nUSBOE** and **USUSPEND**.These signals should be connected to USB transceiver such as PDIUSBP11 provided by Philip Semiconductor. Refer to data sheet PDIUSBP11). The interface of the USBD and the CPU uses DMAC to reduce CPU load of transferring data from external memory to USBD and from USBD to external memory. The CPU can also access the USBD using Interrupt controller, by setting the control register appropriately. This section also defines the interface of USBD and CPU. The USBD uses one dedicated DMA channel for receiving and transmitting data, so the DMAC should be programmed into receiving channel initially for both data transferring. If transferring data to USB host occurs (setting the control register bit), that is, USB host issue IN Token, then DMAC should be programmed to transmitting channel. After transmitting data, DMAC should be programmed to the receiving channel again.

12.10.1Features

- Full universal serial bus specification 1.0 Compliance.
- Receiver and Transceiver have 32 bytes FIFO individually (this supports maximum data packet size of bulk transfer).
- Internal automatic FIFO control logic. (According to FIFO's status, the USBD generates DMA service request signals to DMAC or Interrupt service request signals to the CPU)
- Supports high-speed USB transfer (12Mbps).
- There are two endpoint of transmitter and receiver respectively, totally three endpoints including endpoint 0 that has responsibility of the device configuration.
- CPU can access the internal USB configuration ROM storing the device descriptor for Hand-held PC (HPC) by setting the predefined control register bit.
- USB protocol and device enumeration is performed by internal state-machine in the USBD.
- The USBD only supports bulk transfer of 4 transfer type supported by USB for data transfer.
- Endpoint FIFO (Tx, Rx) has the control logic preventing FIFO's overrun and underrun error.

12.10.2Block Diagram

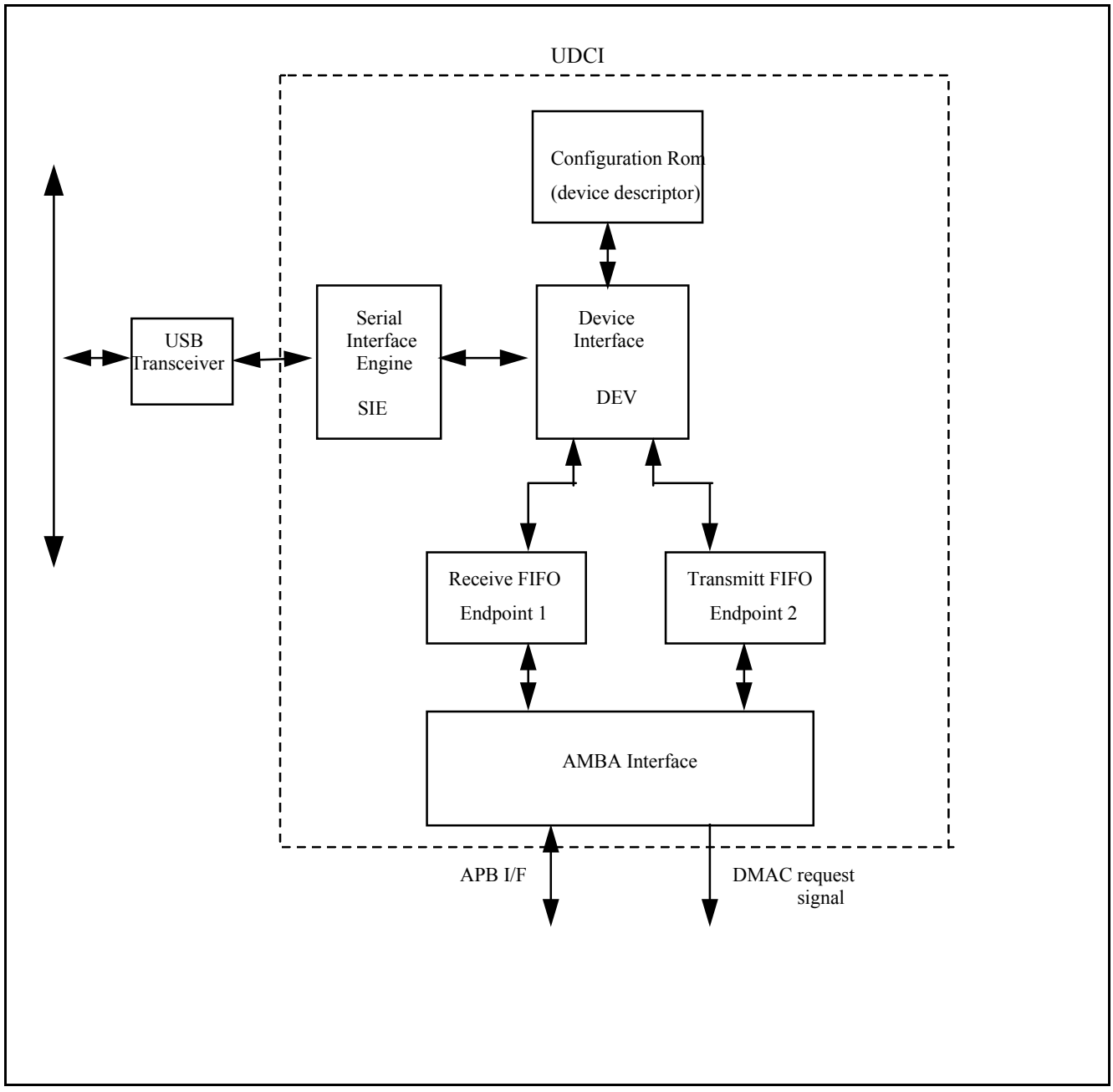


Figure 12-16: USBD Block Diagram

The USB, ***Figure 12-16: USBD Block Diagram***, comprises the Serial Interface Engine (SIE) and Device Interface (DEV). The SIE connects to the USB through a bus transceiver, and performs NRZI conversion, bit un-stuffing, CRC checking, packet decoding and serial to parallel conversion of the incoming data stream. In outgoing data, it does the reverse, that is, parallel to serial of outgoing data stream and packetizing the data, CRC generation, bit stuffing and NRZI generation.

The DEV provides the interface between the SIE and the device's endpoint FIFOs, ROM storing the device descriptor. The DEV handles the USB protocol, interpreting the incoming tokens and packets and collecting and sending the outgoing data packets and handshakes.The endpoints FIFO(RX,TX) give the information of their status (full/empty) to the AMBA interface to generate DMA request signal and AMBA I/F enable the CPU to access the FIFO's status register and the device descriptor stored in ROM. The AMBA interface generates a FIFO read/ write strobe without FIFO's errors, based on APB signal timing. Automatically it requests the DMA data handling when RX FIFO is full. In case of data transmitting through TX FIFO (when USB generates an OUT token, AMBA I/F generates Interrupt to CPU), the user should program the DMAC to transmitting channel, set the transmitting enable bit in the control register. If the error of FIFO (Rx: overrun, TX: underrun) occurs, the AMBA I/F cannot generate FIFO read/ write signals and DMA service request signals.

12.10.3Theory of Operation

The LGS USB Core enables a designer to connect virtually any device requiring incoming or outgoing PC data to the Universal Serial Bus. As illustrated in ***Figure 12-16: USBD Block Diagram*** on page 12-53, the USB core comprises two parts, the SIE and DEV. The SIE connects to the Universal Serial Bus via a bus transceiver. The interface between the SIE and the DEV is a byte-oriented interface that exchanges various types of data packets between two blocks.

Serial Interface Engine

The SIE converts the bit-serial, NRZI encoded and bit-stuffed data stream of the USB into a byte and packet oriented data stream required by the DEV. As shown in ***Figure 12-17: LGS Serial Interface Engine***, it comprises seven blocks: Digital Phase Lock Loop, Input NRZI decode and bit-unstuff, Packet Decoder, Packet Encoder, Output bit stuff and NRZI encode, Counters, and the CRC Generation & Checking block. Each of the blocks is described in the following sections.

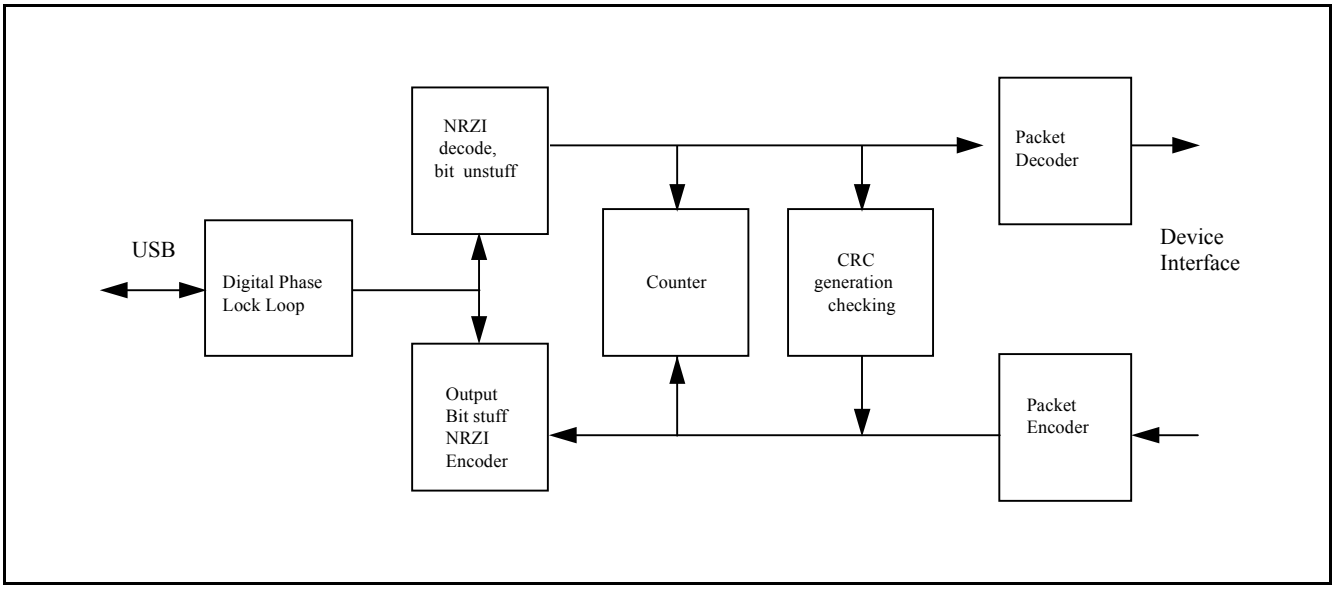


Figure 12-17: LGS Serial Interface Engine

Digital Phase Lock Loop

The Digital Phase Lock Loop module takes the incoming data signals from the USB, synchronizes them to the 48MHz input clock, and then looks for USB data transitions. Based on these transitions, the module creates a divide-by-4 clock called the **usbclock**. Data is then output from this module synchronous to the **usbclock**.

Input NRZI decode and bit-unstuff

The Input NRZI decode and bit-unstuff module extracts the NRZI encoded data from the incoming USB data. Transitions on the input serial stream indicate a 0, while no transition indicates a 1. Six ones in a row cause the transmitter to insert a 0 to force a transition, therefore any detected zero bit that occurs after six ones is thrown out.

Packet Decoder

The Packet Decoder module receives incoming data bits and decodes them to detect packet information. It checks that the PID (Packet ID) is valid and was sent without error. After decoding the PID, the remainder of the packet is split into the address, endpoint, and CRC5 fields, if present. The CRC Checker is notified to verify the data using the incoming CRC5 field. If the packet is a data packet, the data is collected into bytes and passed on with an associated valid bit. ***Table 12-35: Supported PID Types*** shows the PID Types that are decoded (marked as either Receive or Both). At the end of the packet, either the **packetok** or **packetnotok** signal is asserted. **Packetnotok** is asserted if any error condition arose (bad valid bit, bit-stuff, bad PID, wrong length of a field, CRC error, etc.).

PID Type	Value	Send/Receive
OUT	4'b0001	Receive
IN	4'b1001	Receive
SOF	4'b1101	Receive
SETUP	4'b0000	Receive
DATA0	4'b0011	Both
DATA1	4'b1011	Both
ACK	4'b0010	Both
NAK	4'b1010	Send
STALL	4'b1110	Send
PRE	4'b1100	Receive

Table 12-35: Supported PID Types

Packet Encoder

The Packet Encoder creates outgoing packets based on signals from the DEV. ***Table 12-35: Supported PID Types*** shows the PID Types that can be encoded (marked as Send or Both). For each packet type, if the associated signal **sendtype** is received from the DEV, the packet is created and sent. Upon completion of the packet, **packettypesent** is asserted to inform the DEV of the successful transmission. The Packet Encoder creates the outgoing PID, grabs the data from the DEV a byte at a time, signals the CRC Generator to create the CRC16 across the data field, and then sends the CRC16 data. The serial bits are sent to the Output bit stuff and NRZI encoder.

Output bit stuff and NRZI encoder

The Output bit stuff and NRZI encoder takes the outgoing serial stream from the Packet Encoder, inserts stuff bits (a zero is inserted after six consecutive ones), and then encodes the data using the NRZI encoding scheme (zeroes cause a transition, ones leave the output unchanged).

Counter block

The Counter block tracks the incoming data stream in order to detect the following conditions: reset, suspend, and turnaround. It also signals to the transmit logic (Output NRZI and bit stuff) when the bus is idle so transmission can begin.

Generation and Checking block

The Generation and Checking block checks incoming CRC5 and CRC16 data fields, and generates CRC16 across outgoing data fields. It uses the CRC polynomial and remainder specified in the USB Specification Version 1.0.

Device Interface

The DEV shown in ***Figure 12-18: Device Interface*** works at the packet and byte level to connect a number of endpoints to the SIE. It understands the USB protocol for incoming and outgoing packets, so it knows when to grab data and how to correctly respond to incoming packets. A large portion of the DEV is devoted to the setup, configuration, and control features of the USB. As shown in ***Figure 12-18: Device Interface*** the DEV is divided into three blocks: Device Controller, Device ROM, and Start of Frame. The three blocks are described in the following sections.

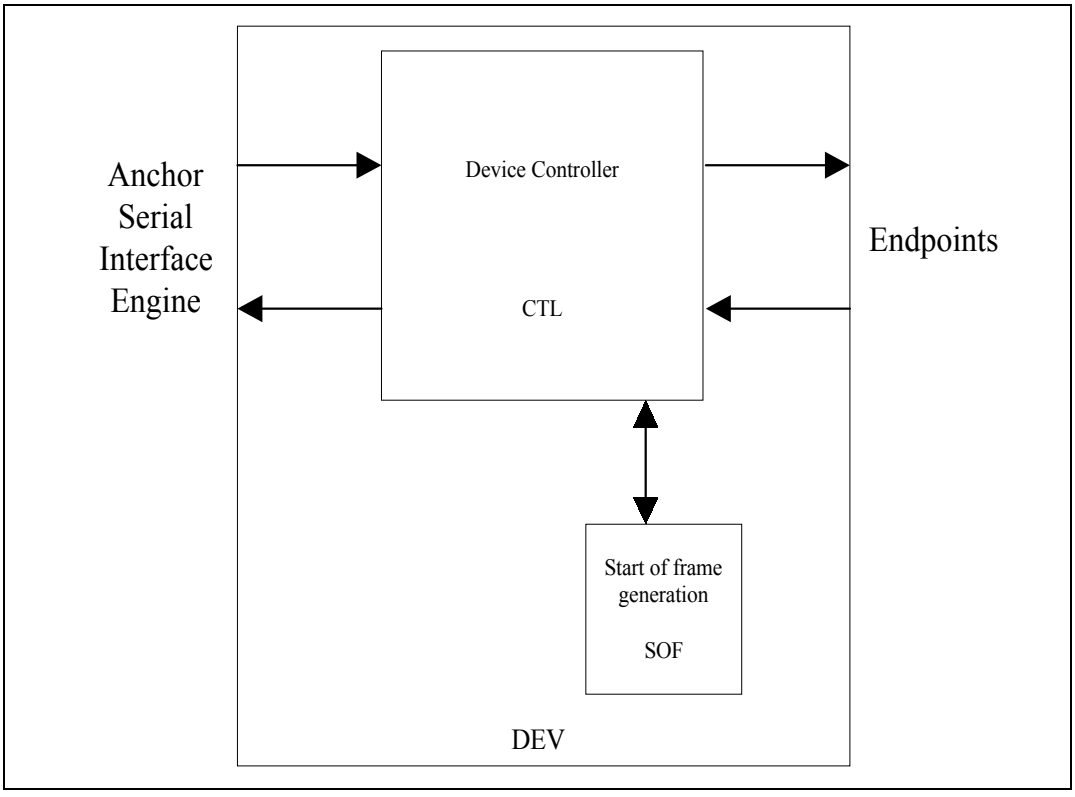


Figure 12-18: Device Interface

Device Controller

The Device Controller contains a state machine that understands the USB protocol. The (SIE) provides the Device Controller with the type of packet, address value, endpoint value, and data stream for each incoming packet. The Device Controller then checks to see if the packet is targeted to the device by comparing the address/endpoint values with internal registers that were loaded with address and endpoint values during the USB enumeration process. Assuming the address/endpoint is a match, the Device Controller then interprets the packet. Data is passed on to the endpoint for all packets except SETUP packets, which are handled specially. Data toggle bits (DATA0 and DATA1 as defined by the USB spec) are maintained by the Device Controller. For IN data packets (device to host) the Device Controller sends either the maximum number of bytes in a packet or the number of bytes available from the endpoint. All packets are acknowledged as per the spec. For SETUP packets, the incoming data is extracted into the relevant internal fields, and then the appropriate action is carried out. ***Table 12-36: Supported Setup Requests*** lists the types of setup operations that are supported.

Setup Request	Value	Supported
Get Status	0	Device, Interface, Endpoint
Clear Feature	1	Endpoints Only
Set Feature	3	Not supported
Set Address	5	Device
Get Descriptor	6	Device
Set Descriptor	7	Not supported
Get Configuration	8	Device
Set Configuration	9	Device
Get Interface	10	Not supported
Set Interface	11	Not supported
Synch Frame	12	Not supported

Table 12-36: Supported Setup Requests

Start of Frame

The Start of Frame logic generates a pulse whenever either the incoming Start of Frame (SOF) packet arrives or approximately 1 ms after it the last one arrived. This allows an isochronous endpoint to stay in sync even if the SOF packet has been garbled.

12.10.4Endpoint FIFOs (Rx,Tx)

Each endpoint FIFO has the specific number of FIFO depth according to data transfer rate. In case of maximum packet size for bulk transfer is 32 bytes that is supported in USBD. Each FIFO generates dataready signals (means FIFO not full or FIFO not empty) to AMBA I/F and causes AMBA I/F to produce DMAC request signals. It contains the control logic for transferring 4 bytes at a read/write strobe generated by AMBA to obtain better efficiency of AMBA bus.

12.10.5AMBA Interface

The AMBA I/F performs the decoding APB signal, generating DMA request signal, comparing with endpoint FIFO status signal. And it also prevents FIFOs overrun/underrun error. There are 5 registers

CONT1,CONT2,RXDATA,TXDATA,STATUS. The section of 1.7 describes these register fully.

12.10.6Pinout

The pin definitions for the LGS USB are divided into four groups: USB connections, clock connections, endpoint connections, and configuration ROM connections. A diagram of the pinout is shown in ***Figure 12-19: Block Diagram of the USB Core Pinout***.

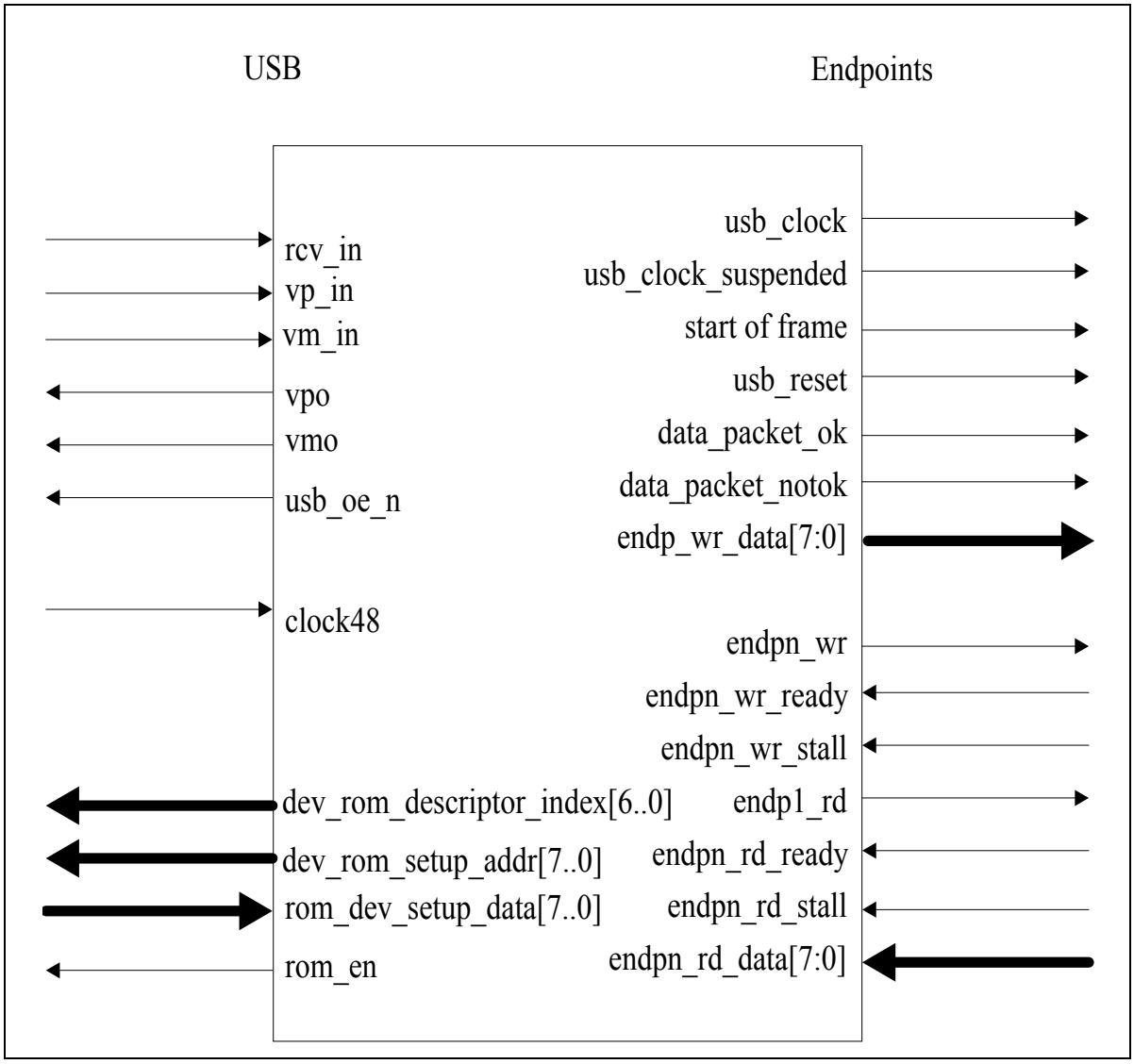


Figure 12-19: Block Diagram of the USB Core Pinout

12.10.7. USBD Signal Descriptions

Signals when the USBD is connected to the APB.

Name	Type	Description
PA[5:2]	IN	The fast APB address signal,
PD[31:0]	INOUT	The fast APB data signal
PSTB	IN	The fast APB data strobe signal
PWRITE	IN	The fast APB read/write signal
PSELusb	IN	The fast APB select signal
BRES	IN	The system reset signal (active LOW)
BCLK	IN	The system clock
PSELdmausb	IN	The fast APB usb select signal on dma access

Table 12-37: APB Address Signals

Signals when the USBD is connected to the DMAC and INTC (Interrupt USB signals

Name	Type	Description
DREQ	OUT	The DMAC request signal (Active HIGH)
INTREQ	OUT	The INTC request signal (Active HIGH)

Table 12-38: DREQ and INTREQ Signals

USB Signals

The following signals connect to a USB compatible transceiver. If the target technology includes a USB compatible pad cell, then the number of USB interface signals can be reduced to the two USB data signals **D+** and **D-**.

Signal Name	Type 1	Description
URCVIN	in	***receive in.*** This is the USB differential input signal. It comes from a differential receiver connected to USB signals **D+** and **D-**.
UVP	in	***vplus in.*** This is the USB **D+** signal received through a standard CMOS receiver. The SIE uses this signal to detect the single-ended zero (SE0) bus state.
UVM	in	***vminus in.*** This is the USB **D-** signal received through a standard CMOS receiver. The SIE uses this signal to detect the single-ended zero (SE0) bus state.
UVPO	out	***vplus out.*** This pin drives the non-inverting (+) input of the differential buffer that drives the **D+**/**D-** USB signals.

Table 12-39: DSB Signal Names

UVMO	out	***vminus out.*** This pin drives the inverting (-) input of the differential buffer that drives the **D+**/**D-** USB signals.
nUSBOE	out	***usb output enable.*** When LOW, this pin enables the output drivers for the **D+**/**D-** USB signals.
USUSPEND	out	***USB low power mode***

Table 12-39: DSB Signal Names

The signal type definitions are defined here as follows:

in Input is a standard input-only signal

out Totem Pole Output is a standard active driver

Clock signal

Signal Name	Type	Description
CCLK	in	**CCLK.** This pin is driven by an internal 48MHz PLL. The digital phase-lock loop within SIE uses this signal to capture the USB data, as well as divide it down to provide a 12MHz clock for the rest of the core and device. This signal should have a 40%-60% duty cycle, and is positive-edge sensitive.

Table 12-40: Clock Signal 48

12.10.8Internal Registers

Table 12-41: USB register address map summarizes the USB internal registers.

Address	Name	Description
Base address + 0x00	CONT0	USB I/F control register 0
Base address + 0x04	CONT1	USB I/F control register 1
	RXDATA	USBD receive data register
	TXDATA	USBD transmit data register
Base address + 0x08	STATUS	USBD status register
Base address + 0x0C	TicRXDATA	USBD receive data register for TIC mode
Base address + 0x10	TicTXDATA	USBD transmit data register for TIC mode
Base address + 0x14	TicSEL	TIC mode select register
Base address + 0x18	TicREG	Input TIC register for TIC mode
Base address + 0x1C	TicRESULT	Output TIC register for TIC mode
Base address + 0x20	CONTswreset	Generate software reset to USBD
Base address + 0x24	CONTdrqmask	DMAC request masking register

Table 12-41: USB register address map

CONT0 (USB I/F control register0) (Base address +0x00)

Bit	7	6	5	4	3	2	1	0
	RESERVED							TXEN

Figure 12-20: Transmit Enable Bit for CONT0

Name	Type	Description
TXEN	Transmit enable bit	When 1, the function of USB is transmitting data from external memory to USB host PC. When '0', receiving data from USB host PC to external memory; default = 0. Before setting this bit 1, the user should program the DMAC to transmitting channel. If it is 1, AMBA I/F generate DMAC request signal for data transmitting after checking if Tx FIFO's status is empty.

Table 12-42: Transmit Enable Bit States

CONT1 (USB I/F control register1) (Base address +0x04)

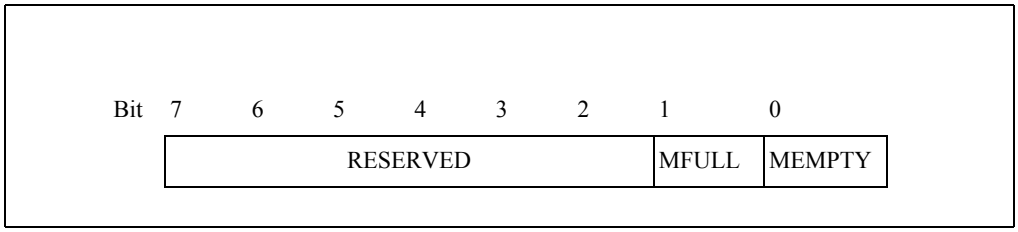


Figure 12-21: FIFO Mask Bits for CONT1

MFULL: Mask FIFO full interrupt bit. When it is '1' the FIFO full interrupt is masked.

MEMPTY: Mask FIFO empty interrupt bit.When it is '1', the FIFO empty interrupt is masked.

RXDATA (USBD Receive Data Register

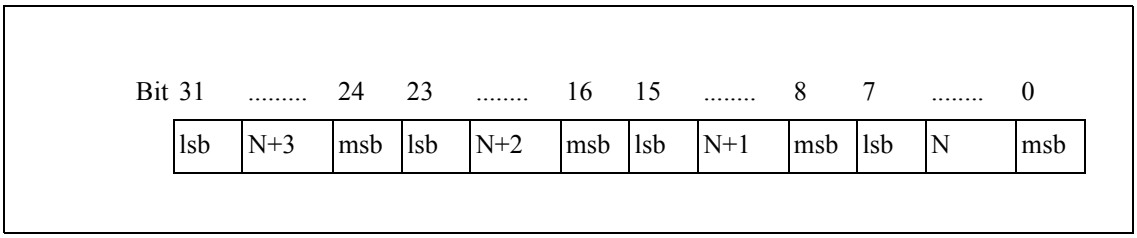


Figure 12-22: USBD Receive Data Register Bits

32-bit receive data register. DMAC reads 4 bytes at data read strobe. Each byte is presented lsb first. N is represented as Rx FIFO address.

TXDATA (USBD Transmit Data Register)

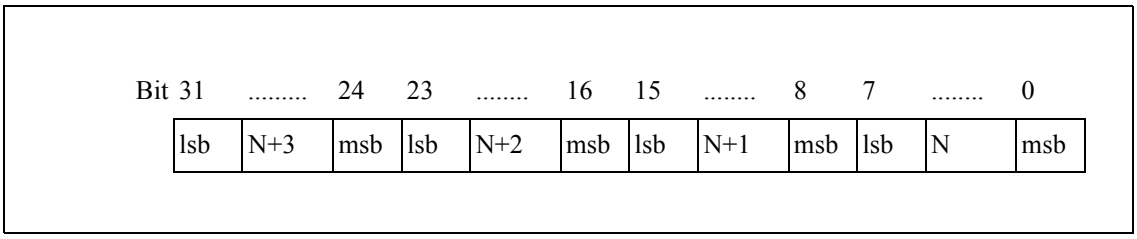


Figure 12-23: USBD Transmit Data Register Bits

32-bit transmit data register. DMAC writes 4 bytes at data write strobe. Each byte is presented lsb first. N is represented as Tx FIFO address.

STATUS (USBD status register) (Base address + 0x08)

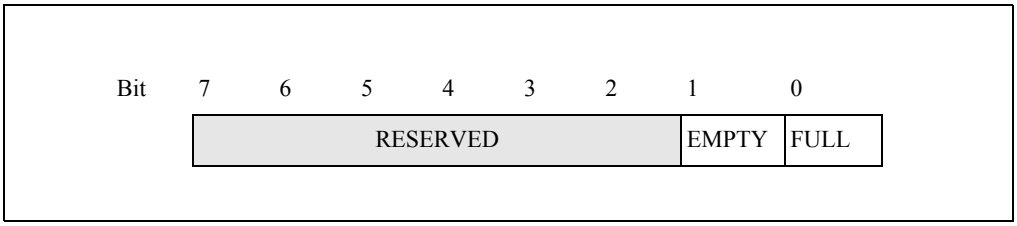


Figure 12-24: USBD Status Register Bits

FULL: this bit indicates that Rx FIFO is filled. When it is 1, automatically, AMBA I/F generates a DMAC request signal. DMAC can read the RxDATA register according to read strobe.

EMPTY: this bit indicates that Tx FIFO is empty when the the CONT0 TXEN bit is set to 1.

TicRXDATA (USBD receive data register for TIC mode) (Base address +0x0C)

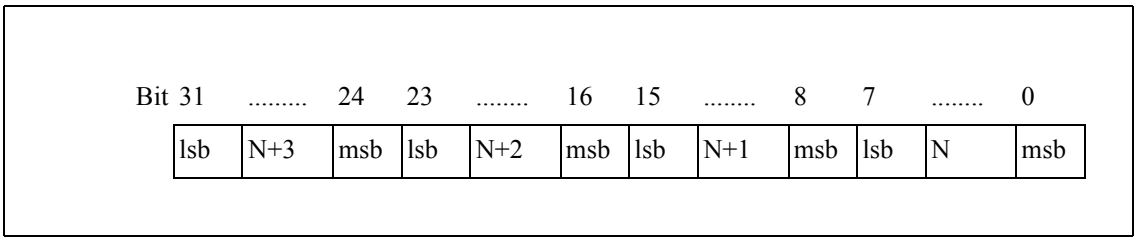


Figure 12-25: TIC Mode Receive Data Register Bits

32-bit test receive data register for TIC mode. TIC reads 4 bytes at data read strobe. each byte is presented lsb first. N is represented as Rx FIFO address.

TicTXDATA (USBD transmit data register for TIC mode) (Base address +0x10)

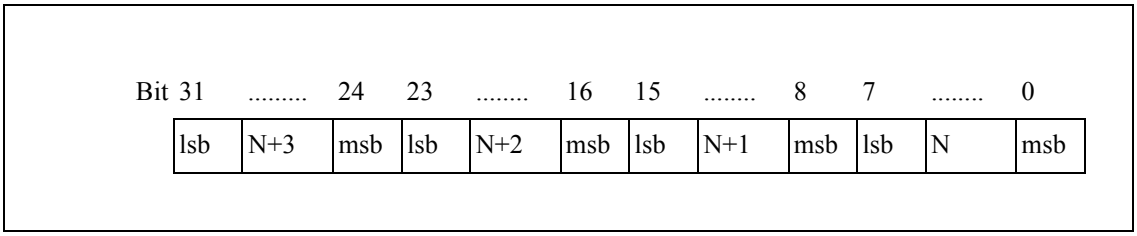


Figure 12-26: TIC Mode Transmit Data Register Bits

32-bit transmit data register for TIC mode. TIC writes 4 bytes at data write strobe. Each byte is presented lsb first. N is represented as Tx FIFO address.

TicSEL (The TIC mode select register) (Base address +0x14)

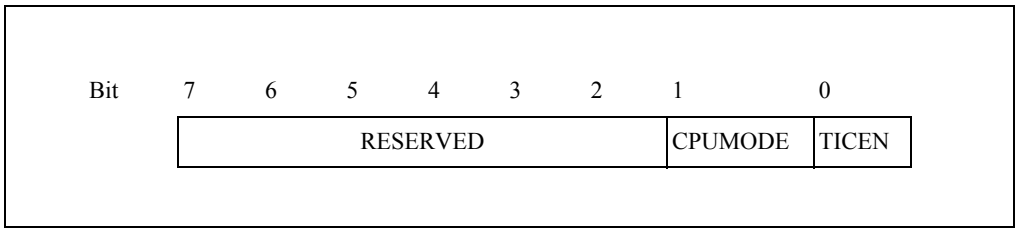


Figure 12-27: TIC Mode Select Register Bit

CPUMODE: CPU access mode enable bit. When CPUMODE = 1, the USBD is allowed to enter the CPU access mode. For DMAC mode, this bit should be 0.

TICEN: TIC mode enable bit. When TICEN =1, USBD enters the TIC mode.

TicREG (The input TIC register for TIC mode) (Base address+0x18)

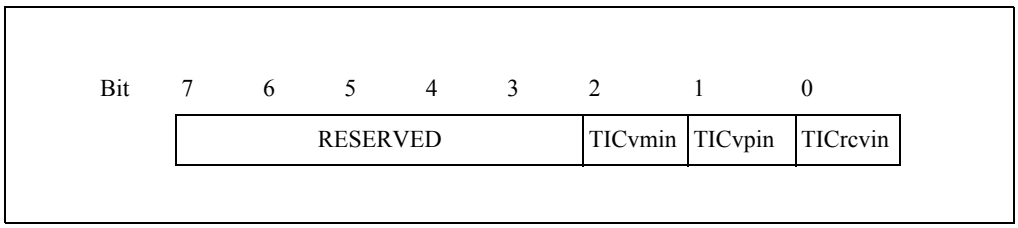


Figure 12-28: Input TIC register for TIC mode

TICvmin, TICvpin, TICrcvin: this value is set to the TIC vector by TIC. These bits are set based on the 48MHz clock signal generated by TIC.

TicRESULT (The output TIC register for TIC mode) (Base address+0x1c)

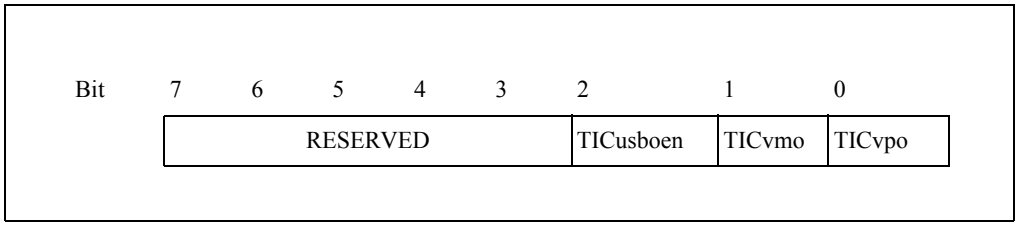


Figure 12-29: Output TIC Register for TIC mode

TICusboen, TICvmo, TICvpo: the values of these bits are compared with the TIC vector by TIC, based on the 48MHz clock signal generated by TIC.

CONTswreset (generate software reset to USBD) (Base address+0x20)

Figure 12-30: Generate software reset to USBD

ESWRES: this bit should be set to ‘1’ after PMU generated 48 Mhz USBD clock. If this bit is ‘1’, the dpll (digital pll) in the USBD is initialized and so USBD operates successfully. After data transfer has finished, the bit should be set to ‘0’.

CONTdrqmask (USBD DMAC request masking register) (Base address+0x24)

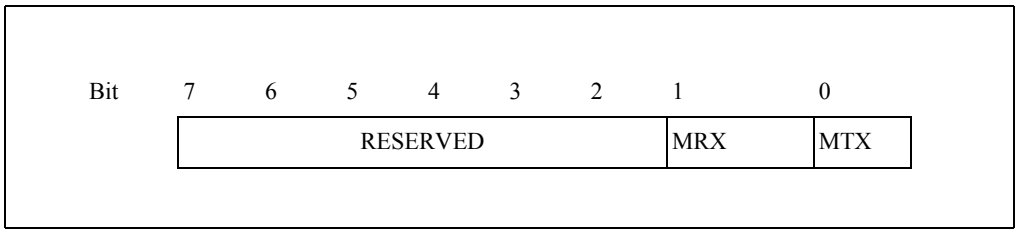


Figure 12-31: DMAC request masking register

In CPU access mode, the DMAC request signal should be masked.

MTX: Mask FIFO empty DMAC request signal
When it is '1', FIFO empty DMAC request signal is masked.

MRX: Mask FIFO full DMAC request signal.
When it is '1', FIFO fullDMAC request signal is masked.

DMAC I/F

This field describes the interface of the DMAC and the USBD. The USBD (Rx/Tx buffer) transmits and receives data from and to the DMAC based on the signal of the fast APB. That is, after generating the DMA request signal, USBD expects the DMAC to produce PSELdmausb, PD, PSTB and PWRITE that are the fast APB signals. With these signals, the FIFOs in Rx buffer put data to PD and the FIFOs in Tx buffer get data from PD through the Quad Word Access. Refer to the timing diagrams in ***12.2.14 DMA Transfer Flow*** on page 12-14.

12.10.9Timing Values

Figure 12-4: Address match value field in the IrAmv Register on page 12-22 and ***Figure 12-5: Bit locations within the Ir Data Register*** on page 12-23 give details of timing values.

12.11 Sound Interface

The Sound Control Unit (SOC) is an interface block to transfer sound data to external speakers. It possesses the following features:

- sound playback
- supports programmable sampling rate
- 32-bit internal data register for DMA
- auto DMA request
- 8-bit resolution DAC control
- supports non-overlapping left/right signal for DAC
- supports test mode

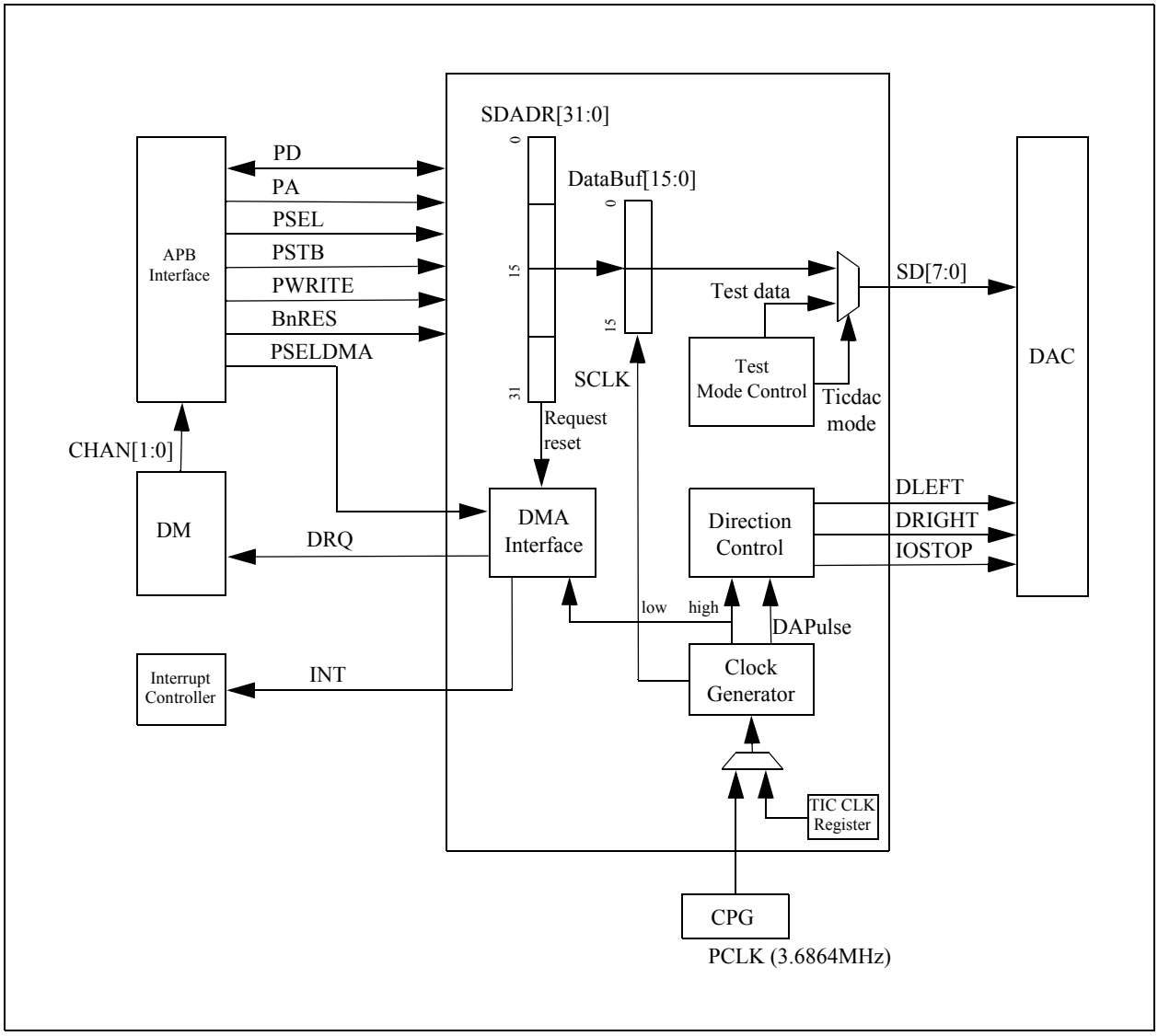


Figure 12-32: Sound control unit module block diagram

12.11.1 Hardware interface and signal description

The SOC module is connected to the internal APB bus.

Name	Type	Source/Destination	Description
PCLK	In	Clock controller	UART clock (3.6864MHz).
BnRES	In	APB Bridge	Reset signal generated from the APB Bridge.
PA[4:2]	In	APB Bridge	This is the peripheral address bus, which is used by an individual peripheral for decoding register accesses to that peripheral. The addresses become valid before PSTB goes HIGH, and remain valid after PSTB goes LOW.
PD[31:0]	InOut	APB Peripherals, BD bus	This is the bidirectional peripheral data bus. The data bus is driven by this block during read cycles (when PWRITE is LOW).
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of PSTB is coincident with the falling edge of BCLK.
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral. When LOW, it indicates a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before PSTB goes HIGH, and remains valid after PSTB goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates that this module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB). For more details, see *AMBA Peripheral Bus Controller* (ARM DDI0044).
PSELDMA	In	APB Bridge	Active HIGH signal provided by the APB Bridge to indicate SoundC DMA access.
DRQ	Out	DMA	When SD transfers upper 16-bit, this signal requests more sound data for the DAC with active HIGH until writing the data at SDADR.
INT	Out	Interrupt Controller	When SD transfer upper 16-bit, this signal requests more sound data for the DAC with active HIGH until writing the data at SDADR.
SD[7:0]	Out	DAC	DAC data bus. During SCLK HIGH, it is upper 8-bit of DataBuf, and during LOW, the lower 8-bit of DataBuf.
IOSTOP	Out	DAC	When HIGH, analog circuits in DAC go to rail-to-rail to save power dissipation. If inactive LOW, the analog circuit in the DAC operates in normal mode.
DLEFT	Out	DAC	When HIGH, this signal indicates that converted left data out is stable in DAC. Left/right signal is non-overlapping signal.
DRIGHT	Out	DAC	When HIGH, this signal indicates that converted right data out is stable in DAC.

Table 12-43: APB signal descriptions

12.11.2 Sound control unit operation

The SOC is an interface block used to send data to the external speaker through the internal 8-bit DA converter. It can process 44.1/22.05/11.025/8KHz sampled 8-bit mono or 16-bit stereo sound data.

This unit has a 32-bit register to receive sound data from the CPU through DMA or interrupt mode. This unit requests the DMA or interrupt controller every 32-bit processing time, which depends on the sampling frequency. It has two separate signals for DAC which indicate the direction of data for the stereo sound. Either higher or lower byte of 16-bit stereo sound data can be played through the left or right speaker by programming the control register. During mono playback, this unit sends the same data for the left and right channels.

There are two test registers. Both these registers should be cleared during normal operation. TICCLK port is also assigned for production test only.

12.11.3 Sound control unit memory map

The base address of the SOC is variable, and the offset of any particular register from the base address is fixed.

Address	Read Location	Write Location
SOC Base + 0x00	SCONT[7:0]	SCONT
SOC Base + 0x04	SDADR[31:0]	SDADR
SOC Base + 0x08	STOR[17:0]	STOR
SOC Base + 0x0C	STIR[14:0]	
SOC Base + 0x10	TICCLK	TICCLK

Table 12-44: Sound control unit register memory map

12.11.4 Sound control unit Register Descriptions

The following registers are provided for the SOC:

- Control Register (SCONT)
- Data Register (SDADR)
- Test Output Register (STOR)
- Test Input Register(STIR)

Refer to ***Table 12-45: SCONTRL bit description*** on page 12-70, ***Table 12-46: SDADR bit description*** on page 12-71,

Control Register (SCONT)

Bit	Initial Value	Description
7	0	0 - stereo 1 - mono
6*	0	DMA request masking bit 0 - masking 1 - unmasking
5	0	This bit should be cleared to minimize power consumption when not in use. 0 - power down mode 1 - normal mode
4	0	DAC operation enable/disable. During disabled, DAC is in power save mode. 0 - DAC disable 1 - DAC enable
3	0	When cleared, lower byte data goes to left speaker. (ADAC [1] pin) 0 lower byte data goes to ADAC [1] pin 1 lower byte data goes to ADAC [0] pin
2–1	2'b0	Programmable sampling rate 00 - 11.025KHz 01 - 22.05KHz 10 - 44.1KHz 11 - 8KHz
0*	0	Interrupt request masking bit 0 - masking 1 - unmasking

Table 12-45: SCONTRL bit description

Note *Those bits marked with an asterisk should not be enabled simultaneously during normal operation. (The programmer can select only one—either Interrupt or DMA mode.)*

Data Register (SDADR)

This register can be programmed after setting Bit 5 of the SCONT register.

Bit	Initial Value	Description
31 (MSB) 30 (data) : : 1 (data) 0 (LSB)	32'b0	Sound Data This register receives data by DMA Controller or CPU. This unit processes the lower 16-bit data followed by the higher 16-bit data. After the lower 16-bit is processed, this unit is ready to receive new data and sends a request signal to DMA Controller or CPU. In mono mode, the lower byte is processed first followed by the higher byte.

Table 12-46: SDADR bit description

Test Output Register (STOR)—programmable register

This register is used for the operation of DAC. This register should only be used for DAC test purposes, and should not be accessed during normal operation.

Bit	Initial Value	Description
17	0	When set, TICCLK is used as clock source instead of normal clock input during production test: 0 - normal mode 1 - TICCLK mode
16	0	Only if set, the values of bit 15 and bit 14 replace the original `Soundclk` and `Dapulse` signal. Used only for test purposes. 0 - normal mode 1 - test mode
15	0	Soundclk signal input - see Note 1. When bit 16 is set, this bit is meaningful. The `Soundclk` signal is changed by the value of this bit.
14	0	Dapulse signal input - see Note 2. When bit 16 is set, this bit is meaningful. The `Dapulse` signal is changed by the value of this bit.
13	0	Ticdac mode for DAC test. In this mode, PCLK is changed by TICCLK register: 0 - normal mode 1 - Ticdac mode

Table 12-47: **Test Output Register (STOR)—programmable register**

Bit	Initial Value	Description
12	0	When only TICdac mode, this bit can be programmed: 0 - interrupt not request 1 - interrupt request
11	0	When only TICdac mode, this bit can be programmed: 0 - DMA not request 1 - DMA request
10	0	When only TICdac mode, this bit can be programmed: 0 - DAC operation run 1 - DAC operation stop
9	0	When only TICdac mode, this bit can be programmed: 0 - DLEFT LOW 1 - DLEFT HIGH
8	0	When only TICdac mode, this bit can be programmed: 0 - DRIGHT LOW 1 - DRIGHT HIGH
7–0	0	SD signal. When only TICdac mode, this bit can be programmed.

Table 12-47: **Test Output Register (STOR)—programmable register (Continued)**

Notes *(1) Soundclk: this is an internal signal used as a reference clock source to play the sound data.*
(2) Dapulse: this is an internal signal used to make the DAC channel select signal.

Test Input Register (STIR)—read-only register

This register is for monitoring the unit status and the signals to DAC in both normal and Ticdac mode.

Bit	Initial Value	Description
14	0	Soundclk state. This bit indicates the state of Soundclk signal.
13	0	Dapulse state. This bit indicates the state of Dapulse signal.
12	0	This bit is set by INT signal in Ticdac and normal mode: 0 - interrupt not request 1 - interrupt request
11	0	This bit is set by DRQ signal in Ticdac and normal mode. 0 - DMA not request 1 - DMA request
10	0	This bit is set by IOSTOP signal in Ticdac and normal mode. 0 - DAC operation run 1 - DAC operation stop
9	0	This bit is set by DLEFT signal in Ticdac and normal mode. 0 - DLEFT LOW 1 - DLEFT HIGH

Table 12-48: **Test Input Register(STIR)—Read-only Register**

Bit	Initial Value	Description
8	0	This bit is set by DRIGHT signal in Ticdac and normal mode. 0 - DRIGHT LOW 1 - DRIGHT HIGH
7–0	0	SD signal. This bit is set by SD signal in Ticdac and normal mode.

Table 12-48: **Test Input Register(STIR)—Read-only Register (Continued)**

TIC Clock Register(TICCLK)

Whenever this register is accessed, TICCLK is generated. When STOR[17] bit is set, TICCLK is used as clock source instead of normal clock input.

13 Slow AMBA Peripherals

Slow AMBA Peripherals

13.1 Introduction

This chapter describes the peripherals that are connected to the 3.68MHz internal peripheral bus; these are peripherals that need relatively low data rates on the internal bus. These peripherals are not supported by the on-chip DMA controller (but can be supported by 'virtual DMA'—that is, software).

Further details on the internal bus are included in ***Chapter 3, Architecture Overview***.

13.1.1 Slow AMBA peripherals register map summary

Table 13-1: Slow AMBA peripherals register map gives a summary of base addresses, names and descriptions of the slow AMBA peripherals.

AMBA Base Address (Hex)	Name	Description
2Gbyte + 0x20000	U1Base	UART 1
2Gbyte + 0x21000	U2Base	UART 2
2Gbyte + 0x22000	KBDBase	KBD
2Gbyte + 0x23000	GPIOBase	GPIO
2Gbyte + 0x24000	INTCBase	INTC
2Gbyte + 0x25000	TimerBase	TIMER
2Gbyte + 0x26000	SPIBase	SPI
2Gbyte + 0x27000	ModemBase	MODEM
2Gbyte + 0x28000	RTCBase	RTC
2Gbyte + 0x29000	ADCBase	ADC

Table 13-1: Slow AMBA peripherals register map

13.2 UART

13.2.1 General description

The 16C550 is a Universal Asynchronous Receiver/Transmitter (UART), with FIFOs, and is functionally identical to the 16450 on power-up (CHARACTER mode). The 16550 can be put into an alternate mode (FIFO mode) to relieve the CPU of excessive software overhead. In this mode internal FIFOs are activated, allowing 16 bytes plus 3 bit of error data per byte in the RCVR FIFO, to be stored in both receive and transmit modes. All the logic is on the chip to minimize the system overhead and to maximize efficiency.

The UART performs serial-to-parallel conversion on data characters received from a peripheral device or a MODEM, and parallel-to-serial conversion on data characters received from the CPU. The CPU can read the complete status of the UART at any time during the functional operation. Status information reported includes the type and condition of the transfer operations being performed by the UART, as well as any error conditions (parity, overrun, framing, or break interrupt).

The UART includes a programmable baud rate generator capable of dividing the timing reference clock input by divisors of 1 to 2^{16}–1, and producing a 16x clock for driving the internal transmitter logic. Provisions are also included to use this 16x clock to drive the receiver logic.

The UART has complete MODEM-control capability, and a processor-interrupt system. Interrupts can be programmed to the user's requirements, minimizing the computing required to handle the communications link.

13.2.2 Features

- Capable of running all existing 16450 software.
- After reset, all registers are identical to the 16450 register set.
- The FIFO mode transmitter and receiver are each buffered with 16 byte FIFOs to reduce the number of interrupts presented to the CPU.
- Adds or deletes standard asynchronous communication bits (start, stop and parity) to or from the serial data.
- Holding and shift registers in the 16450 mode eliminate the need for precise synchronization between the CPU and serial data.
- Independently-controlled transmit, receive, line status and data set interrupts.
- Programmable baud generator divides any input clock by 1 to 65535 and generates 16x clock
- Independent receiver clock input.
- MODEM control functions (CTS, RTS, DSR, DTR, RI and DCD).
- Fully programmable serial-interface characteristics:
  - 5-, 6-, 7- or 8-bit characters
  - Even, odd or no-parity bit generation and detection
  - 1-, 1.5- or 2-stop bit generation and detection
  - Baud generation (DC to 230k baud)
- False start bit detection.
- Complete status reporting capabilities.
- Line break generation and detection.
- Internal diagnostic capabilities:
  - Loopback controls for communications link fault isolation
- Full prioritized interrupt system controls.

13.2.3 Signal description

The 16C550 UART module is connected to the internal APB bus.

Name	Type	Source/ Destination	Description
PCLK	In		UART Clock input This connects the main timing reference to the UART. 3.6864Mhz is input clock frequency recommended.
BnRES	In	APB Bridge	Reset signal generated from the APB Bridge (Master Reset) When this input is LOW, it clears all the registers (except the Receiver Buffer, Transmitter Holding and Divisor Latches) and the control logic of the UART. The states of various output signals (**SOUT**, **INTUART**, **NRTS**, **NDTR**) are affected by an active BnRES input.
PA[5:2]	In	APB Bridge	Register select. Address signals connected to these three inputs select a UART register for the CPU to read from or write to during data transfer. A table of registers and their addresses is shown below (***Table 13-6: Summary of registers*** on page 13-10).
PD[7:0]	InOut	APB Bridge	Data Bus. This bus comprises eight TRI-STATE input/output lines. The bus provides bi-directional communications between the UART and the CPU, Data, control words and status information are transferred via the PD[7:0] data bus.
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of **PSTB** is coincident with the falling edge of **BCLK** (ASB System Clock).
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral. When LOW, it indicates a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates that this module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB).
INTUART	Out	INTC	Interrupt. This pin goes HIGH whenever any one of the following interrupt types has an active HIGH condition and is enabled via IER: Receiver Error Flag Received Data Available:timeout(FIFO Mode only) Transmitter Holding Register Empty MODEM Status The INTUART signal is reset LOW upon the appropriate interrupt service or a Master Reset operation.
SIN	In	External	Serial Input. Serial data input from the communications link (peripheral device, MODEM or data set).

Table 13-2: Signal descriptions

Name	Type	Source/ Destination	Description
NCTS	In	External	Clear to Send. When LOW, this indicates that the MODEM or data set is ready to exchange data. The **NCTS** signal is a MODEM status input whose conditions can be tested by the CPU reading bit 4 (CTS) of the MODEM Status Register indicates whether the **NCTS** input has changed state since the previous reading of the MODEM Status Register. NCTS has no effect on the Transmitter. Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status interrupt is enabled.
NDSR	In	External	Data Set Ready. When LOW, this indicates that the MODEM or data set is ready to establish the communications link with the UART. The **NDSR** signal is a MODEM status input whose conditions can be tested by the CPU reading bit 5 (DSR) of the MODEM Status Register. Bit 5 is the complement of the **NDSR** signal. Bit 1(DDSR) of MODEM Status Register indicates whether the **NDSR** input has changed state since the previous reading of the MODEM status register. Note: Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status interrupt is enabled.
NDCD	In	External	Data Carrier Detect. When LOW, indicates that the data carrier has been detected by the MODEM data set. The signal is a MODEM status input whose condition can be tested by the CPU reading bit 7 (DCD) of the MODEM Status Register. Bit 7 is the complement of the signal. Bit 3 (DDCD) of the MODEM Status Register indicates whether the input has changed state since the previous reading of the MODEM Status Register. **NDCD** has no effect on the receiver. Note: Whenever the DCD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status interrupt is enabled.
NRI	In	RING signal from AFE	Ring Indicator. When LOW, this indicates that a telephone ring signal has been received by the MODEM or data set. The **NRI** signal is a MODEM status input whose condition can be tested by the CPU reading bit 6 (RI) of the MODEM Status Register. Bit 6 is the complement of the **NRI** signal. Bit 2 (TERI) of the MODEM Status Register indicates whether the **NRI** input signal has changed from a LOW to a HIGH state since the previous reading of the MODEM Status Register. Note: Whenever the RI bit of the MODEM Status Register changes from a HIGH to a LOW state, an interrupt is generated if the MODEM Status interrupt is enabled. The **NRI** input from the external PAD is not provided. To use this signal, you should set up the UART control register of the AFE interface. For further information, refer to ***13.9 Analog Front End, AFE (CODEC Interface)*** on page 13-56.

Table 13-2: Signal descriptions (Continued)

Name	Type	Source/ Destination	Description
NDTR	Out	External	Data Terminal Ready. When LOW, this informs the MODEM or data set that the UART is ready to establish communication link. The NDTR output signal can be set to an active LOW by programming bit 0 (DTR) of the MODEM Control Register to HIGH level. A Master Reset operation sets this signal to its inactive (HIGH) state. Loop mode operation holds this signal in its inactive state.
NRTS	Out	External	When LOW, this informs the MODEM or data set that the UART is ready to exchange data. The NRTS output signal can be set to an active LOW by programming bit 1 (RTS) of the MODEM Control Register. A Master Reset operation sets this signal to its inactive (HIGH) state. Loop mode operation holds this signal in its inactive state.
SOUT	Out	External	Serial Output. Composite serial data output to the communications link (peripheral, MODEM or data set). The **SOUT** signal is set to the Marking (logic 1) state upon a Master Reset operation.

Table 13-2: Signal descriptions (Continued)

DLAB	PA[5]	PA[4]	PA[3]	PA[2]	Register
0	0	0	0	0	Receiver Buffer (read). Transmitter Holding Register (write).
0	0	0	0	1	Interrupt enable
x	0	0	1	0	Interrupt identification (read)
x	0	0	1	0	FIFO control (write)
x	0	0	1	1	Line control
x	0	1	0	0	Modem control
x	0	1	0	1	Line status
x	0	1	1	0	Modem status
x	0	1	1	1	Scratch
1	0	0	0	0	Divisor latch (least significant byte)
1	0	0	0	1	Divisor latch (most significant byte)
0	1	0	0	0	UART enable register
0	1	1	0	0	UART test input register (write-only)
0	1	1	0	1	UART test output register (read-only)
0	1	1	1	1	UART TIC clock port (write-only)

Table 13-3: Register address

Register/Signal	Register Control	Register State
Interrupt Enable Register	Master Reset	0000 0000
Interrupt Identification Register		0000 0001
FIFO Control Register		0000 0000
Line Control Register		0000 0000
MODEM Control Register		0000 0000
Line Status Register		0110 0000
MODEM Status Register		xxxx 0000
SOUT	Master Reset	HIGH
INTUART (RCVR Errs)	Read LSR / RESET	LOW
INTUART (RCVR Data Ready)	Read RBR / RESET	LOW
INTUART(THRE)	ReadIIR / Write THR / RESET	LOW

Table 13-4: UART reset configuration

Register/Signal	Register Control	Register State
NRTS	Master Reset	HIGH
NDTR	Master Reset	HIGH

Table 13-4: UART reset configuration (Continued)

13.2.4 Internal block diagram

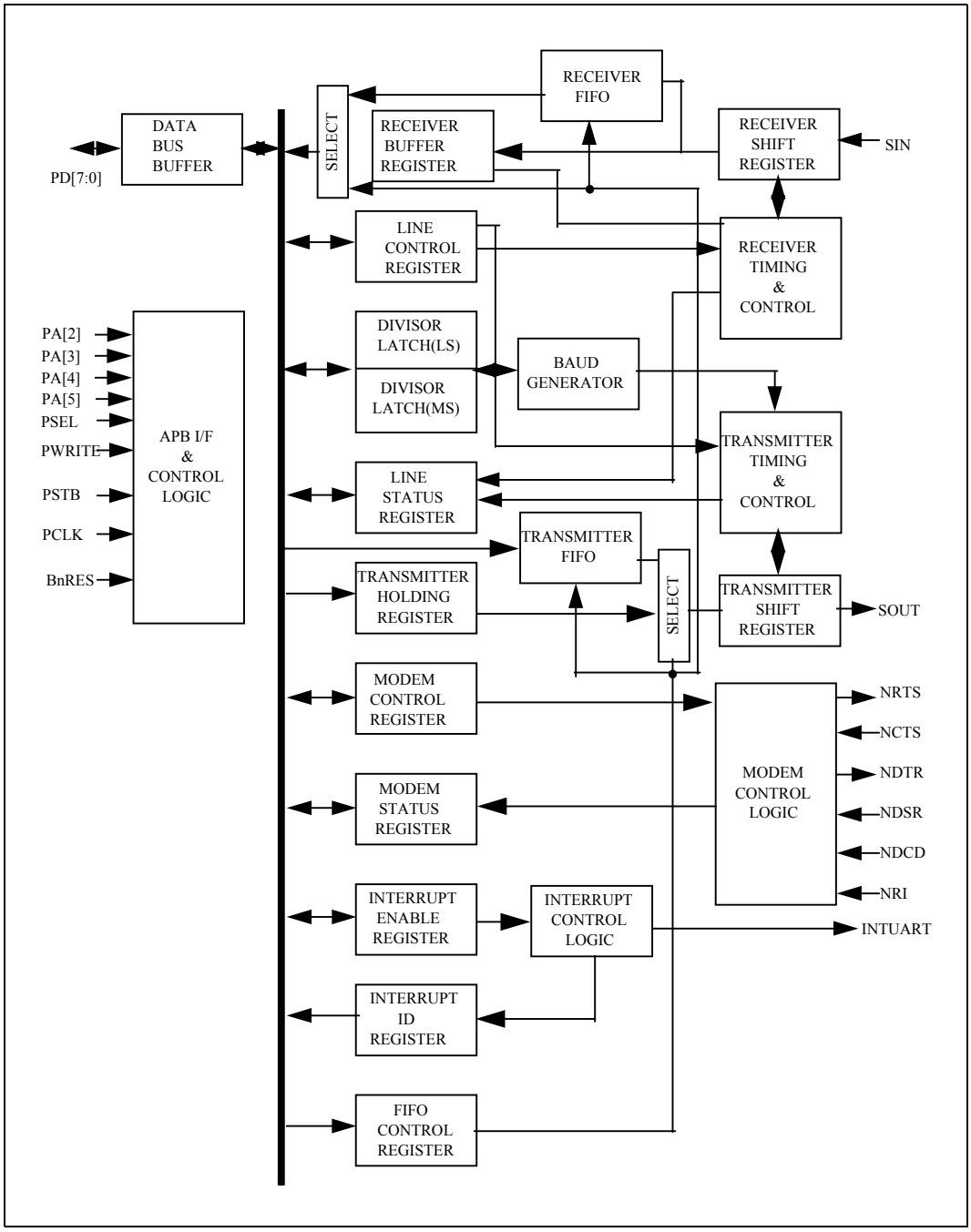


Figure 13-1: Internal block diagram

13.2.5 Registers description

There are two UARTs implemented in the design, the base addresses are U1Base and U2Base. ***UART Enable register is explained in page 13-20(Test Regsters of Uart)***.

In ***Table 13-5: UART register address map***, x can be either 1 or 2.

Address	Name	Description
UxBase + 0x00	Receiver_Buffer	# 8-bit R/O set DLAB=0
UxBase + 0x00	Transmitter_Holding	# 8-bit W/O set DLAB=0
UxBase + 0x04	Interrupt_Enable	# 8-bit R/W
UxBase + 0x08	Interrupt_Identification	# 8-bit R/O
UxBase + 0x08	FIFO_Control	# 8-bit W/O
UxBase + 0x0C	Line_Control	# 8-bit R/W
UxBase + 0x10	MODEM_Control	# 8-bit R/W
UxBase + 0x14	Line_Status	# 8-bit R/W
UxBase + 0x18	MODEM_Status	# 8-bit R/W
UxBase + 0x1C	Scratch	# 8-bit R/W
UxBase + 0x00	Divisor_Latch_LS	# 8-bit R/W set DLAB=1
UxBase + 0x04	Divisor_Latch_MS	# 8-bit R/W set DLAB=1

Table 13-5: UART register address map

Table 13-6: Summary of registers gives details of the UART registers.

	Register Address											
	0 DLAB=0	0 DLAB=0	1 DLAB=0	1	2	3	4	5	6	7	0 DLAB=1	1 DLAB=1
Bit No.	Receiver Buffer Register (R/O)	Transmitter Holding Register (W/O)	Interrupt Enable Register	Interrupt Ident Register (R/O)	FIFO Control Register (W/O)	Line Control Register	Modem Control Register	Line Status Register	Modem Status Register	Scratch Register	Divisor Latch (LS)	Divisor Latch (MS)
	RBR	THR	IER	IIR	FCR	LCR	MCR	LSR	MSR	SCR	DLL	DLM
0	Data Bit 0 (Note 1)	Data Bit 0	Enable received data available interrupt	0 if interrupt pending	FIFO enable	Word length select Bit 0	Data Terminal Ready (DTR)	Data Ready (DR)	Delta Clear to Send (DCTS)	Bit	0 Bit	0 Bit
1	Data Bit 1	Data Bit 1	Enable transmitter holding register empty interrupt	Interrupt ID Bit 0	RCVR FIFO reset	Word length select Bit 1	Request to Send (RTS)	Overrun Error (OE)	Delta Data Set Ready (DDSR)	Bit 1	Bit 1	Bit 9

Table 13-6: Summary of registers

Register Address													
2	Data Bit 2	Data Bit 2	Enable receiver line status interrupt	Interrupt ID Bit 1	XMIT FIFO reset	Number of stop bits		Parity Error (PE)	Trailing Edge Ring Indicator (TERI)	Bit	2	Bit	2 Bit
3	Data Bit 3	Data Bit 3	Enable modem status interrupt	Interrupt ID Bit 2 (Note 2)		Parity enable		Framing Error (FE)	Deltas Data Carrier Detect (DDCD)	Bit 3	Bit 3		Bit 11
4	Data Bit 4	Data Bit 4	0	0	Reserved	Even parity select	Loop	Break Interrupt (BI)	Clear to Send (CTS)	Bit	4	Bit	4 Bit
5	Data Bit 5	Data Bit 5	0	0	Reserved	Stick parity	0	Transmitter holding register empty (THRE)	Data Set Ready (DSR)	Bit 5	Bit 5		Bit 13
6	Data Bit 6	Data Bit 6	0	FIFO enabled (Note 2)	RCVR trigger (LSB)	Set break	0	Transmitter empty (TEMT)	Ring Indicator (RI)	Bit	6	Bit	6 Bit
7	Data Bit 7	Data Bit 7	0	FIFO enabled (Note 2)	RCVR trigger (MSB)	Divisor latch access bit	0	Error in RCVR FIFO (Note 2)	Data Carrier Detect (DCD)	Bit 7	Bit 7		Bit 15

Table 13-6: Summary of registers (Continued)

Notes *(1) Bit 0 is the least significant bit. It is the first bit serially transmitted or received.*

(2) These bits are always 0 in the 16450 mode.

The system programmer may access any of the UART registers summarized in ***Table 13-3: Register address*** on page 13-7 via the CPU. These registers control UART operation including transmission and reception of data. Each register bit in the table has its name and reset state shown.

Line Control Register

The system programmer specifies the format of the asynchronous data communications exchange and set the Divisor Latch Access bit via the Line Control Register (LCR). The programmer can also read the contents of the Line Control Register. The read capability simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. ***Table 13-6: Summary of registers*** on page 13-10 shows the contents of the LCR. Details on each bit follow.

Bit 0 and 1: These two bits specify the number of bits in each transmitted and received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Character Length
0	0	5 bits
0	1	6 bits
1	0	7 bits
1	1	8 bits

Table 13-7: Line control register encoding

Bit 2: This bit specifies the number of Stop bits transmitted and received in each serial character. If bit 2 is a logic 0, one Stop bit is generated in the transmitted

data. If bit 2 is a logic 1 when a 5-bit word length is selected via bits 0 and 1, one and a half Stop bits are generated. If bit 2 is a logic 1 when either a 6-, 7- or 8-bit word length is selected, two Stop bits are generated. The Receiver checks the first Stop-bit only, regardless of the number of Stop bits selected.

Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1s when the data word bits and the Parity bit are summed.)

Bit 4: This bit is the Even Parity Select bit. When bit 3 is a logic 1 and bit 4 is a logic 0, an odd number of logic 1s is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of logic 1s is transmitted or checked.

Bit 5: This bit is the Stick Parity bit. When bits 3, 4 and 5 are logic 1 the Parity bit is transmitted and checked as a logic 0. If bits 3 and 5 are 1 and bit 4 is a logic 0 then the Parity bit is transmitted and checked as a logic 1. If bit 5 is a logic 0 Stick Parity is disabled.

Bit 6: This bit is the Break Control bit. It causes a break condition to be transmitted to the receiving UART. When it is set to a logic 1, the serial output (**SOUT**)is forced to the Spacing (logic 0) state. The break is disabled by setting bit 6 to a logic 0. The Break Control bit acts only on **SOUT** and has no effect on the transmitter logic.

Note: This feature enables the CPU to alert a terminal in a computer communications system. If the following sequence is followed, no erroneous or extraneous characters will be transmitted because of the break.

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set HIGH (logic 1) to access the Divisor Latches of the Baud Generator during a Read or Write operation. It must be set LOW (logic 0) to access the Receiver Buffer, the Transmitter Holding Register or the Interrupt Enable Register.

Programmable Baud Generator

The UART contains a programmable Baud Generator that is capable of taking any clock input from DC to 8.0MHz and dividing it by any divisor from 2 to 2^{16}–1. 4MHz is the highest input clock frequency recommended when the divisor=1. The output frequency of the Baud Generator is 16 x the Baud [divisor # = (frequency input) / (baud rate x 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization to ensure proper operation of the Baud Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded.

Table 13-8: Baud rates on page 13-13 provide decimal divisors to use with a crystal frequency of 3.6864MHz. For baud rates of 38400 and below, the error obtained is minimal. The accuracy of the desired baud rate is dependent on the crystal frequency chosen. Using a divisor of zero is not recommended.

3.6864Mhz		
Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	4608	-
-	-	-
110	2094	0.026
-	-	-
-	-	-
300	768	-
-	-	-
1200	192	-
-	-	-
-	-	-
2400	96	-
-	-	-
4800	48	-
-	-	-
9600	24	-
19200	12	-
38400	6	-
57600	4	
115200	2	

Table 13-8: Baud rates

Line Status Register

This register provides status information to the CPU concerning the data transfer. ***Table 13-6: Summary of registers*** on page 13-10 shows the contents of the Line Status Register. Details on each bit follow.

Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register or the FIFO. Bit 0 is reset to a logic 0 by reading all of the data in the Receiver Buffer Register or the FIFO.

Bit 1: This bit is the Overrun Error (OE) indicator. Bit 1 indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is set to a logic 1 upon detection of an overrun condition and reset whenever the CPU reads the contents of the Line Status Register. If the FIFO mode data continues to fill the FIFO beyond the trigger level, an overrun error will occur only after the FIFO is full and the next character has been completely received in the shift register. OE is indicated to the CPU as soon as it happens. The character in the shift register is overwritten, but it is not transferred to the FIFO.

Bit 2: This bit is the Parity Error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even-parity-select bit. The PE bit is set to a logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU reads the contents of the Line Status Register. In the FIFO mode, this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO.

Bit 3: This bit is the Framing Error (FE) indicator. Bit 3 indicates that the received character did not have a valid stop bit. Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is detected as a logic 0 bit (Spacing level). The FE indicator is reset whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO. The UART will try to re-synchronize after a framing error. To do this it assumes that the framing error was due to the next start bit, so it samples this “start” bit twice and then takes in the “data”.

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits). The BI indicator is reset whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO. When break occurs, only one zero character is loaded into the FIFO. The next character transfer is enabled after **SIN** goes to the marking state and receives the next valid start bit.

Note: Bits 1—4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected and the interrupt is enabled.

Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. Bit 5 indicates that the UART is ready to accept a new character for transmission. In addition, this bit causes the UART to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set HIGH. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 concurrently with the loading of the Transmitter Holding Register by the CPU. In the FIFO mode this bit is set when the XMIT FIFO is empty; it is cleared when at least 1 byte is written to the XMIT FIFO.

Bit 6: This bit is the Transmitter Empty (TEMT) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Holding Register (THR) and the Transmitter Shift Register (TSR) are both empty. It is reset to a logic 0 whenever either the THR or TSR contains a data character. In the FIFO mode this bit is set to one whenever the transmitter FIFO and register are both empty.

Bit 7: In the 16450 mode this is a 0. In the FIFO mode LSR7 is set when there is at least one parity error, framing error or break indication in the FIFO. LSR7 is cleared when the CPU reads the LSR, if there are no subsequent errors in the FIFO.

Note *The Line Status Register is intended for read operations only.*

FIFO Control Register

This is a write-only register at the same location as the IIR (the IIR is a read-only register). This register is used to enable the FIFOs, clear the FIFOs and set the RCVR FIFO trigger level.

Bit 0: Writing a 1 to FCR0 enables both the XMIT and RCVR FIFOs. Resetting FCR0 will clear all bytes in both FIFOs. When changing from FIFO Mode to 16C450 Mode and vice versa, data is automatically cleared from the FIFOs. This bit must be a 1 when other FCR bits are written to or they will not be programmed.

Bit 1: Writing a 1 to FCR1 resets its counter logic to 0. The shift register is not cleared. The 1 that is written to this bit position is self-clearing.

Bit 2: Writing a 1 to FCR2 resets its counter logic to 0. The shift register is not cleared. The 1 that is written to this bit position is self-clearing.

Bit 3: FCR3 is not used.

Bit 4, 5: FCR4 to FCR5 are reserved for future use.

Bit 6, 7: FCR6 and FCR7 are used to set the trigger level for the RCVR FIFO interrupt.

FCR[7:6]	RCVR FIFO Trigger Level (Bytes)
00	01 (default)
01	04
10	08
11	14

Table 13-9: RCVR FIFO interrupt

Interrupt Identification Register

In order to provide minimum software overhead during data character transfers, the UART prioritizes interrupts into four levels and records these in the Interrupt Identification Register. The four levels of interrupt conditions are, in order of priority:

- Receiver Line Status
- Received Data Ready
- Transmitter Holding Register Empty
- MODEM Status.

When the CPU accesses the IIR, the UART freezes all interrupts and indicates the highest priority pending interrupt to the CPU. While this CPU access is occurring, the UART records new interrupts, but does not change its current indication until the access is complete. ***Table 13-6: Summary of registers*** on page 13-10 shows the contents of the IIR.

Details on each bit are outlined below.

Bit 0: This bit can be used in a prioritized interrupt environment to indicate whether an interrupt is pending. When bit 0 is a logic 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logic 1, no interrupt is pending.

Bit 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in ***Table 13-10: Interrupt control functions*** on page 13-16.

Bit 3: In the 16450 mode this bit is 0. In the FIFO mode, this bit is set along with bit 2 when a time-out interrupt is pending.

Bit 4 and 5: These two bits of the IIR are always logic 0.

Bit 6 and 7: These two bits are set when FCR0 = 1.

FIFO Mode Only	**Interrupt Identification Register**			**Interrupt Set and Reset Functions**			
Bit 3	Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	0	1	-	None	None	-
0	1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
0	1	0	0	Second	Receiver Data Available	Receiver Data Available or Trigger Level Reached	Reading the Receiver Buffer Register or the FIFO drops below the trigger level
1	1	0	0	Second	Character Time-out Indication	No Characters have been removed from or input to the RCVR FIFO during the last 4 Character times and there is at least 1 Character in it during this time	Reading the Receiver Buffer Register
0	0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or writing into the Transmitter Holding Register
0	0	0	0	Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

Table 13-10: Interrupt control functions

Interrupt Enable Register

This register enables the five types of UART interrupts. Each interrupt can individually activate the interrupt (INTUART) output signal. It is possible to totally disable the interrupt Enable Register (IER). Similarly, setting bits of the IER register to a logic 1, enables the selected interrupt(s). Disabling an interrupt prevents it from being indicated as active in the IIR and from

activating the INTUART output signal. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. ***Table 13-6: Summary of registers*** on page 13-10 shows the contents of the IER. Details on each bit follow.

Bit 0: This bit enables the Received Data Available Interrupt (and time-out interrupts in the FIFO mode) when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

Bit 4–7: These four bits are always logic 0.

MODEM Control Register

This register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register are indicated in ***Table 13-6: Summary of registers*** on page 13-10 and are described below.

Bit 0: This bit controls the Data Terminal Ready (**NDTR**) output. When bit is set to a logic 1, the **NDTR** output is forced to a logic 0. When bit 0 is reset to a logic 0, the **NDTR** output is forced to a logic 1.

Note: The **NDTR** output of the UART may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding MODEM or data set.

Bit 1: This bit controls the Request to Send (**NRTS**) output. Bit 1 affects the **NRTS** output in a manner identical to that described above for bit 0.

Bit 2: Not used

Bit 3: Not used

Bit 4: This bit provides a local loopback feature for diagnostic testing of the UART. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (**SOUT**) is set to the Marking (logic 1) state; the receiver Serial Input (**SIN**) is disconnected; the output of the Transmitter Shift Register is “looped back” into the Receiver Shift Register input; the four MODEM Control inputs (**NCTS**, **NDSR**, **NDCD** and **NRI**) are disconnected; and the two MODEM Control outputs (NDTR and NRTS) are internally connected to the four MODEM Control inputs, and the MODEM Control output pins are forced to their inactive state (HIGH). On the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit- and received-data paths of the UART.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. Their sources are external to the part. The MODEM Control interrupts are also operational, but the interrupts sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

Bit 5–7: These bits are permanently set to logic 0.

MODEM Status Register

This register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM change state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

The contents of the MODEM Status Register are indicated in ***Table 13-6: Summary of registers*** on page 13-10 and described below.

Bit 0: This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the **NCTS** input to the chip has changed state since the last time it was read by the CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the **NDSR** input to the chip has changed state since the last time it was read by the CPU.

Bit 2: This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the **NRI** input to the chip has changed from a LOW to a HIGH state.

Bit 3: This bit is the Delta Data Carrier Detect (DDCD) indicator. Bit 3 indicates that the **NDCD** input to the chip has changed state since the last time it was read by the CPU.

Note: Whenever bit 0, 1, 2 or 3 is set to logic 1, a MODEM Status Interrupt is generated.

Bit 4: This bit is the complement of the Clear to Send (**NCTS**) input. If bit 4 (loop) of the MCR is set to a 1, this bit is equivalent to RTS in the MCR.

Bit 5: This bit is the complement of the Data Set Ready (**NDSR**) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to DTR in the MCR.

Bit 6: This bit is the complement of the Ring Indicator (**NRI**) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT1 in the MCR.

Bit 7: This bit is the complement of the Data Carrier Detect (**NDCD**) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT2 in the MCR.

Scratch Register

This 8-bit Read/Write Register does not control the UART in any way. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.

FIFO Interrupt Mode Operation

When the RCVR FIFO and receiver interrupts are enabled (FCR 0 = 1, IER 0 = 1) RCVR interrupts occur as follows:

1. The received data available interrupt will be issued to the CPU when the FIFO has reached its programmed trigger level. It will be cleared as soon as the FIFO drops below its programmed trigger level.
2. The IIR receive data available indication also occurs when the FIFO trigger level is reached, and like the interrupt, it is cleared when the FIFO drops below the trigger level.
3. The receiver line status interrupt (IIR-06), as before, has higher priority than the received data available (IIR-04) interrupt.
4. The data ready bit (LSR 0) is set as soon as a character is transferred from the shift register to the RCVR FIFO. It is reset when the FIFO is empty.

When RCVR FIFO and receiver interrupts are enabled, RCVR FIFO time-out interrupts occurs as follows:

1 A FIFO time-out interrupt occurs if the following conditions exist:
   - at least one character is in the FIFO
   - the most recent serial character received was longer than four continuous character times ago (if two stop bits are programmed, the second one is included in this time delay)
   - the most recent CPU read of the FIFO was longer than four continuous character times ago

   This will cause a maximum character received to interrupt issued delay of 160 ms at 300 baud with a 12-bit character.

2 Character times are calculated by using the RCLK input, which is the internal signal of UART for a clock signal (this makes the delay proportional to the baud rate).

3 When a time-out interrupt has occurred, it is cleared and the timer is reset when the CPU reads one character from the RCVR FIFO.

4 When a time-out interrupt has not occurred the time-out timer is reset after a new character is received or after the CPU reads the RCVR FIFO.

When the XMIT FIFO and transmitter interrupts are enabled (FCR 0 = 1, IER 1 = 1), XMIT interrupts occurs as follows:

1 The transmitter holding register interrupt (02) occurs when the XMIT FIFO is empty. It is cleared as soon as the transmitter holding register is written to (1 to 16 characters may be written to the XMIT FIFO while servicing this interrupt) or the IIR is read.

2 The transmitter FIFO empty indications will be delayed 1 character time minus the last stop bit time whenever the following occurs: THRE = 1 and there has not been at least two bytes at the same time in the transmit FIFO since the last THRE = 1. The first transmitter interrupt affect changing FCR0 will be immediate if it is enabled.

Character time-out and RCVR FIFO trigger level interrupts have the same priority as the current received data available interrupt; XMIT FIFO empty has the same priority as the current transmitter holding register empty interrupt.

Test Registers of Uart

Four extra registers are provided inside the UART for test purposes. They are memory mapped as shown in ***Table 13-11: UART test registers***.

Registers	Read	Write	Width
Base Address + 0x20	UartEN	UartEN	1-bit
Base Address + 0x30		UartTIR	7-bit. Write-only.
Base Address + 0x34	UartTOR		3-bit. Read-only.
Base Address + 0x3C		UartTICCLK	Dummy address for generating TIC Clock. Write-only.

Table 13-11: UART test registers

Note *These registers should only be used for test purposes, and should not be accessed during normal operation.*

Detailed descriptions for each of the four registers now follow.

UART enable register

Bit	Description
0	0 = UART disable (power-down, default value), UART Clock stop 1 = UART enable

Table 13-12: UART enable register

UART test input register

This register is for programming on TIR[4:0] when the TIR[5] is set.

Bit	Description
6	This bit selects the source clock of the UART core block. 0 = the original UART clock(=3.6864MHz) 1 = TIC clock whenever the TIC clock port (UartTICCLK) is accessed, the TIC clock is generated.
5	This bit selects the source of the internal **NRI**, **SIN**, **NCTS**, **NDSR** and **NDCD** inputs. When it is 0, the external inputs from PADs are used. When it is set, the values programmed on TIR[4:0] are internally driven into the UART Core block to corresponding lines.
4	Programmable **NDCD** input when the TIR[5] is set. When the TIR[5] is 0 (default), the external input from PAD is used (normal operation).
3	Programmable **NDSR** input when the TIR[5] is set. When the TIR[5] is 0 (default), the external input from PAD is used (normal operation).
2	Programmable **SIN** input when the TIR[5] is set. When the TIR[5] is 0 (default), the external input from PAD is used (normal operation).
1	Programmable **SIN** input when the TIR[5] is set. When the TIR[5] is 0 (default), the external input from PAD is used (normal operation).
0	Programmable **NRI** input when the TIR[5] is set. When the TIR[5] is 0 (default), the external input from AFE I/F is used (normal operation).

Table 13-13: UART test input register

UART test output register

This register is for monitoring the external outputs from Uart.

Bit	Description
0	SOUT output line
1	NRTS output line
2	NDTR output line
3	INTUART output line

Table 13-14: UART test output register

Uart TIC clock port

This register is the Dummy Register used for generating the TIC clock in test mode. The TIC clock source is available only after Bit 6 of UART is set to 1.

13.3 SIR

GM30C7201 also contains a IrDA (Infra-red data association) SiR protocol encoder, which is attached to UART2, this encoder can be switched in to the **Tx** and **Rx** signals of UART2 so they can be used to drive an infra-red interface directly. For more details on the IrDA SiR protocol, see the appropriate document detailing this protocol standard. If the SiR protocol encoder is enabled, the UART **Tx** line is held in the passive state and transitions of the modem status or the **Rx** line will have no effect.

Section 12.4.1 gives information on enabling the IrDA encoder function. Section 3.2 describes programming UART2.

13.4 Keyboard Interface

The Keyboard Interface Controller is an AMBA slave module which connects to the Advanced Peripheral Bus (APB). For more information about AMBA, please refer to the *AMBA Specification* (ARM IHI 0001).

13.4.1 Overview

The keyboard interface controller unit has:

- four scanning modes
- 8x11 Matrix
- 11 byte key buffers
- TIC mode

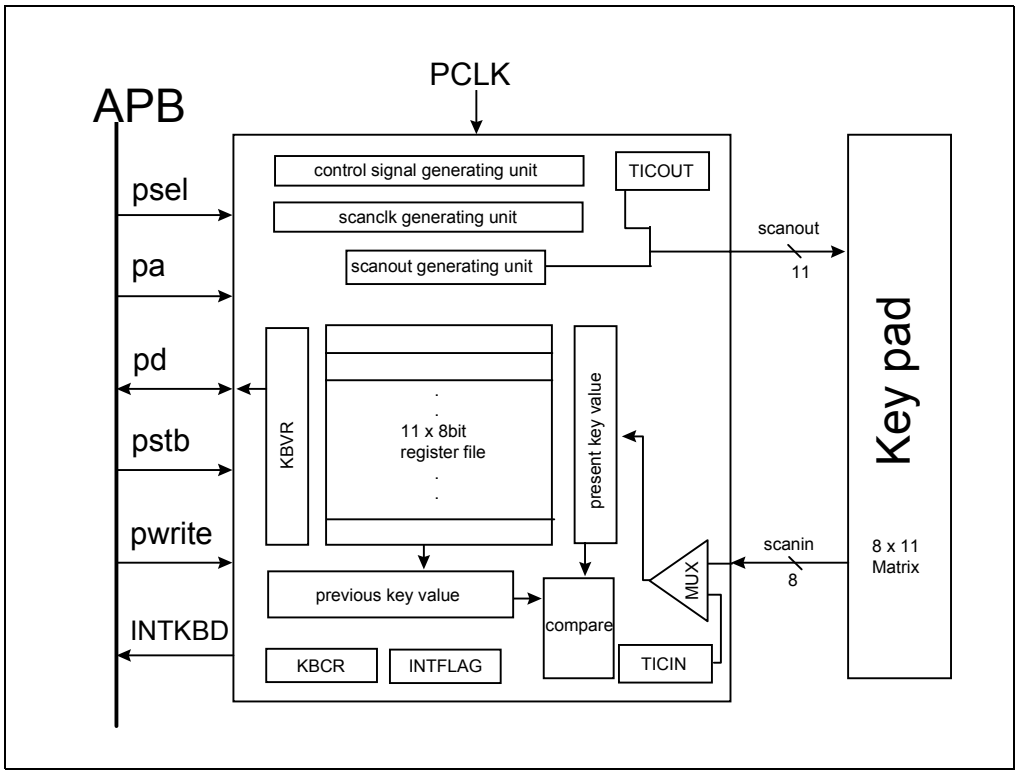


Figure 13-2: Keyboard interface block diagram

13.4.2 Hardware interface and signal description

The Keyboard Interface Controller is connected to the APB bus.

Name	Type	Source/ Destination	Description
PCLK	In	UART Clock	UART Clock. The input frequency from UART is 3.6864MHz. **PCLK** is used to generate the scan clock.
RESET	In	APB Bridge	Active LOW reset signal.
PA	In	APB Bridge	This is the peripheral address bus, which is used by an individual peripheral for decoding register accesses to that peripheral. The addresses become valid before **PSTB** goes HIGH, and remain valid after **PSTB** goes LOW.
PD[31:0]	InOut	APB Bridge/ Keyboard Interface	This is the bi-directional peripheral data bus. The data bus is driven by this block during read cycles (when **PWRITE** is LOW).
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of **PSTB** is coincident with the falling edge of **BCLK**.
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral; when LOW, it indicates a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates that this module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB). For more details, see *AMBA Peripheral Bus Controller* (ARM DDI 0044).
Scanout	Out	Keypad	This assigns the x-axis' scan line. The value is changed periodically so as to cover every key matrix. During one keyboard scan, **scanout** can have 11 different values. Active LOW signal.
Scanin	In	Keypad	This indicates which key is pressed in the assigned scan line. Active LOW signal.
INTKBD	Out	Interrupt Controller	Interrupt signal to the Interrupt Controller module. This signal indicates if 11th column key scan is finished. Active HIGH signal.

Table 13-15: Signal descriptions

13.4.3 Keyboard interface controller unit introduction

The interface controller is designed to communicate with the external keyboard. The keyboard interface uses the pins `scanin`, `scanout` and all of the APB signal. It is possible to select one of four scan clock modes.

Scan clock:

PCLK/2 1.84MHz, test mode

PCLK/128 28 kHz
PCLK/256 14 kHz
PCLK/512 7 kHz

`scanclock` is generated using **PCLK** (3.6864MHz).

Programmable scan rate:

6.5K times/sec test mode scan rate
101 times/sec
50 times/sec
25 times/sec

11Byte key buffer: 11 column key scan values are stored (8 x 11 key matrix).

13.4.4 Keyboard interface controller unit operation

To start key input scanning, set the SCANEN bit and POWERDOWN bit of **KBCR** (Keyboard Configuration Register) and the **CLKSEL** bit of the KBCR. The key scan control signal is generated. Periodically, column scan code is saved in the 11-byte key buffer. After the 11th column key data is stored, **INTKBD** is generated to make the CPU read 11 scan values.

The keyboard interface block leaves reset in power down mode. To activate the block bits [7] and [2] of the KBCR register should be programmed HIGH, then the keyboard will be automatically scanned according to the programmed rate, and scan data will be stored in the KBVR registers. When all the keyboard has been scanned, an interrupt is generated, and, by interrogating the KBVR registers, software can determine which keys have been pressed. It is software's responsibility to debounce the key pressed information. Keyboard key press interrupts are generated in all PMU states except deep sleep.

13.4.5 Keyboard interface controller unit register map

The base address of the keyboard interface controller unit is 0x80022000, and the offset of any particular register from the base address is fixed.

Bit	Address	Access (R/W)	Read location/Write location
8 bit	KIC Base + 0x00	R/W	KBCR/KBCR
11 bit	KIC Base + 0x04	R	TICOUT
8 bit	KIC Base + 0x08	R/W	TICIN/TICIN
32 bit	KIC Base + 0x0c	R	KBVR0
32 bit	KIC Base + 0x10	R	KBVR1
32 bit	KIC Base + 0x14	R	KBVR2
1 bit	KIC Base + 0x18	R	KBSR
0 bit	KIC Base + 0x1c		TCLK This is a virtual register, used to generate TIC CLK in test mode.

Table 13-16: Keyboard interface controller unit register memory map

13.4.6 Keyboard interface controller unit register descriptions

The following registers are provided for the keyboard interface controller unit.

Keyboard configuration register (KBCR)

This is an 8-bit writable and readable register that selects scan clock mode and scan enable, and test mode, etc. The Scan bit and Power down bit are usually set or reset simultaneously. During a test, the Scan bit and Power down bit could have differing values.

Bit	Name	Initial value	Description
7	Scan En	0	Start and stop scanning 0 = stop 1 = start
6	TIC EN	0	TIC mode 0 = disable (normal mode) 1 = enable (test mode)
5	TIC Clock EN	0	Select input clk 0 = PCLK (normal mode) 1 = TCLK (test mode)
4	Scan Clock select	0	Select Scan Clk for fast test 0 = normal scan clk 1 = test mode scan clk = 1.84MHz
3	Reserved	0	Reserved
2	Power down	0	Power down 0 = power down mode, where clock is not operating 1 = normal mode, where clock is operating
1:0	Clksel	0	Scanning rate control 00 = 1.84MHz 01 = 28kHz 10 = 14kHz 11 = 7kHz

Table 13-17: KBCR bit description

Keyboard value register (KBVR0–KBVR2)

This is a 32-bit readable register that has a value of `scanin`. For example, if the value of KBVR0[32:24] is 00001100, the 5th and 6th keys are pressed and the others are released. In TIC mode, the TICIN value is inverted, compared and stored to the keyboard value register.

Bit	Name	Initial value	Description
31:24	KBVR0	0	10st column `scanin` value
23:16	KBVR0	0	9nd column `scanin` value
15:8	KBVR0	0	8rd column `scanin` value
7:0	KBVR0	0	7th column `scanin` value
31:24	KBVR1	0	6th column `scanin` value
23:16	KBVR1	0	5th column `scanin` value
15:8	KBVR1	0	4th column `scanin` value
7:0	KBVR1	0	3th column `scanin` value
31:24	KBVR2	0	2th column `scanin` value
23:16	KBVR2	0	1th column `scanin` value
15:8	KBVR2	0	0zth column `scanin` value
7:0	KBVR2	0	Reserved

Table 13-18: KBVR0–KBVR2 bit description

TIC out register (TICOUT)

This is a test register that allows the KBD output signal to the keypad to be read back in test mode. No more than two bits can be reset, causing the interface to scan only one line during each scan period, except when the keyboard is disabled (ScanEn bit of KBCR=0).

Bit	Initial value	Description
10	0	0 = 1st line will be scanned 1 = no scan
9	0	0 = 2nd line will be scanned 1 = no scan
8	0	0 = 3rd line will be scanned 1 = no scan
7	0	0 = 4th line will be scanned 1 = no scan
6	0	0 = 5th line will be scanned 1 = no scan

***Table 13-19:* TIC out register**

Bit	Initial value	Description
5	0	0 = 6th line will be scanned 1 = no scan
4	0	0 = 7th line will be scanned 1 = no scan
3	0	0 = 8th line will be scanned 1 = no scan
2	0	0 = 9th line will be scanned 1 = no scan
1	0	0 = 10th line will be scanned 1 = no scan
0	0	0 = 11th line will be scanned 1 = no scan

Table 13-19: **TIC out register** ***(Continued)***

Value (binary number)	Description
011_1111_1111	1st line `scanout` value
101_1111_1111	2nd line `scanout` value
110_1111_1111	3rd line `scanout` value
111_0111_1111	4th line `scanout` value
111_1011_1111	5th line `scanout` value
111_1101_1111	6th line `scanout` value
111_1110_1111	7th line `scanout` value
111_1111_0111	8th line `scanout` value
111_1111_1011	9th line `scanout` value
111_1111_1101	10th line `scanout` value
111_1111_1110	11th line `scanout` value
000_0000_0000	All line `scanout` value (when `scanen = 0`)

Table 13-20: TICOUT bit description

TIC in register (TICIN)

8-bit readable and writable register that is used instead of the `scanin` value (value from KeyPad) in case of TIC mode. For example, 1110_1100 means 4th key, 7th key and 8th key in the current scan column are pressed. Refer to ***Table 13-21: TIC in register*** on page 13-29.

Bit	Initial value	Description
7	1	Indicates whether 1st key in the selected scan column is pressed: 0 = pressed 1 = not pressed
6	1	Indicates whether 2nd key in the selected scan column is pressed: 0 = pressed 1 = not pressed
5	1	Indicates whether 3rd key in the selected scan column is pressed: 0 = pressed 1 = not pressed
4	1	Indicates whether 4th key in the selected scan column is pressed: 0 = pressed 1 = not pressed
3	1	Indicates whether 5th key in the selected scan column is pressed: 0 = pressed 1 = not pressed
2	1	Indicates whether 6th key in the selected scan column is pressed: 0 = pressed 1 = not pressed
1	1	Indicates whether 7th key in the selected scan column is pressed: 0 = pressed 1 = not pressed
0	1	Indicates whether 8th key in the selected scan column is pressed: 0 = pressed 1 = not pressed

Table 13-21: **TIC in register**

Value (binary number)	Initial value	Description
1111_1111		
1111_1110		
1111_1101		
:	0000_0000	present `scanin` value
:		
:		
0000_0001		

Value (binary number)	Initial value	Description
0000_0000		

Table 13-22: TICIN bit description

Keyboard status register (KBSR)

This is a 2-bit readable register that indicates whether a keyboard interrupt has occurred. The interrupt and the KBSR bit are cleared after the CPU reads KBSR. The KBSR bit is set when the key buffer is full, or when the key is pressed in powerdown mode (keyboard disabled).

Bit	Initial value	Description
1 (Wake up)	0	Wake up state: 0 = no key pressed in powerdown mode 1 = key pressed in powerdown mode
0 (Interrupt state)	0	Key bufferstate: 0 = key buffer is not full 1 = key buffer is full

Table 13-23: KBSR bit description

TCLK register

This register does not exist: it is used to generate **TIC clk** in test mode. When the APB address (PA) is **TCLK** register address and **PSEL** and **PSTB** is HIGH phase in test mode, **TCLK** is HIGH, otherwise **TCLK** is LOW stage.

13.5 GPIO

This document describes the Programmable Input /Output module (PIO). This is an AMBA slave module which connects to the Advanced Peripheral Bus (APB). For more information about AMBA, please refer to the *AMBA Specification* (ARM IHI 0001).

13.5.1 Module overview

The PIO is an APB peripheral which provides 32 bits of programmable input/output divided into four 8-bit ports: port A, port B, port C and port D. Each pin is configurable as either input or output. At system reset, all ports default to input.

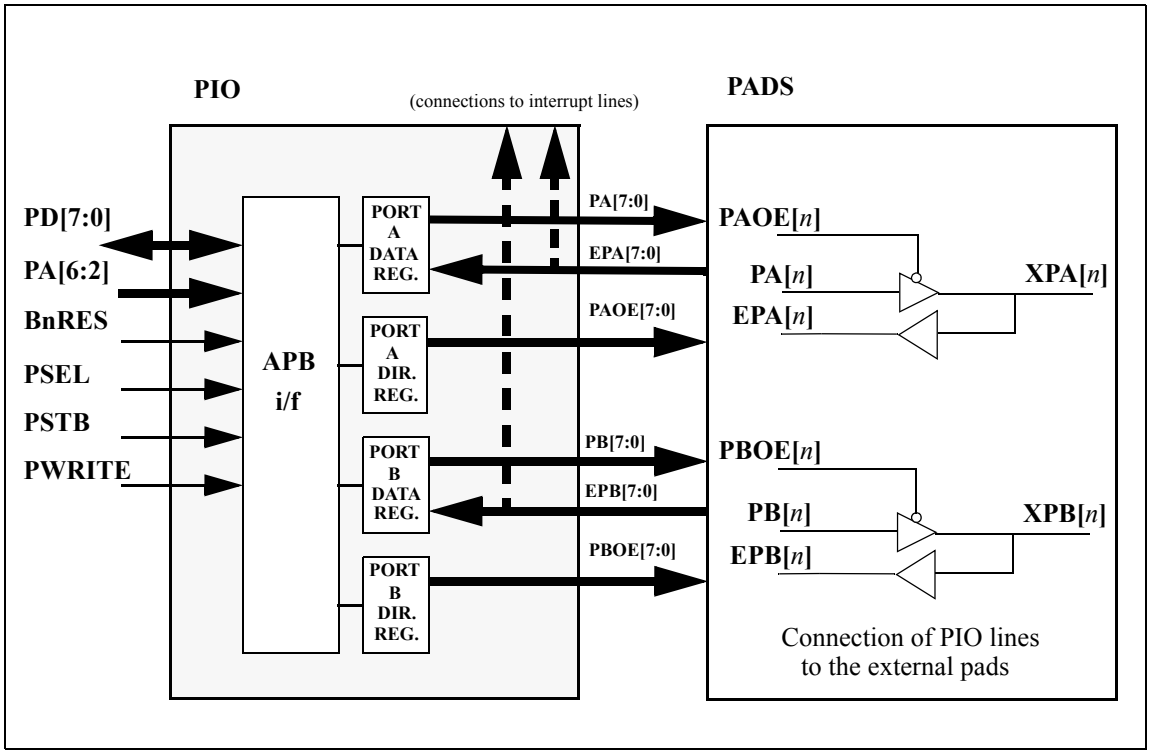


Figure 13-3: PIO block diagram and PADS connections

Note ***Figure 13-3: PIO block diagram and PADS connections*** *shows only the configuration for 16 bits.*

Each port has a data register and a data direction register, both 8 bits wide. The data direction register defines whether each individual pin is an input or an output. The data register is used to read the value of the PIO pins—both input and output—as well as to set the values of pins that are configured as outputs. When the PIO pin is defined as input, this input can be an interrupt source with register setting.

13.5.2 Signal description

The PIO module is connected to the APB bus. ***Table 13-24: Signal descriptions*** on page 13-32 describes the APB signals used and produced. ***Table 13-25: Specific block signal descriptions*** on page 13-32 shows the non-AMBA signals from the block.

Name	Type	Source/ Destination	Description
BnRES	In	Reset Controller	This signal indicates a power-on reset status of the bus (active LOW).
PA[6:2]	In	APB Bridge	This is part of the peripheral address bus, which is used by the peripheral for decoding its own register accesses. The addresses become valid before **PSTB** goes HIGH and remain valid after **PSTB** goes LOW.
PD[7:0]	InOut	APB Peripherals, BD bus	This is part of the bidirectional peripheral data bus. The data bus is driven by this block during read cycles (when **PWRITE** is LOW).
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of **PSTB** is coincident with the falling edge of **BCLK** (ASB system clock).
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral and when LOW, a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates the PIO module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB). For more details, see *AMBA Peripheral Bus Controlle*r (ARM DDI 0044).

Table 13-24: Signal descriptions

Name	Type	Source/ Destination	Description
PA[7:0]	Out	PADS	Port A output driver. Values written on PADR register are put onto these lines and driven out to the port A pins if the corresponding data direction bits are set HIGH (PADDR register).
EPA[7:0]	In	PADS	Port A input driver. It reflects the external state of the port. This information is obtained when reading the PADR register.
PAOE[7:0]	Out	PADS	Port A output enable (active LOW). Values written on PADDR register are put onto these lines.
PB[7:0]	Out	PADS	Port B output driver. Values written on PBDR register are put onto these lines and driven out to the port B pins if the corresponding data direction bits are set LOW (PBDDR register).
EPB[7:0]	In	PADS	Port B input driver. It reflects the external state of the port. This information is obtained when reading the PBDR register.

Table 13-25: Specific block signal descriptions

Name	Type	Source/ Destination	Description
PBOE[7:0]	Out	PADS	Port B output enable (active LOW). Values written on PBDDR register are put onto these lines.
PC[7:0]	Out	PADS	Port C output driver. Values written on PCDR register are put onto these lines and driven out to the port C pins if the corresponding data direction bits are set HIGH (PCDDR register).
EPC[7:0]	In	PADS	Port C input driver. It reflects the external state of the port. This information is obtained when reading the PCDR register.
PCOE[7:0]	Out	PADS	Port C output enable (active LOW). Values written on PCDDR register are put onto these lines.
PD[7:0]	Out	PADS	Port D output driver. Values written on PDDR register are put onto these lines and driven out to the port D pins if the corresponding data direction bits are set LOW (PDDDR register).
EPD[7:0]	In	PADS	Port D input driver. It reflects the external state of the port. This information is obtained when reading the PDDR register.
PDOE[7:0]	Out	PADS	Port D output enable (active LOW). Values written on PDDDR register are put onto these lines.

Table 13-25: Specific block signal descriptions (Continued)

13.5.3 Functional description

All pins are defined as input during reset (**BnRES** LOW).

For each port there is a Data Register and a Data Direction Register. On reads, the Data Register contains the current status of correspondent port pins, whether they are configured as input or output. Writing to a Data Register only affects the pins that are configured as outputs.

All PIO input pins can be used as interrupt source with enabled interrupt mask register bit. These interrupt sources can be selected as active HIGH/LOW, EDGE/LEVEL trigger mode.

Bits[5:0] of port B and bit[3:0] of port D are multiplexed with other functions and regarded as multi-function pins. In order to use these multi-function pins as PIO pins, the Multi-function Pin Selection register (PMPS) bit should be set.

13.5.4 Programmer's model

PIO registers

The following user registers are provided:

P[A,B,C,D]DR — Data Register. Values written to this 8-bit read/write register will be output on port [A,B,C,D] pins if the corresponding data direction bits are set Low (port output). Values read from this register reflect the external state of port [A,B,C,D] not necessarily the value written to it. All bits are cleared by a system reset.

P[A,B,C,D]DDR — Port [A,B,C,D] Data Direction Register. Bits set in this 8-bit read/write register will select the corresponding pin in port [A,B,C,D] to become an input, clearing a bit sets the pin to output. All bits are set by a system reset.

All PIO signals can be used as interrupt sources according to the settings. Each port has the following registers and interrupt signals to interrupt controller. Interrupt controller receives active HIGH, level mode interrupt sources only. But PIO block can receive not only active HIGH or active LOW, but also level or edge mode signals. Then interprets and sends interrupt

request to the interrupt controller. All bits can be controlled separately.

P[A,B,C,D]IM — Interrupt Mask Register. Bits set in this 8-bit read/write register will select the corresponding pin to become an interrupt source. All bits are cleared by a system reset.

0 = disable interrupt (default)

1 = enable interrupt

P[A,B,C,D]IS — Interrupt Status Register. Values in this 8-bit read-only register represents that the interrupt requests are pending on corresponding pins. All bits are cleared by a system reset.

0 = no interrupt request

1 = interrupt pending

(masked interrupt is always 0)

P[A,B,C,D]IE — Edge Mode Register. Bits set in this 8-bit read/write register will select the corresponding pin to become an edge mode interrupt source. All bits are cleared by a system reset.

0 = level mode (default)

1 = edge mode

P[A,B,C,D]IC — Clear Register. Bits set in this 8-bit write-only register will clear the stored interrupt request of corresponding bit in edge mode. All bits are automatically cleared after written.

0 = no action (default)

1 = clear interrupt source (self reset)

P[A,B,C,D]IP — Polarity Register. Bits set in this 8-bit read/write register will select the corresponding pin to become an active LOW mode interrupt source. All bits are cleared by a system reset. After accessing this register, the Edge Mode register should be cleared with the Clear register.

0 = active HIGH mode

1 = active LOW mode

TICR — TIC Mode Selection Register. This 1-bit write register will select TIC mode for PORTC[7:0] and PORTD[6:4]. When set, the read of PDR returns the value of the PDR register instead of external pin signals.This is for production test purposes only. It should always be clear in normal operation.

PMPSB — Multi-function Pin Selection Register for PORTB. Bits set in this 6-bit read/write register will select the corresponding pin to become a PIO pin.

PMPSB	Value	Description
Bit 0	0 1	UVPO PORTB bit[0]
Bit 1	0 1	UVMO PORTB bit[1]
Bit 2	0 1	nUSBOE PORTB bit[2]

Table 13-26: PIO multi-function pin selection for PORT B

PMPSB	Value	Description
Bit 3	0 1	URTVIN PORTB bit[3
Bit 4	0 1	UVP PORTB bit[4]
Bit 5	0 1	UVM PORTB bit[5]

Table 13-26: PIO multi-function pin selection for PORT B

PMPSD — Multi-function Pin Selection Register for PORTD. Bits set in this 4-bit read/write register will select the corresponding pin to become a PIO pin.

PMPSD	Value	Description
Bit 0	0 1	BCLK PORTD bit[0]
Bit 1	0 1	nRCS3 PORTD bit[1]
Bit 2	0 1	nRCS4 PORTD bit[2]
Bit 3	0 1	nRCS5 PORTD bit[3

Table 13-27: PIO multi-function pin selection for PORT D

Register memory map

The base address of the PIO is not fixed and may be different for any particular system implementation. However, the offset of any particular register from the base address is determined.

Address	Access (R/W)	Initial Value	Name	Register Description
PIO Base + 0x00	R/W	0x00	PADR	Port A Data Register
PIO Base + 0x04	R/W	0xFF	PADDR	Port A Direction Register
PIO Base + 0x08	R/W	0x00	PAIM	Port A Interrupt Mask Register
PIO Base + 0x0C	R	0x00	PAIS	Port A Interrupt Status Register
PIO Base + 0x10	R/W	0x00	PAIE	Port A Interrupt Edge Mode Register
PIO Base + 0x14	W	0x00	PAIC	Port A Interrupt Clear Register
PIO Base + 0x18	R/W	0x00	PAIP	Port A Interrupt Polarity Register
PIO Base + 0x20	R/W	0x00	PBDR	Port B Data Register
PIO Base + 0x24	R/W	0xFF	PBDDR	Port B Direction Register
PIO Base + 0x28	R/W	0x00	PBIM	Port B Interrupt Mask Register
PIO Base + 0x2C	R	0x00	PBIS	Port B Interrupt Status Register
PIO Base + 0x30	R/W	0x00	PBIE	Port B Interrupt Edge Mode Register
PIO Base + 0x34	W	0x00	PBIC	Port B Interrupt Clear Register
PIO Base + 0x38	R/W	0x00	PBIP	Port B Interrupt Polarity Register
PIO Base + 0x40	R/W	0x00	PCDR	Port C Data Register
PIO Base + 0x44	R/W	0xFF	PCDDR	Port C Direction Register

Table 13-28: PIO register memory map

Address	Access (R/W)	Initial Value	Name	Register Description
PIO Base + 0x48	R/W	0x00	PCIM	Port C Interrupt Mask Register
PIO Base + 0x4C	R	0x00	PCIS	Port C Interrupt Status Register
PIO Base + 0x50	R/W	0x00	PCIE	Port C Interrupt Edge Mode Register
PIO Base + 0x54	W	0x00	PCIC	Port C Interrupt Clear Register
PIO Base + 0x58	R/W	0x00	PCIP	Port C Interrupt Polarity Register
PIO Base + 0x60	R/W	0x00	PDDR	Port D Data Register
PIO Base + 0x64	R/W	0xFF	PDDDR	Port D Direction Register
PIO Base + 0x68	R/W	0x00	PDIM	Port D Interrupt Mask Register
PIO Base + 0x6C	R	0x00	PDIS	Port D Interrupt Status Register
PIO Base + 0x70	R/W	0x00	PDIE	Port D Interrupt Edge Mode Register
PIO Base + 0x74	W	0x00	PDIC	Port D Interrupt Clear Register
PIO Base + 0x78	R/W	0x00	PDIP	Port D Interrupt Polarity Register
PIO Base + 0x3C	R/W	0x00	PMPSB	Multi-function Pin select for PORT B
PIO Base + 0x5C	W	0x00	TICR	TIC Mode Selection Register
PIO Base + 0x7C	R/W	0x00	PMPSD	Multi-function Pin select for PORT D

Table 13-28: PIO register memory map (Continued)

13.6 Interrupt Controller

The interrupt controller has the following features:

- a status register
- selection of the output path (IRQ or FIQ for each input)
- enabling the interrupt

The interrupt controller provides a simple software interface to the interrupt system.

In an ARM system, two levels of interrupt are available:

- FIQ (Fast Interrupt Request) for fast, low-latency interrupt handling
- IRQ (Interrupt Request) for more general interrupts

Ideally, in an ARM system, only a single FIQ source would be in use at any particular time. This provides a true low-latency interrupt, because a single source ensures that the interrupt service routine may be executed directly without the need to determine the source of the interrupt. It also reduces the interrupt latency because the extra banked registers, which are available for FIQ interrupts, may be used to maximum efficiency by preventing the need for a context save.

The interrupt controller provides a bit position for each different interrupt source. Bit positions are defined for a software programmed interrupt.

Any interrupt source can be programmed as a source to FIQ or IRQ interrupt.

All interrupt source inputs must be active HIGH and level sensitive.

No hardware priority scheme nor any form of interrupt vectoring is provided, because these functions can be provided in software. Any interrupt source may be masked.

13.6.1 In/Out signals

Signal	Type	Description
PD[23:0]	InOut	Data Bus
PA[4:2]	In	Address Bus
PSEL	In	Chip Select
PSTB	In	Strobe Signal
PWRITE	In	nRead/Write
BnRES	In	Reset Signal
INT[23:0]	In	Interrupt Request Inputs
nIRQ	Out	Interrupt Request to CPU
nFIQ	Out	Fast Interrupt Request to CPU

Table 13-29: Interrupt Controller in/out signals

13.6.2 Interrupt control

The interrupt controller provides interrupt request status, interrupt enable and interrupt direction selection registers. The enable register is used to determine whether or not an active interrupt source should generate an interrupt request to the processor. All bits are cleared by system reset.

The interrupt request status indicates whether or not the interrupt source is causing a processor interrupt.

The direction register is used to determine which interrupt request is generated to the CPU. If the bit is set, FIQ request is activated. All bits are cleared by system reset.

TIC registers are used only for the production test. TIC input select register is used to drive interrupt request sources by CPU. When this register is set, TIC register bits is regarded as interrupt sources. This bit is cleared by system reset and should be cleared in normal operation.

TIC register is used as interrupt source when TIC input selection register is set.

Bit 23 is used as a software interrupt source. When Enable Register bit [23] is HIGH, an interrupt request occurs. To disable the software interrupt, Enable Register bit [23] should be LOW.

13.6.3 Register map

Register	Address	R/W	Initial Value
Enable Register	IntBase + 0x08	R/W	0x000000
Direction Register	IntBase + 0x0C	R/W	0x000000
Status Register	IntBase + 0x10	R	0x000000
TIC Input Selection Register	IntBase + 0x18	W	0x0000
TIC Register as interrupt sources	IntBase + 0x1C	W	0x000000

Table 13-30: Interrupt controller register map

Enable Register: enable each interrupt source
0 : disable interrupt (default)
1 : enable interrupt

Direction Register: interrupt sources will trigger nIRQ or nFIQ
0 : activate nIRQ output (default)
1 : activate nFIQ output

Status Register: current interrupt request status (read-only)
0 : no interrupt request
1 : interrupt pending (masked interrupt is always "0")

TIC Input Select Register: Chip test purpose only.
Should be "0" during normal operation.
0 : normal function
1 : select TIC Register as interrupt sources

TIC Register: Chip test purpose only.
Used as interrupt sources in TIC mode.
Should be "0" during normal operation.

13.6.4 Interrupt configuration

Bit	Interrupt Source
0	PMU
1	DMA
2	LCD
3	VGA
4	PCMCIA 1
5	PCMCIA 2
6	AFE (Modem Codec Interface)
7	AIC (Analog Interface)
8	KBD (Keyboard Interface)
9	TIMER
10	RTC
11	SOUND
12	USB
13	IrDA (MIR/FIR)
14	UART 1
15	UART 2 (SIR)
16	SPI
17	GPIO Port A
18	GPIO Port B
19	GPIO Port C
20	GPIO Port D
21	ARM core (COMMRX: debug only)
22	ARM core (COMMTX: debug only)
23	Always HIGH (software interrupt)

Table 13-31: Interrupt configuration

13.7 Timers

The timer has the following features:

- 32-bit up ripple counter
- Auto repeat mode
- Count enable/disable
- Interrupt enable/disable
- 3-timer channel

13.7.1 In/Out signals

Signal	Type	Description
PCLK	In	Clock Source
PD[31:0]	In/Out	Data Bus
PA[6:2]	In	Address Bus
PSEL	In	Chip Select
PSTB	In	Strobe
PWRITE	In	nRead/Write
INTTIMER	Out	“1” when counter register value is equal to the base register value.
BnRES	In	Reset Signal

Table 13-32: Timers in/out signals

13.7.2 Register map

Address	R/W	Initial Value	Register
Timer Base + 0x00	R/W	0xFFFFFFFF	Timer0 Base Register
Timer Base + 0x08	R	0x00000000	Timer0 Counter Register
Timer Base + 0x10	R/W	0x00	Timer0 Control Register
Timer Base + 0x14	W	0x00	Timer0 Test Register
Timer Base + 0x20	R/W	0xFFFFFFFF	Timer1 Base Register
Timer Base + 0x28	R	0x00000000	Timer1 Counter Register
Timer Base + 0x30	R/W	0x00	Timer1 Control Register
Timer Base + 0x34	W	0x00	Timer1 Test Register
Timer Base + 0x40	R/W	0xFFFFFFFF	Timer2 Base Register
Timer Base + 0x48	R	0x00000000	Timer2 Counter Register
Timer Base + 0x50	R/W	0x00	Timer2 Control Register
Timer Base + 0x54	W	0x00	Timer2 Test Register
Timer Base + 0x60	R/W	0x0	Top Control Register
Timer Base + 0x64	R	0x0	Status Register
Timer Base + 0x68	W	0x0	TICCLK selection in test mode
Timer Base + 0x7C	W		TICCLK Generation Note: whenever this port is written to, it will generate TICCLK pulse.

Table 13-33: Timer port addresses

Base Register: 32-bit target count value (interval)

Counter Register: 32-bit up counter

Channel Control Register (Timer0, Timer1)

bit 0 1 = start count

0 = stop count

This bit clears automatically when the counter reaches the target value if it is in non-repeat mode.

bit 1 1 = count repeat mode

bit 2 1 = reset counter register

bit[7:3] Reserved

Channel Control Register (Timer2)		
	bit 0	1 = start count
		0 = stop count
	bit 1	1 = count repeat mode
	bit 2	1 = reset Counter Register
	bit[7:3]	Reserved
Channel Test Register	This register is for chip test purposes only. All bits should be 0 during normal operation.	
	bit 0	“1” clock of 5th bit (bit 4) is from clock source
	bit 1	“1” clock of 9th bit (bit 8) is from clock source
	bit 2	“1” clock of 13th bit (bit 12) is from clock source
	bit 3	“1” clock of 17th bit (bit 16) is from clock source
	bit 4	“1” clock of 21st bit (bit 20) is from clock source
	bit 5	“1” clock of 25th bit (bit 24) is from clock source
	bit 6	“1” clock of 29th bit (bit 28) is from clock source
	bit 7	Reserved
Top Control Register		
	bit 0	Timer 0 interrupt enable/disable
	bit 1	Timer 1 interrupt enable/disable
	bit 2	Timer 2 interrupt enable/disable
		1 = interrupt enable
		0 = interrupt disable. If all three bits are 0, then INTTIMER is always 0.
	bit 3	Timer enable
		1 = enable (normal mode)
		0 = disable (lower power mode - default)
	bit 4	enable 64-bit counter
		1 = enable. When this bit is set, the period of Timer 1 is used as the clock source of Timer 2.
		In this mode, the period of Timer 2 is the product of the base register value of Timer 1 and the base register value of Timer 2.
		0 = disable
	bit[7:5]	Reserved

Status Register	Auto reset after read	
	bit 0	Timer 0 interrupt
	bit 1	Timer 1 interrupt
	bit 2	Timer 2 interrupt
	bit[7:3]	Reserved
Top Test Register	Production test purposes only	
	bit 0	TICCLK
		1 = enable
		0 = disable (normal mode)

Note 1 The interrupt interval in repeat mode is (Base Register value + 1) clock periods.

For example, if the Base Register is set to 0x3333, then the timer generates an interrupt request every 0x3333 + 1 clock cycles.

In 64bit mode, the repeat mode period is

(Timer1 Base Register value + 1) x (Timer2 Base Register value + 1).

13.8 Synchronous Serial Interface

The SPI is a high-speed synchronous serial port for communicating to external devices. The SPI in this document is for MMC.

As with any other SPI device, the SPI-MMC circuit consists of the following four signals:

- CS Host to card chip select signal
- SPICLK Host to card clock signal
- MOSI Host to card data signal
- MISO MMC to host data signal

SPI-MMC is byte-orientated and every command, response and data block is built with a byte (8-bit). SPI-MMC messages are built from command, response and data-block tokens. All communication between CP and MMC is controlled by the CP (master).

Serial data transmission through SPI starts when the chip-select (CS) is asserted (ie. when the CS goes to LOW) and ends when the chip-select is released (ie. when the CS goes to HIGH).

Every MMC token transferred on the data signal is protected by CRC bits. But MMC offers a non-protected mode that enables a system built with reliable data links to exclude the hardware or firmware required for implementing the CRC generation and verification functions.

In the non-protected mode, the CRC bits of the command, response and data tokens are still required in the tokens; they are, however, defined as “don’t care” for the transmitters and are ignored by the receivers.

MMC is initialized in the non-protected mode. The CP can turn this option on and off using the CRCONOFF command (CMD39). We assume that CRC is processed by software.

13.8.1 Input and output signals

Figure 13-4: Block diagram of the SPI-MMC on page 13-47 shows the input and output signals of the SPI-MMC circuit. Remember that the APB is an internal peripheral bus.

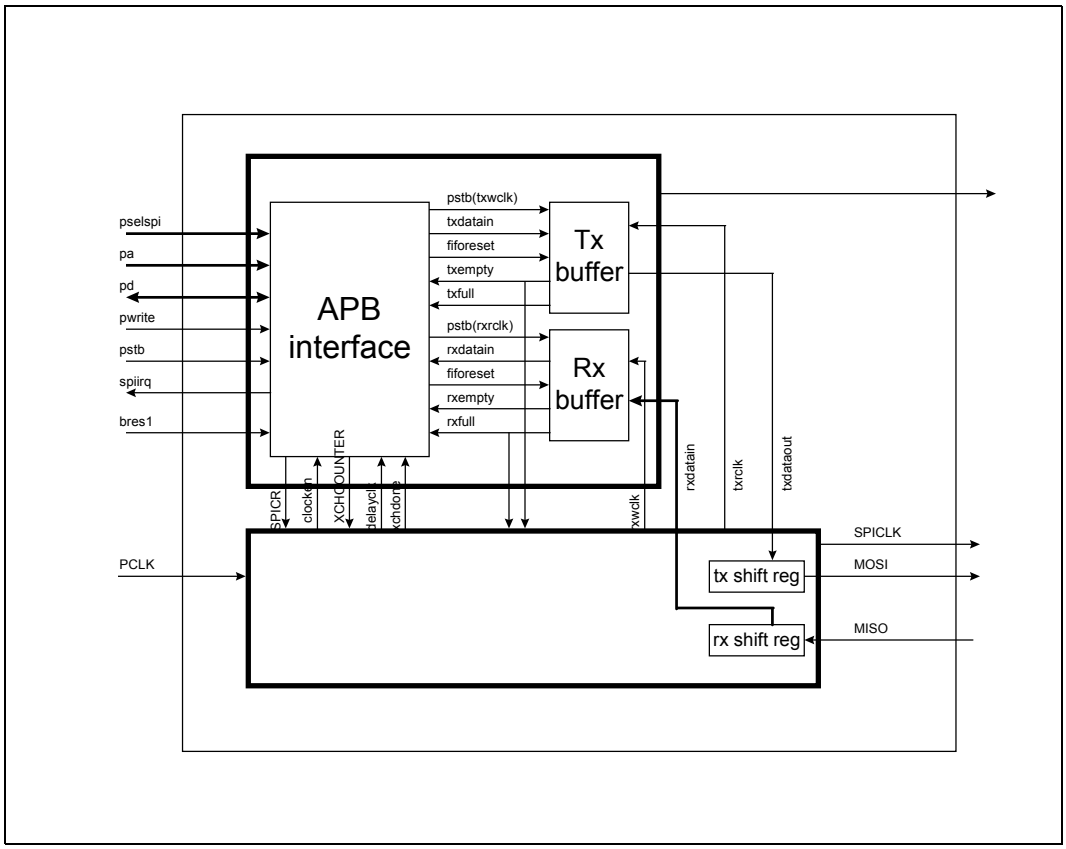


Figure 13-4: Block diagram of the SPI-MMC

The following signals come from (or go to) the APB bus controller of the ARM core.

Name	Description
PA[5:2]	This is part of the peripheral address bus, and is used by the peripheral for decoding its own register accesses. The addresses become valid before **PSTB** goes HIGH and remain valid after **PSTB** goes LOW.
PSPISEL	When HIGH, this signal indicates the SPI-MMC module has been selected by the APB bridge. This selection is a decode of the system address.
PD[15:0]	This is part of the peripheral data bus. The data bus is driven by this block during read cycles (when **PWRITE** is LOW).
PSTB	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of PSTB is coincident with the falling edge of **BCLK**.
PWRITE	When HIGH, this signal indicates a write to a peripheral, and when LOW, a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.

Table 13-34: Signal description

Name	Description
BnRES	Reset signal (active LOW)
SPIIRQ	This signal goes to HIGH if this interrupt is enabled and if either the TX or RX operation is completed.
PCLK	This input clock signal has the frequency of 3.6864MHz

Table 13-34: Signal description (Continued)

The SPI-MMC has four signals connected to the external MMC:

- CS - Chip select signal for external MMC
- SPICLK - Serial clock signal to the external MMC
- MOSI - Serial data output signal to the external MMC
- MISO - Serial data in signal from the external MMC

13.8.2 Overall structure and operation

A block diagram of the SPI-MMC is shown in ***Figure 13-5: SPI-MMC block diagram overview***.

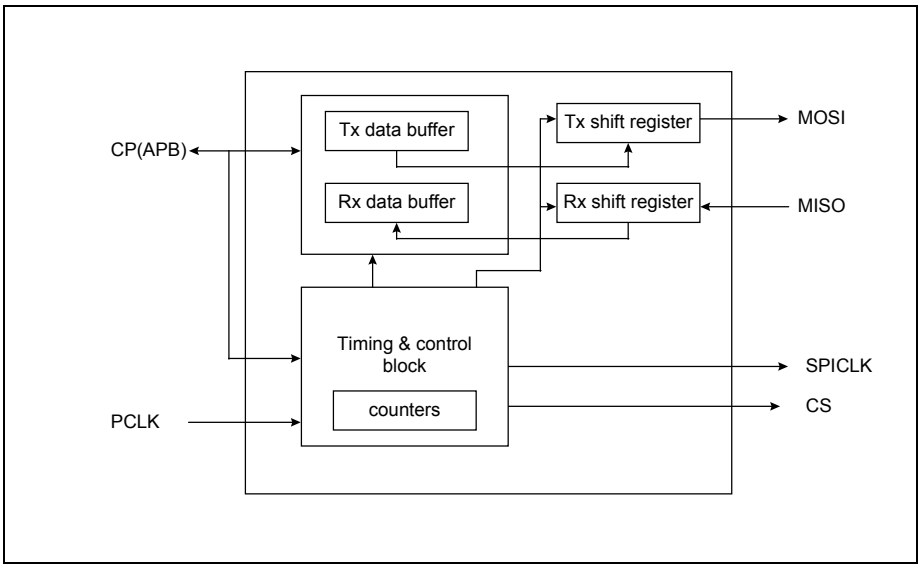


Figure 13-5: SPI-MMC block diagram overview

The TX FIFO and RX FIFO in ***Figure 13-5: SPI-MMC block diagram overview*** are FIFO buffers. In the current design, it is assumed that each buffer contains eight entries, where each entry is 8-bit wide.

After CP writes a sequence of data to the TX FIFO, the content of the FIFO is loaded into the TX shift register and is shifted out serially one byte at a time. When all elements in the TX FIFO are transferred to the TX shift register, the SPI-MMC issues an interrupt to CP, which may fill the TX FIFO for further data transfer.

Serial input data is shifted into the RX shift register. After 8 bits are shifted in, the content of the RX shift register is copied into the RX FIFO. When the RX FIFO is full, the SPI-MMC issues an interrupt to CP through the **SPIIRQ** signal. CP reads the content of the RX FIFO in an interrupt service routine.

The timing and control block produces all necessary control signals of the SPI-MMC block including **SPICLK**. The frequency of **SPICLK** signal is programmable.

SPI-MMC transfer's protocol is command and response. Whenever CP sends a command to MMC (via SPI), MMC sends CP (via SPI) a response. The response is variable length for command—for example, there is 1-, 6-, 17-byte. There is only 6 byte in command.

Consider the sequence of operations that occur in a read transfer.

1 CP send a reset signal to the SPI-MMC block. In other word, CP write "0" to bit in the ResetReg register. The signal is used to clear counters inside the block.Before new exchange begins and the content of XCHCOUNTER is changed, and transmit mode is changed (XCHMODE BIT in the SPICR), CP must send a reset signal to the SPI-MMC block.

2 First, CP set up the SPICR register. In this example, XCHMODE is send mode.

3 CP write number to send into XCHCOUNTER register.

4 CP write "Data read command(CMD17)" into the TX FIFO.

5 CP asserts CS signal. In other words, CP write 0 to CS bit in the SPICR.

6 CP send a start signal to SPI-MMC. In other word, CP set XCH bit in the SPICR.

7 The SPI-MMC block sends out 6 byte of command data from TX FIFO through TX shift register.

8 The SPI-MMC block issues the interrupt after it send all data in TX FIFO.

9 The CP reads the SPISR register in The SPI-MMC block and disable start signal (reset XCH bit). In other words, CP writes the SPICR register.

10 CP send a reset signal to the SPI-MMC block. In other word, CP write 0 to bit in the ResetReg register. The signal is used to clear counters inside the block. Before new exchange begins and the content of XCHCOUNTER is changed, and transmit mode is changed (XCHMODE BIT in the SPICR), CP must send a reset signal to the SPI-MMC block.

11 CP changes transmit mode.(XCHMODE is receive mode)

12 The CP write number to be received into XCHCOUNTER register.

13 CP send a start signal to SPI-MMC (set XCH bit).

14 Then SPI-MMC block receives response from MMC.

15 After SPI-MMC receives 1 byte (for CMD17 command), it sets XCH DONE status bits and it issues interrupt to a CP.

16 The CP reads the SPISR register in the SPI-MMC block and disable start signal (reset XCH bit). In other words, CP writes the SPICR register.

17 The CP read data RX fifo.

18 After CP takes this response data and examine it, CP act as response data.If there is no error indication in response, CP informs SPI-MMC block that MMC sends data to it.

19 CP sends a reset signal to the SPI-MMC block. In other words, CP write 0 to bit in the Reset register. The signal is used to clear counters inside the block. Before new exchange begins and the content of XCHCOUNTER is changed, and transmit mode is changed (XCHMODE BIT in the SPICR), CP must send a reset signal to the SPI-MMC block.

20 The CP write number to be received into XCHCOUNTER register.

21 CP send a start signal to SPI-MMC (set XCH bit).

22 The SPI-MMC block receives data from MMC (for example, data length is from 4 byte to 515 byte.)

23 If SPI-MMC receives data like RX FIFO size, SPI-MMC block sets the "RX FIFO full" status bit and issues an interrupt to CP.At this time SPICLK disable start signal for prevention of RX FIFO overrun.If CP takes all data in RX FIFO, CP sends a start signal a and receives response to remain. Repeat it.

24 After SPI-MMC block receive all data from MMC, it sets the XCH DONE status bit and issues an interrupt to CP.

25 The CP reads the SPISR register in the SPI-MMC block and disable start signal (reset XCH bit). In other words, CP writes the SPICR register.

26 After CP take last data from RX FIFO, CP de-asserts CS signal.

13.8.3 Register map

Table 13-35: SPI-MMC block register map shows a register map of the SPI-MMC block.

Offset	Register Name	Type	Value in reset	Description
0x00	SPICR	R/W	0100000	SPI control register
0x04	SPISR	R	00000000	SPI status register
0x08	XCHCOUNTER	R/W	00000000000	Number of exchange data
0x0C	TXdatabuffer	W		TX data buffer (8*8 bits)
0x10	RXdatabuffer	R		RX data buffer (8*8 bits)
0x14	Testregister1	R	000000000	SPI test register1
0x18	Testregister2	R	000000000	SPI test register2
0x1C	ResetReg	R/W	0”	SPI reset register
0x20	Dummy	R	Does not exist	Test clock generation
0x24	TIC	R		TIC register

Table 13-35: SPI-MMC block register map

6	5	4	3	2	1	0
DATA RATE	CS	xchmode	TestMode	LOOP	SPIEN	XCH

SPICR register

DATA RATE — These bits select the baud rate of the **SPICLK** based on divisions of the system clock. The master clock for the SPIMMC is **PCLK**.The bits are encoded as:

0 = bypass

1 = Divide by 2

CS	This bit is Chip select signal. In order to communicate external device(MMC), CP asserts 0 in this bit. 0 = when CP can exchange data with external device (MMC) 1 = when CP cannot exchange data with external device (MMC)
XCHMODE	This bit determines the direction of transfer 0 = when CP have valid data to send to MMC (send mode) 1 = when CP have valid data to receive from MMC (receive mode)
TestMode	When TestMode bit is set, SPI-MMC block is in TIC mode. When Tic mode, the operation of the SPI-MMC is same in normal mode except that Clock source is not **PCLK** but **TCLK** which is made in the block. 0 = Normal operation 1 = The SPI-MMC block is in TIC mode
LOOP	When set, this bit selects the local loopback operation. The transmitter output is internally connected to the receiver input. When in loopback mode, the operation of SPI-MMC block is same in normal mode except MISO is internally connected MOSI. 0 = Normal operation 1 = The SPI-MMC block is in loopback mode
SPIEN	This bit enables the SPIMMC. The enable should be asserted before initiating an exchange and should be negated after the exchange is complete. When the SPIEN bit is cleared, consumes minimal power. 0 = SPI master disable 1 = SPI master enable
XCH	This bit triggers the state machine to generate clocks at the selected bit rate. 1 = Initiate exchange 0 = No exchange occurs

SPISR register

7	6	5	4 ... 0
TX empty	XCHDONE	Rx full	Reserved

TX empty	This bit is set when TX data buffer is empty. If TX empty goes HIGH, a serial peripheral interrupt is generated. Clearing the TX empty bit is accomplished by reading the SPISR.
XCHDONE	This bit is set when exchange is completed between CP and MMC. If XCHDONE bit goes HIGH, a serial peripheral interrupt is generated. Clearing the XCHDONE bit is accomplished by reading the SPISR.

RX full	This bit is set when RX data buffer is full. If RX full bit goes HIGH, a serial peripheral interrupt is generated. Clearing the RX full bit is accomplished by reading the SPISR.

XCHCOUNTER register

XCHCOUNTER	Number of bytes to be exchanged between CP and SPI.

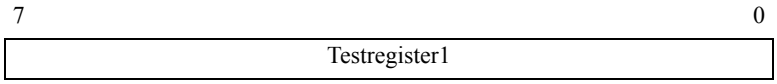
Txdatabuffer register

This 8-bit register is an entry point of the TX FIFO. When CP writes an 8-bit data to this register, the SPI-MMC block shifts the content of the TX FIFO and appends the new data to the FIFO.

Rxdatabuffer register

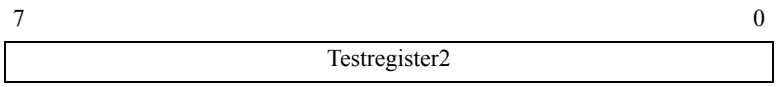
This register is the access point of the RX FIFO. When CP reads one data item from this register, the SPI-MMC block shifts the RX FIFO so that the next data item becomes available at this location.

Test register1



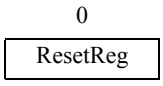
One bit data from MISO pin shifts in Testregister1 when **SPICLK** is rising.

Test register2



One bit data from MOSI pin shifts in Testregister2 when **SPICLK** is rising.

ResetReg



RESET	When CP writes 0 to this location, all registers and counters of the SPI-MMC block are cleared.

TCLK register

When this register is read or written in the **TIC** mode, **TCLK** is generated.

TIC register

This test register allows the SPIMMC output signal to the MMC to be read back.

1	0
SPIclk	CS

13.8.4 Signal timing

All timing diagrams use the following schematics and abbreviations.

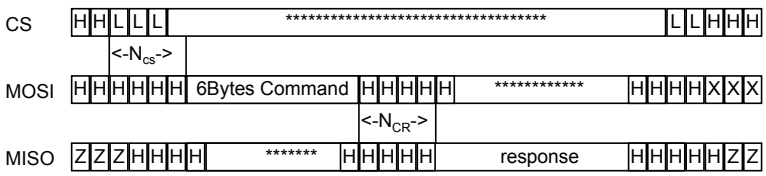
Name	Description
H	Signal is HIGH (logical ‘1’)
L	Signal is LOW (logical ‘0’)
X	Don’t care
Z	High impedance state
*	Repeater
Busy	Busy token
Command	Command token
Response	Response token
Data block	Data token

Table 13-36: Timing diagram abbreviations

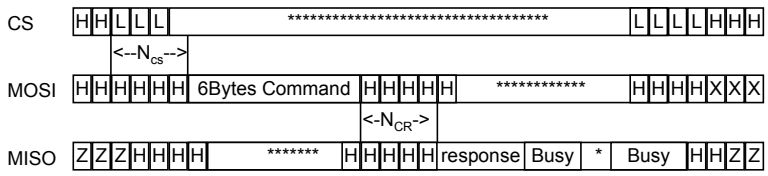
All timing values are defined as outlined below.

Command/Response

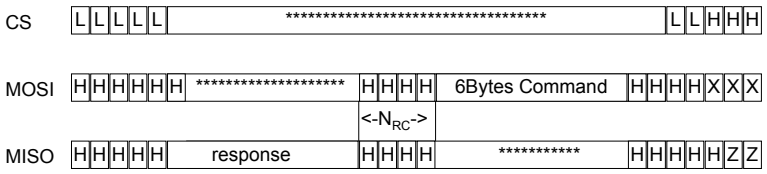
Host command to card response: card is ready



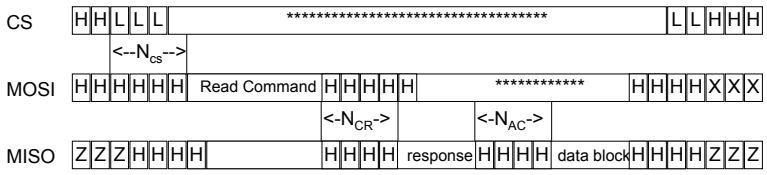
Host command to card response: card is busy



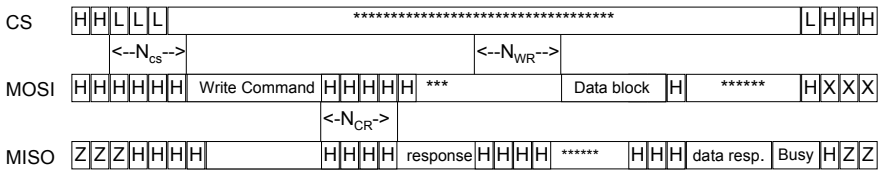
Card response to host command:



Data read



Data write



Timing constants definitions

Name	Minimum	Maximum	Unit
N_{CS}	0	-	8 clock cycles
N_{CR}	1	2	8 clock cycles
N_{RC}	1	-	8 clock cycles
N_{AC}	1		8 clock cycles
N_{WR}	1	-	8 clock cycles

Table 13-37: Timing constants definitions

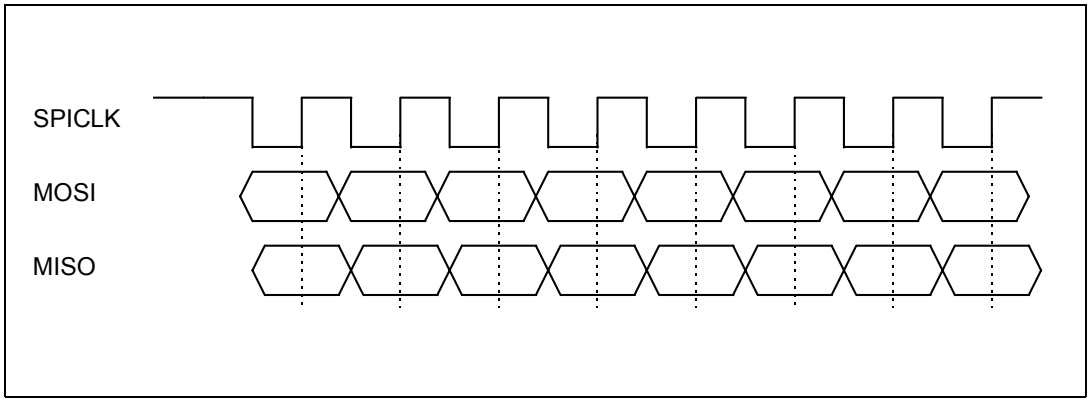


Figure 13-6: Input and output timing diagram

13.9 Analog Front End, AFE (CODEC Interface)

13.9.1 Module overview

The Analog Front End (AFE) interface is an APB peripheral which allows direct connection to a telephony-type CODEC. It provides all the necessary clocks and timing pulses, and also performs the parallel-to-serial conversion on output data and serial-to-parallel conversion on input data. The interface is full duplex, and contains two separate data FIFOs.

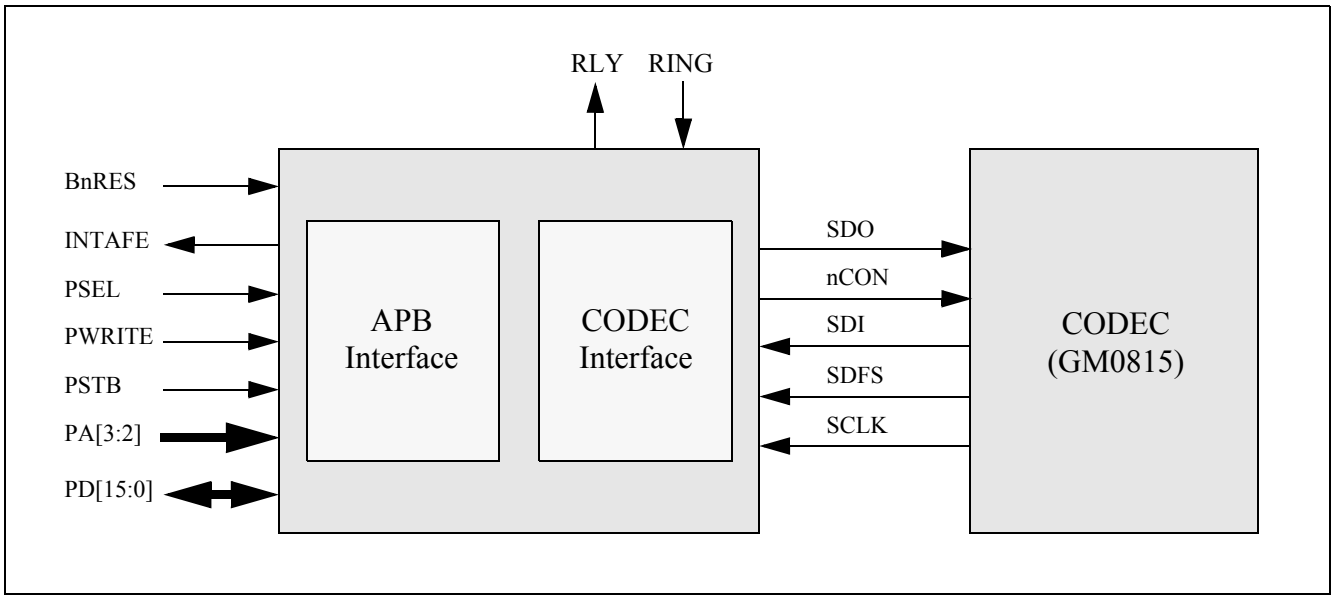


Figure 13-7: Signal connections of the AFE interface

Data is transferred to or from the CODEC at 56Kb per second. The data is either written to, or read from, the appropriate 32-byte FIFO. An interrupt is generated when the FIFO is empty, when an error occurs or when the ringing signal is detected.

13.9.2 Signal description

The AFE Interface module is connected to the APB. ***Table 13-38: Signal descriptions*** describes the relevant APB signals. ***Table 13-39: Specific block signal descriptions*** on page 13-57 shows the non-AMBA signals from the block.

Name	Type	Source/ Destination	Description
BnRES	In	Reset Controller	This signal indicates system reset status of the bus (active LOW).
PA[6:2]	In	APB Bridge	This is part of the peripheral address bus, which is used by the peripheral for decoding its own register accesses. The addresses become valid before **PSTB** goes HIGH and remain valid after **PSTB** goes LOW.

Table 13-38: Signal descriptions

Name	Type	Source/ Destination	Description
PD[31:0]	InOut	APB Peripherals, **BD**	This is part of the bidirectional peripheral data bus. The data bus is driven by this block during read cycles (when **PWRITE** is LOW).
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of **PSTB** is coincident with the falling edge of **BCLK** (ASB system clock).
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral, and when LOW, a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates the AFE module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB). For more details see *AMBA Peripheral Bus Controller* (ARM DDI 0044).
UARTRING	Out	UART	Ring detect signal to UART.

Table 13-38: Signal descriptions

Name	Type	Source/ Destination	Description
nCON	Out	CODEC	Serial control/data input select (control at LOW).
RING	In	DAA	Ringing signal input.
RLY	Out	DAA	Relay control output.
SDI	In	CODEC	Serial data input/ Control data input.
SDO	Out	CODEC	Serial data output. Active when TREN bit is set, otherwise held LOW.
SDFS	In	CODEC	Serial data frame synchronous signal input.
SCLK	In	CODEC	Serial data clock input at a frequency of 864kHz.
INTAFE	Out	Interrupt Controller	AFE Interrupt. Active HIGH.

Table 13-39: Specific block signal descriptions

13.9.3 Functional Description

The AFE interface is a serial-to-parallel and parallel-to-serial converter providing full duplex transmission with an external serial AFE(CODEC). Two 32-byte FIFOs are provided to help optimize processor usage. Data is clocked in and out of the block with the **SCLK** signal which runs at 864kHz.

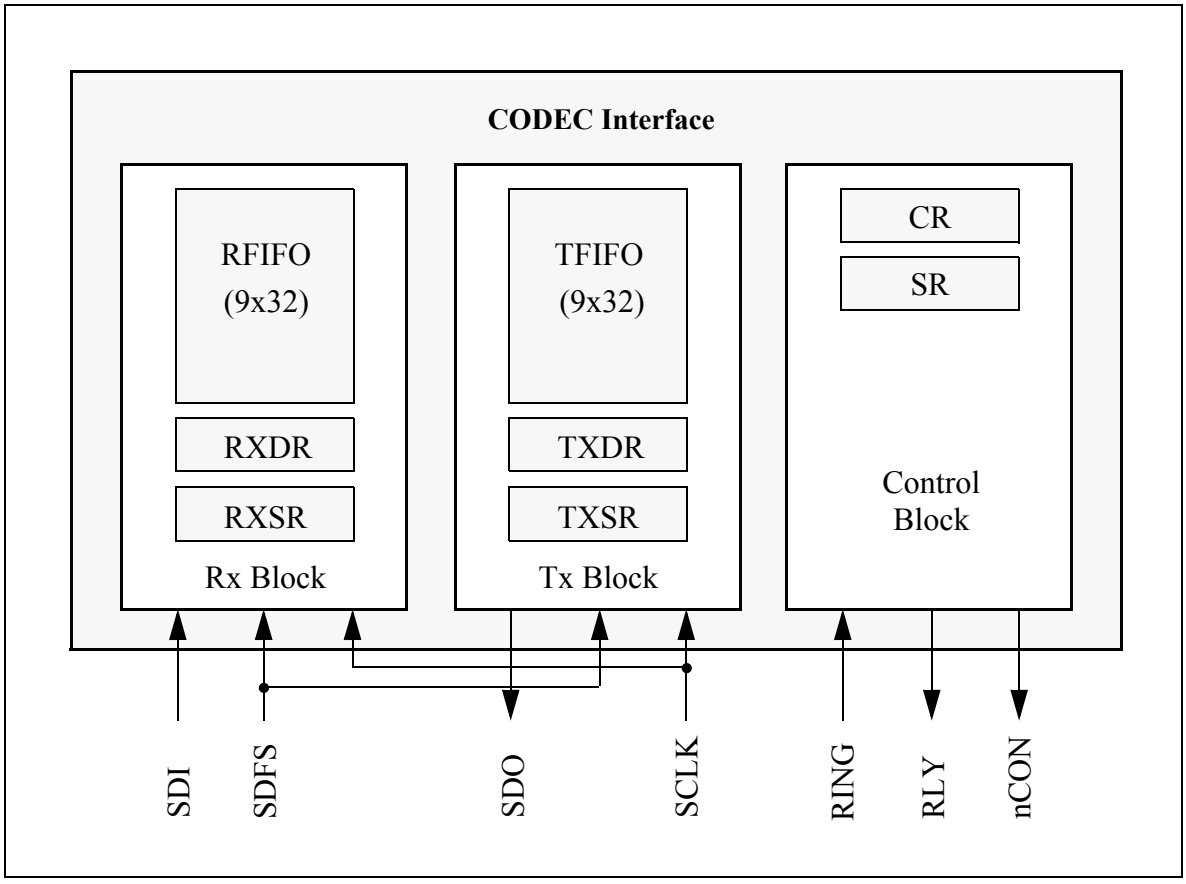


Figure 13-8: AFE Interface block diagram

Transmit and receive modes are enabled by asserting TREN bit in the control register. When asserted, the FIFO is enabled. Additionally, if the TREN bit is cleared, the SDO output is disabled. Asserting the enable bit causes the interrupt generation logic to become active, otherwise it is disabled.

Data is loaded into the transmit FIFO by writing to the TXDR register. At the beginning of a transmit cycle, this data is loaded into a shift register where it is shifted out serially to SDO, MSB first, according to the CODEC protocol mode (See ***Figure 13-9: CODEC protocol diagram*** on page 13-59).

Data is received by taking data in serially through SDI, again MSB first, shifting it through the shift register and loading the complete half-word into the receive FIFO when a half-word has been received.

The Status register is provided to indicate the status of the FIFOs, whether an interrupt occurred, and whether an error occurred.

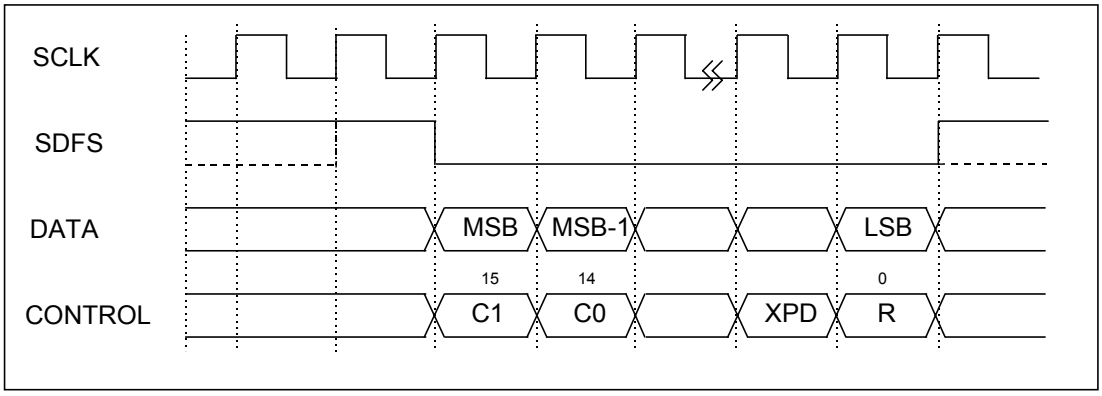


Figure 13-9: CODEC protocol diagram

Interrupts

The AFE Interface has three interrupt sources that assert an active HIGH interrupt. These interrupts are unmasked by setting the enable bits in the CR to 1.

Interrupt Source	Issuance Condition	Method to clear Interrupt
TXFEI	Transmit-FIFO empty	After read SR
RINGI	RING input is HIGH	After read SR
FIFOERRORI	FIFO Error (no data to send)	After read SR

Table 13-40: Interrupt sources

13.9.4 Programmer's Model

AFE registers

The following user registers are provided:

- Transmit Data Register (TXDR)
- Receive Data Register (RXDR)
- Reference Value Register (RVR)
- Control Register (CR)
- Status Register (SR)

TXDR (Transmit Data Register) is a 32-bit write-only register, in which transmit data is stored. Writes to this register will push the data into the transmit FIFO. All the bits in this register are initialized to 0 at reset.

Bit	Name	Function
31:0	data	Transmit data

Table 13-41: TXDR bit functions

RXDR (Receive Data Register) is a 32-bit read-only register, in which receive data is stored. Reads from this register will give the data stored in the receive FIFO. All the bits in this register are initialized to 0 at reset.

Bit	Name	Function
31:0	data	Receive data

Table 13-42: RXDR bit functions

CR (Control Register) is an 8-bit read/write register, which is used to control the AFE interface. All the bits in this register are initialized to 0 at reset.

Bit	Name	Function
7	PWDN	Power down mode 0 : Power down mode (initial value) Stops codec signals (SCLK, SDFS) 1 : Normal mode
6	LOOP	Internal Loopback mode 0 : Normal mode (initial value) 1 : Loopback mode
5	TREN	Transmit /Receive Enable 0 : disable 1 : enable
4	SCDS	Serial Control/ Data output select 0 : Serial Data output (initial value) 1 : Serial Control output
3	RLYEN	RLY output value 0 : RLY set LOW 1 : RLY set HIGH
2	nRINGIM	Ring Interrupt Enable 0 : Interrupt disable 1 : Interrupt enable
1	nTXRXIM	FIFO Empty Interrupt Enable 0 : Interrupt disable 1 : Interrupt enable
0	nFEIM	FIFO Error Interrupt Enable 0 : Interrupt disable 1 : Interrupt enable

Table 13-43: CR bit functions

RVR (Reference Value Register) is an 11-bit read-only register which is used to indicate the value of RING input, the difference of read pointer position from write pointer position in the FIFO for transmit/receive.

Bit	Name	Function
10	UARTRING	This bit shows the state of the ring signal to the UART block.
9	TxFifoFull	When the transmit FIFO is full, this bit is set. When FIFO is full, TxPosi and RxPosi values are zero.
8	RINGIV	RING input value
7:4	TXPD	Difference of read/write pointer for TX FIFO (bits 3:0)
3:0	RXPD	Difference of read/write pointer for RX FIFO (bits 3:0)

Table 13-44: RVR bit functions

SR (Status Register) is a 3-bit read-only register which is used to indicate the status of the AFE interface. These bits show the interrupt sources regardless of interrupt disable.

Bit	Name	Function
2	RINGI	RING interrupt 0 : No interrupt detected (initial value) 1 : Interrupt occurred
1	FEMPTY	FIFO empty (active HIGH) interrupt 0 : No interrupt detected (initial value) 1 : Interrupt occurred
0	TXR	TX error (active HIGH) interrupt 0 : No interrupt detected (initial value) 1 : Interrupt occurred

Table 13-45: SR bit functions

UARTCR (UART Control Register) is a 1-bit read/write register, which controls RING signal to UART block.

Bit	Name	Function
0	RINGEN	Enable RING to UART 0 : RING signal is connected to UART **nRI** 1 : disable mode (initial value) In this mode, RING signal to UART is always HIGH.

Table 13-46: UARTCR bit functions

TIR (Test Register for Input) is a 5-bit write-only register defined for test purposes. This register allows simulation of input signals to the block, as well as the generation of a special test clock signal for use with production test vectors.

Bit	Name	Function
4	TNFLAG	Mode select bit 0 : Normal operation mode 1 : Test mode
3	TSCLK	Programmable serial clock for test
2	TSDFS	Programmable data frame sync. for test
1	TSDI	Programmable serial data input for test
0	TRING	Programmable ring input for test

Table 13-47: TIR bit functions

TOR (Test Register for Output) is a 3-bit read-only register defined for test purposes. This register allows simulation of input signals to the block, as well as the generation of a special test clock signal for use with production test vectors.

Bit	Name	Function
2	TSDO	Serial data output line
1	TNCON	Serial control/data input select output line
0	TRLY	Relay control output line

Table 13-48: TOR bit functions

Register memory map

The base address of the AFE interface is selectable by software. The offset of any particular register from the base address is shown below.

Address	Register	Read location	Write location
AFE Base	CR	Control Register	Control Register
AFE Base + 0x04	SR	Status Register	
AFE Base + 0x08	RVR	Reference Value Register	
AFE Base + 0x0C	UARTCR	UART Control Register	UART Control Register
AFE Base + 0x10	TIR		TIR
AFE Base + 0x14	TOR	TOR	
*AFE Base + 0x20–0x3C	TX		Transmit Data Register
*AFE Base + 0x40–0x5C	RX	Receive Data Register	

Table 13-49: AFE Interface register memory map

Note: *The asterisk denotes that access to any address in the range produces the same result.*

Communication Procedure

The modem codec (GM0815) can receive normal data or codec control data with an nCON pin. Procedures 1–3 below are for sending codec control data to GM0815. If there is no control data, procedures 1 to 3 are not necessary.

Procedures 4 to 8 are for sending/receiving normal data to/from GM0815. This peripheral shares one interrupt request for both directions (ie. transmit/receive) of data transfer, so outgoing and incoming data should be synchronized for FIFO fill and empty.

1. In order to send control data to the external modem codec, program the control register to drive nCON output pin to LOW and enable AFE.
2. Send control data by writing two words to the transmit FIFO. These four half words have the same data to synchronize timing with codec.
3. When there is an interrupt request which indicates that FIFO needs to be filled, then drive nCON pin HIGH, which represents that AFE is in normal data transfer mode. Normally nCON pin should be HIGH.
4. Disable AFE to initialize FIFO.
5. To transmit and receive data, enable AFE (nCON pin should stay HIGH).
6. Write the outgoing data to the transmit FIFO. There will be no interrupt for the first data request. The data size should be 1 + burst size of words (32 bit). If software requires an eight-word burst, then it should write nine words first. Then write/read eight words to/from transmit/receive FIFO on each interrupt. The burst size can be selected from one to eight words.
7. For each FIFO fill interrupt request, write outgoing data and then read the incoming data. FIFO over-/underrun error is checked only by transmit part, and incoming data has more latency than outgoing data.
8. Interrupt should be serviced before one word (two data) transmit time.
9. The last word cannot be transmitted if AFE is disabled before completing a transfer. To confirm it, include one filler word as the last one and wait until FIFO fill interrupt request is generated.

If there is a FIFO error, software should disable AFE to initialize FIFO again.

13.10 Real Time Clock

13.10.1Module Overview

This module is a 32-bit counter clocked by a 32768Hz clock. This clock needs to be provided by the system, as there is no oscillator inside the block. The clock is divided in the RTC core to provide a 1Hz clock which is used to drive a 32-bit counter which forms the Real Time Clock (RTC). It also contains a 32-bit match register which can be programmed to generate an interrupt signal when the time in the RTC matches the specific value written to this register (alarm function - RTC event). The RTC has two event outputs, one which is synchronized to PCLK, SRTCEV, and the second, URTCEV synchronized to the 32768Hz clock. SRTCEV is to be connected to the system interrupt controller, and URTCEV is used in the PMU to provide a system alarm wakeup.

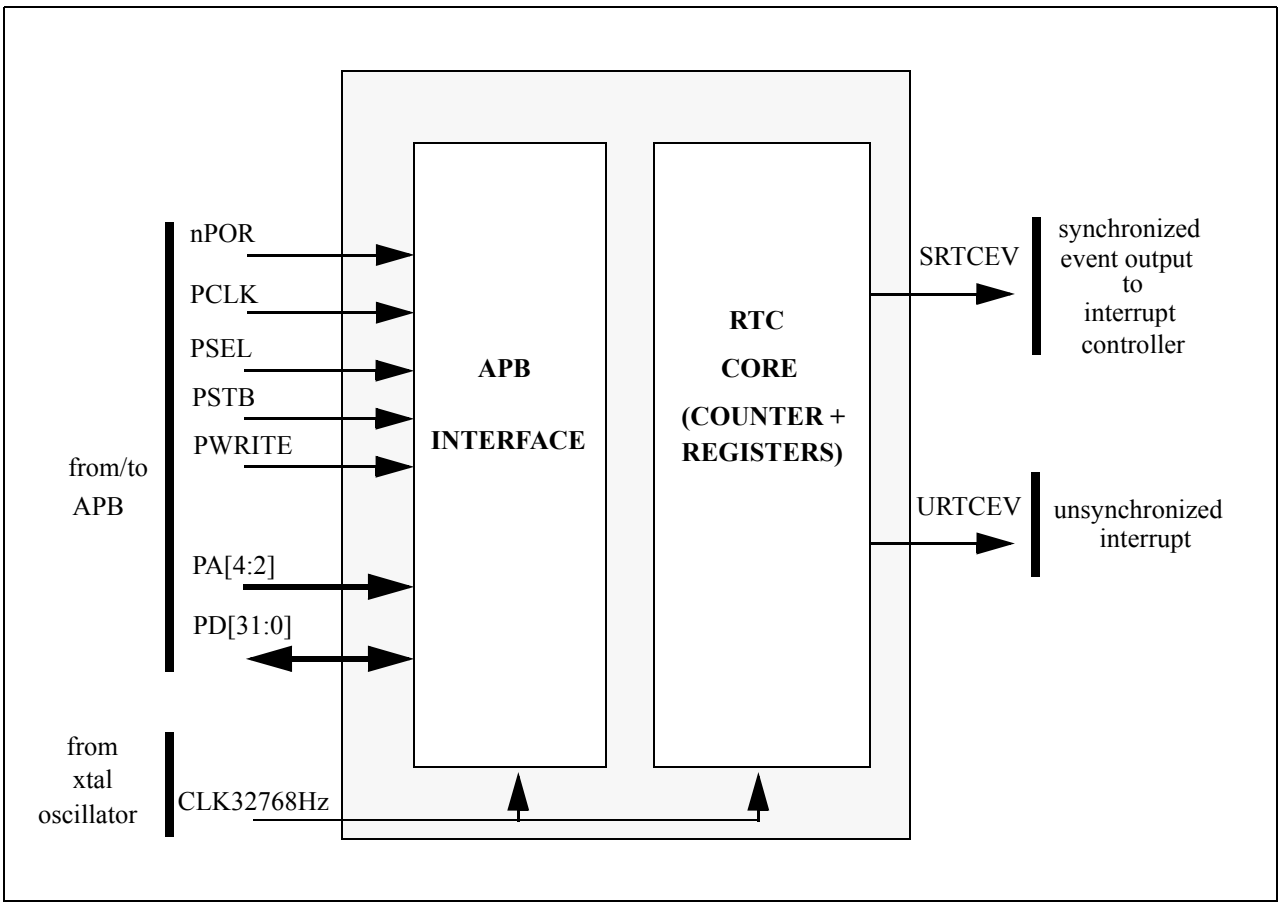


Figure 13-10: Real Time Clock connections diagram

13.10.2Signal Description

The RTC module is connected to the APB. ***Table 13-50: APB signal descriptions*** describes the APB signals used and produced. ***Table 13-51: Specific block signal descriptions*** shows the non-AMBA signals from the block.

Name	Type	Source/ Destination	Description
nPOR	In	External	Power on reset.
PA[4:2]	In	APB Bridge	This is part of the peripheral address bus, which is used by this peripheral for decoding its own register accesses. The addresses become valid before **PSTB** goes HIGH and remain valid after **PSTB** goes LOW.
PD[31:0]	InOut	APB Peripherals, BD bus	This is the bidirectional peripheral data bus. The data bus is driven by this block during read cycles (when **PWRITE** is LOW).
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of **PSTB** is coincident with the falling edge of **BCLK** (ASB system clock).
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral and when LOW, a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates the RTC module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB). For more details, see *AMBA Peripheral Bus Controlle*r (ARM DDI 0044).
PCLK	In	APB Clock Gen	The slow APB clock used to re-synchronize data transfers between the 32768Hz clock and the APB.

Table 13-50: APB signal descriptions

Name	Type	Source/ Destination	Description
CLK32KHZ	In	Clock generator	32768Hz clock input. This is the signal that clocks the counter during normal operation.
SRTCEV	Out	APB peripheral (Interrupt Controller)	Interrupt signal to the Interrupt module. When HIGH, this signal indicates a valid comparison between the counter value and the match register. It also indicates 1Hz interval with enable bit in control register.
URTCEV	Out	APB peripheral	When HIGH, this signal indicates a valid comparison between the counter value and the match register. This signal is used to wake up the GM30C7201 when it is asleep.
CLK4K	Out	APB peripheral	This signal is used in the power management block.

Table 13-51: Specific block signal descriptions

13.10.3Functional Description

The counter is loaded by writing to the RTC data register. The counter will count up on each rising edge of the clock and loops back to 0 when the maximum value (0xFFFFFFFF) is reached. At any moment the counter value can be obtained by reading the RTC data register.

The value of the match register can also be read at any time, and the read does not affect the counter value. The status of the interrupt signal is available in the status register. The status bit is set if a comparator match event has occurred or 1 second has elapsed. Reading from the status register will clear the status register.

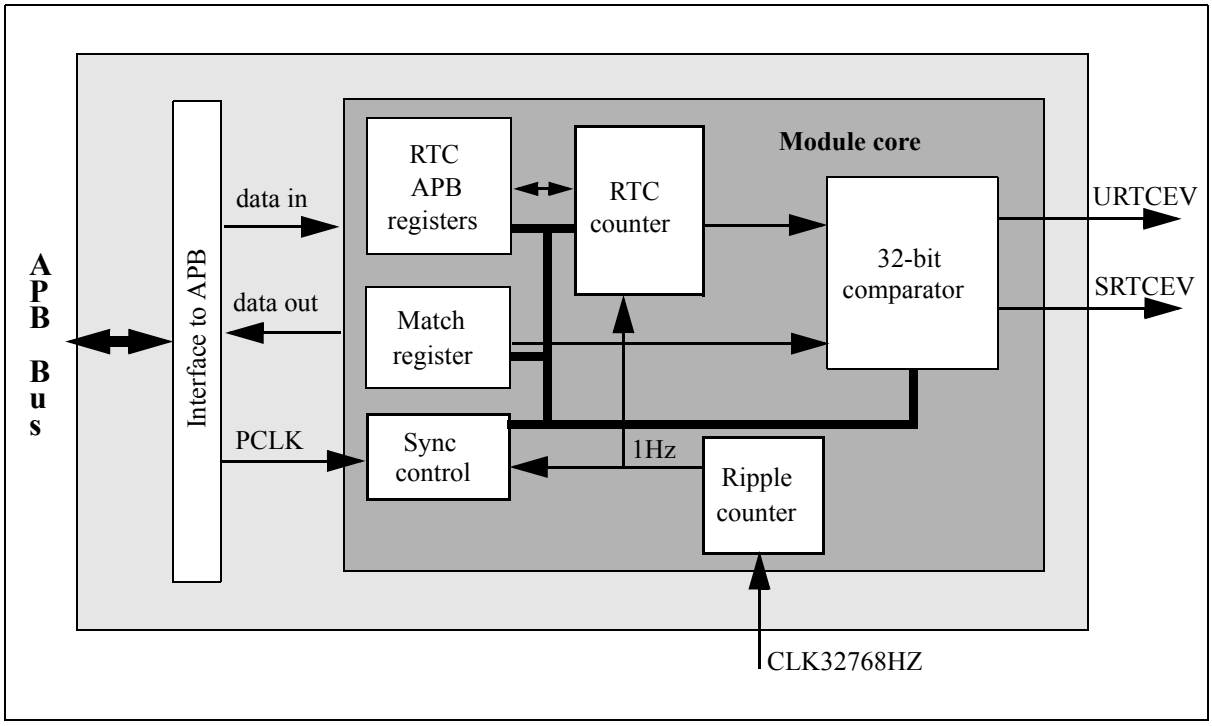


Figure 13-11: RTC block diagram

13.10.4Programmer's Model

RTC registers

The following user registers are provided:

Register	Name	Type	Description
RTCDR	RTC Data Register	Read/Write	Writing to this 32-bit register will load the counter. A read will give the current value of the counter.
RTCMR	RTC Match Register	Read/Write	Writing to this 32-bit register will load the match register. This value can also be read back.
RTCS	RTC Status	Read-only	When performing a read from this location the interrupt flag will be cleared. If a match event has occurred, bit[1] will be set. For a second event, bit[0] will be set. This register is affected by the control register.

Table 13-52: RTC register description

Register	Name	Type	Description
RTCDV	RTC Clock Divider	Read/Write	Reads to the register will return only four bits of the clock divider output. Bits [3:0] will return bits (14,11, 7, 3) of the divider output. Write zero to bit[0] clears this divider.
RTCCR	RTC Control	Read/Write	This register enables the interrupt. Bit[1] enables the match event interrupt (default disable = 0). Bit[0] enables second event interrupt (default disable = 0).
RTCTS	RTC Tic Selection	Write-only	This register is for production test purposes. Bit[0] enables TicCLK32K for 32kHz clock replacement. Bit[1] enables TicCLKPCLK for PCLK clock replacement.
TicCLK32K		Write-only	This generates 32kHz clock for production test purposes.
TicCLKPCLK		Write-only	This generates PCLK clock for production test purposes.

Table 13-52: RTC register description (Continued)

Register memory map

The base address of the RTC is not fixed and may be different for any particular system implementation. The offset, however, of any particular register from the base address is determined.

Address	Read location	Write location
RTC Base	RTC data register (RTCDR)	RTC data register (RTCDR)
RTC Base + 0x04	RTC match register (RTCMR)	RTC match register (RTCMR)
RTC Base + 0x08	RTC status (RTCS)	Reserved
RTC Base + 0x0C	RTC clock divider (RTCDV)	RTC clock divider (RTCDV)
RTC Base + 0x10	RTC control register (RTCCR)	RTC control register
RTC Base + 0x14	Reserved	RTC Tic selection register (RTCTS)
RTC Base + 0x18	Reserved	TicCLK32K
RTC Base + 0x1C	Reserved	TicCLKPCLK

Table 13-53: RTC register memory map

Note *The RTC clock divider register may only be written to when in test mode.*

13.11 Analog–Digital Converter Interface Controller (AIC)

13.11.1 Overview

The AIC is a peripheral which includes an ADC, and allows the CPU to read touch panel position, battery level and microphone signals.

Test Mode Notes:

Test Mode 1: In this mode the AIC is tested without using the ADC and touch panel. There is an additional register which provides 8-bit data instead of the ADC output. There are some additional registers which allow all the AIC output signals to the ADC to be read back. These registers can be written to by software.

Test Mode 2: In this mode, the ADC is tested. Additional registers provide the input data to the ADC (ADC test). These registers can be written to by software. There is an additional register to allow the ADC data output signal (AD signal) to the AIC to be read back.

Features:

- 8kHz and 11.02kHz counter operation clock
- Software Control of ADC operation
- Five analog input channels, allocated to three functions:
  1) Touch panel position
  - uses full 10-bit ADC resolution
  - touch panel drive signal generation
  - multiple data reads for each single touch
  - four sample rates
  2) Battery check
  - main and backup battery check
  3) Input sound
  - 8-bit mono sound data
  - two sample rates mode (8kHz or 11.02kHz)
  - 16-byte sound data buffers
- Interrupt status register (for multiple interrupt sources)
- Power down mode

13.11.2 AIC Unit block diagram

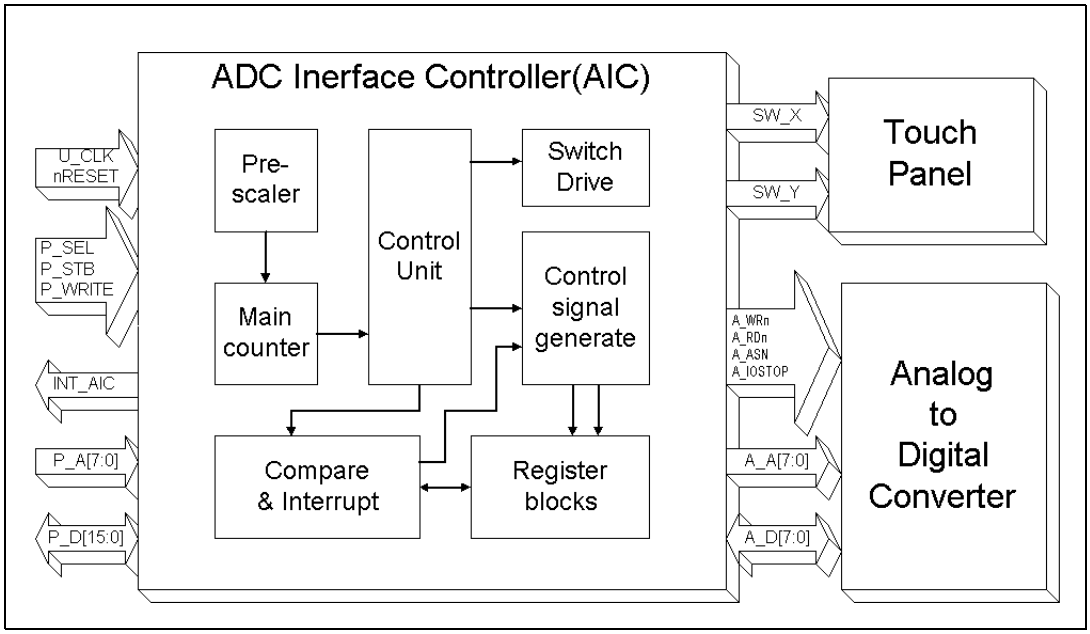


Figure 13-12: AIC module block diagram

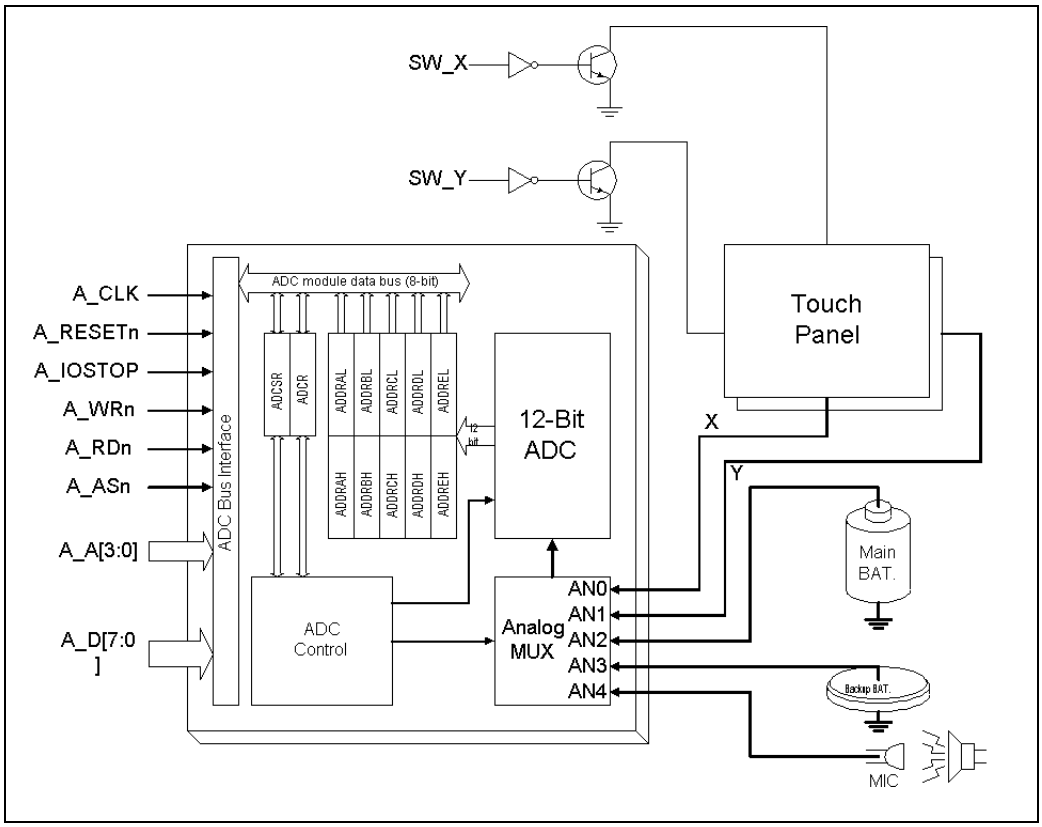


Figure 13-13: ADC block & external connection diagram

13.11.3 AIC Unit hardware interface and signal description

Name	Type	Source/ Destination	Description
PCLK	In	UART Clock	UART clock.
BnRES	In	APB Bridge	Reset signal generated from the APB Bridge.
ACLK	Out	AIC unit/ADC	ADC operation clock output (PCLK or test clk).
PA[7:2]	In	APB Bridge	This is the peripheral address bus, which is used by an individual peripheral for decoding register accesses to that peripheral. The addresses become valid before **PSTB** goes HIGH, and remain valid after **PSTB** goes LOW.
PD[31:0]	InOut	APB Peripherals, BD Bus	This is the bi-directional peripheral data bus. The data bus is driven by this block during read cycles (when **PWRITE** is LOW).
PSTB	In	APB Bridge	This strobe signal is used to time all accesses on the peripheral bus. The falling edge of **PSTB** is coincident with the falling edge of **BCLK**.
PWRITE	In	APB Bridge	When HIGH, this signal indicates a write to a peripheral. When LOW, it indicates a read from a peripheral. This signal has the same timing as the peripheral address bus. It becomes valid before **PSTB** goes HIGH and remains valid after **PSTB** goes LOW.
PSEL	In	APB Bridge	When HIGH, this signal indicates that this module has been selected by the APB bridge. This selection is a decode of the system address bus (ASB). For more details, see *AMBA Peripheral Bus Controller* (ARM DDI0044).
INTAIC	Out	Interrupt Controller	Interrupt request when either touch panel buffer is full, or sound buffer is full, or battery data is checked.
AA[3:0]	Out	AD Converter	This is the address bus used to select a register in the ADC unit.
AD[7:0]	InOut	AD Converter, AIC	This is a the bi-directional data bus which is connected to the ADC.
AASN	Out	AD Converter	Address strobe signal for accessing ADC registers. When there is a signal transition to a LOW state, address is valid.
AWRn	Out	AD Converter	Write strobe signal to write control or status register of ADC. This is an active LOW signal.
ARDn	Out	AD Converter	Read strobe signal to read the contents of ADC registers. This is an active LOW signal.
AIOSTOP	Out	AD Converter	This signal is used to stop input/output operation of the ADC. When this signal becomes HIGH (1), all ADC operations will stop, to save power to the ADC drive.
ATEST	Out	AD Converter	ADC test signal. This signal has the value of 0 in normal mode.
CALTEST	Out	AD Converter	ADC test signal. This signal has the value of 0 in normal mode.
DATA	In	AD Converter	ADC test signal.

Table 13-54: AIC signal descriptions

Name	Type	Source/ Destination	Description
OTR	In	AD Converter	ADC test signal.
P0	In	AD Converter	ADC test signal.
AIADN	In	AD Converter	ADC test signal.
SWXN	Out	AD Converter	Drive signal for the touch panel of X-axis. This signal is used to switch touch panel bias transistors. Note: when driving the touch panel, only one of the **SWXN** or **SWYN** signals go HIGH at any one time.
SWYN	Out	AD Converter	Drive signal for the touch panel of Y-axis. This signal is used to switch the bias transistors. Note: when driving the touch panel, only one of the **SWXN** or **SWYN** signals go HIGH at any one time.
SWXP	Out	AD Converter	Inverted **SWXN** register.
SWYP	Out	AD Converter	Inverted **SWYN** register.

Table 13-54: AIC signal descriptions (Continued)

13.11.4 AIC Unit Register Address Map

Address	Name	Description	R/W	Size (bits)
AIC Base + 0x00	AICTCTR	Touch panel control register	R/W	8
AIC Base + 0x04	AICBCTR	Battery control register	R/W	8
AIC Base + 0x08	AICSCTR	Sound control register	R/W	8
AIC Base + 0x0C	AICPOWER	Power down control register	R/W	8
AIC Base + 0x10	AICSR	Status register	RO	8
AIC Base + 0x30	AICTPDR0	Touch panel 1st sample data register (X,Y)	RO	32
AIC Base + 0x34	AICTPDR1	Touch panel 2nd sample data register (X,Y)	RO	32
AIC Base + 0x38	AICTPDR2	Touch panel 3rd sample data register (X,Y)	RO	32
AIC Base + 0x3C	AICTPDR3	Touch panel 4th sample data register (X,Y)	RO	32
AIC Base + 0x50 to AIC Base + 0x5C	AICSDR0 to AICSDR3	Sound data register 0–3. Each register stores 4 x 8-bits of sound data.	RO	32
AIC Base + 0x70	AICBDR	Main and backup battery data register	RO	16
AIC Base + 0x80	AICTSTCR	Test control register	R/W	8
AIC Base + 0x84	AICTSTR0	This is a test register to allow the clock (ACLK) signal to the ADC to be read back. This register is readable in both normal mode and test mode.	RO	8

Table 13-55: AIC unit register address map

Address	Name	Description	R/W	Size (bits)
AIC Base + 0x88	AICTSTR1	This is a test register that provides 8-bit data (AD) instead of the ADC output in AIC test mode.	R/W	8
AIC Base + 0x90	AICTSTR2	This is a test register to allow AIC output signals to the ADC and to the interrupt controller to be read back (INTAIC, AASN, AWRN, ARDN, AIOSTOP).	RO	8
AIC Base + 0x94	AICTSTR3	This is a test register to allow the AIC output signal to the ADC and the touch panel to be read back (SWXN, SWYN, SWXP, SWYP and AA).	RO	8
AIC Base + 0x98	AICTSTR4	This is a test register to allow the AIC output signal to the ADC to be read back in AIC test mode (AD).	RO	8
AIC Base + 0xC0	ADCTSTR0	This register provides AA, AASN, AWRN, ARDN and AIOSTOP values instead of the AIC output in ADC test mode.	R/W	8
AIC Base + 0xC4	ADCTSTR1	This register provides AD values instead of the AIC output in ADC test mode.	R/W	8
AIC Base + 0xD0	ADCTSTR2	This is a test register to allow the ADC output signal to the AIC to be read back in ADC test mode.	RO	8
AIC Base + 0xD4	ADCTSTR3	This register provides ATEST and CALTEST values.	R/W	8
AIC Base + 0xD8	ADCTSTR4	This is a test register to allow the ADC output signal to the AIC to be read back in ADC test mode (DATA, OTR, PO and AIADN).	RO	16
AIC Base + 0xDC	TCLK	This register is used to generate TCLK in test mode.	Virtual register	0

Table 13-55: AIC unit register address map (Continued)

13.11.5 AIC unit register description

Status Register (AICSR)

The Status Register shows the interrupt sources. When the AIC unit generates the interrupt INTAIC to the CPU, the CPU first reads the status register of the AIC unit. INTS bits of AICSR indicate which device generated the interrupt.

Bit	Initial	Name	Function
7	0	INTTPS	Interrupt touch panel status bit
6	0	INTMBS	Interrupt main battery status bit
5	0	INTBBS	Interrupt backup battery status bit
4	0	INTSDS	Interrupt sound status bit
3–0	-	-	Reserved bits

Table 13-56: AICSR bit functions

Touch Panel Control Register (AICTCTR)

The AIC touch panel control register is used to control touch panel sampling operations. If TPC bit is set to 1, the touch panel data sampling starts (enabled). If TPC is reset to 0, the sampling operation stops (disabled). The AIC unit has eight sampling modes for touch panel data acquisition. The sampling rate is selected by the TCLK bits and SCLK of AICSCTR.

For eight sample rates, TCLK bits have to be set between 00–11, and SCLK of AICSCTR has to be set between 0–1 as shown in ***Table 13-57: AICTCTR bit functions***. The TPC bit should be cleared at the beginning of the Interrupt Service Routine to disable the touch panel.

Bit	Initial	Name	Function
7	0	TPC	Touch panel sampling control bit: 0 - disable 1 - enable
6	0	TMSK	Touch panel interrupt mask bit: 0 - mask 1 - unmask
5–2	-	-	Reserved bits
1–0	00	TCLK	Touch panel data sampling rate selection. When SCLK = 0: 00 - approx. 400 samples/sec 01 - approx. 200 samples/sec 10 - approx. 100 samples/sec 11 - approx. 50 samples/sec When SCLK = 1: 00 - approx. 550 samples/sec 01 - approx. 275 samples/sec 10 - approx. 138 samples/sec 11 - approx. 69 samples/sec

Table 13-57: AICTCTR bit functions

Battery Control Register (AICBCTR)

The AIC battery check control register is used to control the battery check operations. If the MBC bit is set to 1, the battery check operation is enabled for the main battery. If MBC is reset to 0, the check operation is stopped (disabled). The BBC bit performs a similar function for the backup battery. When a battery check operation is needed, the enable bit (MBC,BBC) must first be reset to 0 and then set to 1(start).

Bit	Initial	Name	Function
7	0	MBC	Main battery check control bit: 0 - disable 1 - enable
6	0	MMSK	Main battery interrupt mask bit: 0 - mask 1 - unmask
5	0	BBC	Backup battery check control bit: 0 - disable 1 - enable

Table 13-58: AICBCTR bit functions

Bit	Initial	Name	Function
4	0	BMSK	Backup battery interrupt mask bit: 0 - mask 1 - unmask
3–0	-	-	Reserved bits

Table 13-58: AICBCTR bit functions (Continued)

Sound Control Register (AICSCTR)

The AIC sound sampling control register controls the sound input and voice recording operations. If the SDC bit is set to 1, the input sound data sampling operation is enabled. If SDC is reset to 0, the operation stops (disabled). The SDC bit should be cleared at the beginning of the Interrupt Service Routine to disable the sound sampling.

Bit	Initial	Name	Function
7	0	SDC	Sound sampling control bit: 0 - disable 1 - enable
6	0	SCLK	Sound data sampling rate selection: 0 - 8kHz 1 - 11.02kHz
5		SMSK	Sound interrupt mask bit: 0 - mask 1 - unmask
4–0	-	-	Reserved bits

Table 13-59: AICSCTR bit functions

Touch panel data register 0–3 (AICTPDR0–3)

Four 32-bit registers are used to store two (X, Y axis) 12-bit values from the ADC. (That is, in one 32-bit register the 12 least significant bits are the Y-data, and bits 16–27 are the X-data.) The AIC is connected to the ADC by an 8-bit bus, but connected to the CPU by a 32-bit bus. Therefore the hardware reads the ADC twice to get the 12-bit value.

When a touch panel with a resolution of 1024 bits per axis is used, the software need only use the ten most significant bits (the software should ignore the lower two bits). When the last half-word of the last buffer register is written, INTAIC is generated and CPU reads all four buffer registers simultaneously (burst read).

Bit	Initial	Name	Function
31–28	All 0		Reserved
27–16	All 0	1st TP X data	The 1st touch panel X-axis sample data
15–12	All 0		Reserved
11–0	All 0	1st TP Y data	The 1st touch panel Y-axis sample data

Table 13-60: AICTPDR0 bit functions

Bit	Initial	Name	Function
31–28	All 0		Reserved
27–16	All 0	2nd TP X data	The 2nd touch panel X-axis sample data
15–12	All 0		Reserved
11–0	All 0	2nd TP Y data	The 2nd touch panel Y-axis sample data

Table 13-61: AICTPDR1 bit functions

Bit	Initial	Name	Function
31–28	All 0		Reserved
27–16	All 0	3rd TP X data	The 3rd touch panel X-axis sample data
15–12	All 0		Reserved
11–0	All 0	3rd TP Y data	The 3rd touch panel Y-axis sample data

Table 13-62: AICTPDR2 bit functions

Bit	Initial	Name	Function
31–28	All 0		Reserved
27–16	All 0	4th TP X data	The 4th touch panel X-axis sample data
15–12	All 0		Reserved
11–0	All 0	4th TP Y data	The 4th touch panel Y-axis sample data

Table 13-63: AICTPDR3 bit functions

Sound buffer registers (AICSDR0 - AICSDR3)

The AIC has four word-length, read-only registers for 8-bit sound input data from a microphone. Each register can store four sound samples. The last byte of the last buffer register is written into, the interrupt INTAIC is generated and the CPU reads all four buffer registers (burst read).

Bit	Initial	Name	Function
31–24	All 0	SD7–SD0	4th sound output
23–16	All 0	SD7–SD0	3rd sound output
15–8	All 0	SD7–SD0	2nd sound output
7–0	All 0	SD7–SD0	1st sound output

Table 13-64: AICSDR0 bit functions

Bit	Initial	Name	Function
31–24	All 0	SD7–SD0	8th sound output
23–16	All 0	SD7–SD0	7th sound output
15–8	All 0	SD7–SD0	6th sound output
7–0	All 0	SD7–SD0	5th sound output

Table 13-65: AICSDR1 bit functions

Bit	Initial	Name	Function
31–24	All 0	SD7–SD0	12th sound output
23–16	All 0	SD7–SD0	11th sound output
15–8	All 0	SD7–SD0	10th sound output
7–0	All 0	SD7–SD0	9th sound output

Table 13-66: AICSDR2 bit functions

Bit	Initial	Name	Function
31–24	All 0	SD7–SD0	16th sound output
23–16	All 0	SD7–SD0	15th sound output
15–8	All 0	SD7–SD0	14th sound output
7–0	All 0	SD7–SD0	13th sound output

Table 13-67: AICSDR3 bit functions

Main/backup battery data register (AICBDR)

This 16-bit register stores the levels of the main and backup battery.

Bit	Initial	Name	Function
15–8	All 0	MBD7–MBD0	Main battery check data
7–0	All 0	BBD7–BBD0	Backup battery check data

Table 13-68: AICBDR bit functions

Test mode control registers (AICTSTCR)

This 8-bit register controls the AIC unit test mode.

Bit	Initial	Name	Function
7	0	TSTCR7	Test enable 0 - normal mode 1 - test mode
6	0	TSTCR6	Test mode bit 0 - AIC test mode 1 - ADC test mode
5	0	TSTCR5	TicClkEn 0 - AIC input clk - PCLK 1 - AIC input clk - TCLK
4	0	TSTCR4	OclkSelect. This bit is used only to test the AIC block: 0 - The clock (OCLK) generated inside AIC is used. OCLK is slow. 1 - PCLK is used instead of OCLK to increase the speed in test mode.
3	0	TSTCR3	TcountMode This bit is used to reduce the touch panel interrupt interval, and is used only in test mode. 0 - normal interrupt interval 1 - fixed interrupt interval, regardless of sampling rate
2	0	TSTCR2	Test bit 0 - normal output to ADC 1 - test output to ADC
1–0	-	-	Reserved

Table 13-69: AICTSTCR bit functions

Test registers (AICTSTR0–AICTSTR4, ADCTSTR0–ADCTSTR4)

Ten 8-bit registers are used to save data for the AIC unit test and ADC unit test. The functions of each of the registers are described below. The AICTSTR0, AICTSTR1, ADCTSTR0 and ADCTSTR1 registers can be read and written, but the others are read-only registers.

Bit	Initial	Name	Function
7	X	Clock	Bit is to read outgoing ACLK signals to ADC. In normal mode, PCLK is bypassed to ACLK, but in test mode ACLK has the value of TCLK. This register is readable in both normal mode and test mode.
6–0	-	-	Reserved

Table 13-70: AICTSTR0 bit functions

Bit	Initial	Name	Function
7–0	all 0	TSTD7–0	Test register to provide 8-bit data (AD) instead of the ADC output in AIC test mode.

Table 13-71: AICTSTR1 bit functions

Bit	Initial	Name	Function
7	0	TSTD7	Bit to read outgoing INTAIC signal value to ADC
6–4	-	-	Reserved
3	1	TDTD3	Bit to read outgoing AASN data to ADC
2	1	TSTD2	Bit to read outgoing ARDn data to ADC
1	1	TSTD1	Bit to read outgoing AWRn data to ADC
0	1	TSTD0	Bit to read outgoing AIOSTOP data to ADC

Table 13-72: AICTSTR2 bit functions

Bit	Initial	Name	Function
7	0	TSTD7	Bit to read outgoing SWXN signal to touch panel
6	0	TSTD6	Bit to read outgoing SWYN signal to touch panel
5	1	TSTD5	Bit to read outgoing SWXP signal to touch panel
4	1	TSTD4	Bit to read outgoing SWYP signal to touch panel
3–0	all 0	TSTD3–0	Bits to read outgoing AA[3:0] data to ADC

Table 13-73: AICTSTR3 bit functions

Bit	Initial	Name	Function
7–0	all 0	TSTD7–0	Bits to read outgoing AD[7:0] data to ADC

Table 13-74: AICTSTR4 bit functions

Bit	Initial	Name	Function
7–4	all 1	TSTD7–4	Bits to drive AA for testing ADC
3	1	TSTD3	Bit to drive AASN for testing ADC
2	1	TSTD2	Bit to drive ARDN for testing ADC
1	1	TSTD1	Bit to drive AWRN for testing ADC
0	1	TSTD0	Bit to drive AIOSTOP for testing ADC

Table 13-75: ADCTSTR0 bit functions

Bit	Initial	Name	Function
7–0	all 0	TSTD7–0	Bits to drive AD for testing ADC

Table 13-76: ADCTSTR1 bit functions

Bit	Initial	Name	Function
7–0	X	TSTD7–0	Bits to read AD data from the ADC in ADC test mode

Table 13-77: ADCTSTR2 bit functions

Bit	Initial	Name	Function
7	0	TSTD7	Bits to drive ATEST for testing ADC
6	0	TSTD6	Bits to drive CALTEST for testing ADC
5–0	-	-	Reserved

Table 13-78: ADCTSTR3 bit functions

Bit	Initial	Name	Function
15–13	X	TSTD15–13	Reserved
12–3	X	TSTD12–3	Bits to read DATA data from the ADC in ADC test mode
2	X	TSTD2	Bits to read OTR data from the ADC in ADC test mode

Table 13-79: ADCTSTR4 bit functions

Bit	Initial	Name	Function
1	X	TSTD1	Bits to read P0 data from the ADC in ADC test mode
0	X	TSTD0	Bits to read AIADN data from the ADC in ADC test mode

Table 13-79: ADCTSTR4 bit functions (Continued)

Power control register (AICPOWER)

Bit	Initial	Name	Function
7–1	X	PCR7–1	Reserved
0	0	PCR0	0 - power down mode (PCLK and TCLK are disabled) 1 - normal mode

Table 13-80: AICPOWER bit functions

TCLK

This register does not exist; it is used to generate TIC clk in test mode. When APB address (PA) is TCLK register address, and PSEL and PSTB are HIGH in test mode, TCLK is HIGH; otherwise TCLK is LOW.

14 Debug and Test Interface

Debug and Test Interface

14.1 Overview

The GMS30C7201 has built-in features which enable debug and test in a number of different contexts. Firstly, there are circuit structures to help with software development. Secondly, the device contains boundary scan cells for circuit board test. Finally, the device contains some special test modes which enable the generation production patterns for the device itself.

14.2 Software Development Debug and Test Interface

The ARM720T and Piccolo processors incorporated inside GMS30C7201 contain hardware extensions for advanced debugging features. These are intended to ease user development and debugging of application software, operating systems, and the hardware itself.

Full details of the debug interfaces and their programming can be found in *ARM720T Data Sheet* (ARM DDI-0087) and *Piccolo Data Sheet* (ARM DDI-0128). The MultiICE product enables the ARM720T and Piccolo macrocells to be debugged in one environment. Refer to *Guide to MultiICE* (ARM DUI-0048).

14.3 Test Access Port and Boundary-Scan

GMS30C7201 contains full boundary scan on its inputs and outputs to help with circuit board test. This supports both INTEST and EXTEST, allowing patterns to be applied serially to the GMS30C7201 when fixed in a board and for full circuit board connection respectively. The boundary-scan interface conforms to the IEEE Std. 1149.1- 1990, Standard Test Access Port and Boundary-Scan Architecture. (Please refer to this standard for an explanation of the terms used in this section and for a description of the TAP controller states.) The boundary-scan interface provides a means of testing the core of the device when it is fitted to a circuit board, and a means of driving and sampling all the external pins of the device irrespective of the core state. This latter function permits testing of both the device's electrical connections to the circuit board, and (in conjunction with other devices on the circuit board having a similar interface) testing the integrity of the circuit board connections between devices. The interface intercepts all external connections within the device, and each such "cell" is then connected together to form a serial register (the boundary scan register). The whole interface is controlled via 5 dedicated pins: **TDI**, **TMS**, **TCK**, **nTRST** and **TDO**. ***Figure 14-1: Test Access Port (TAP) Controller State Transitions*** shows the state transitions that occur in the TAP controller.

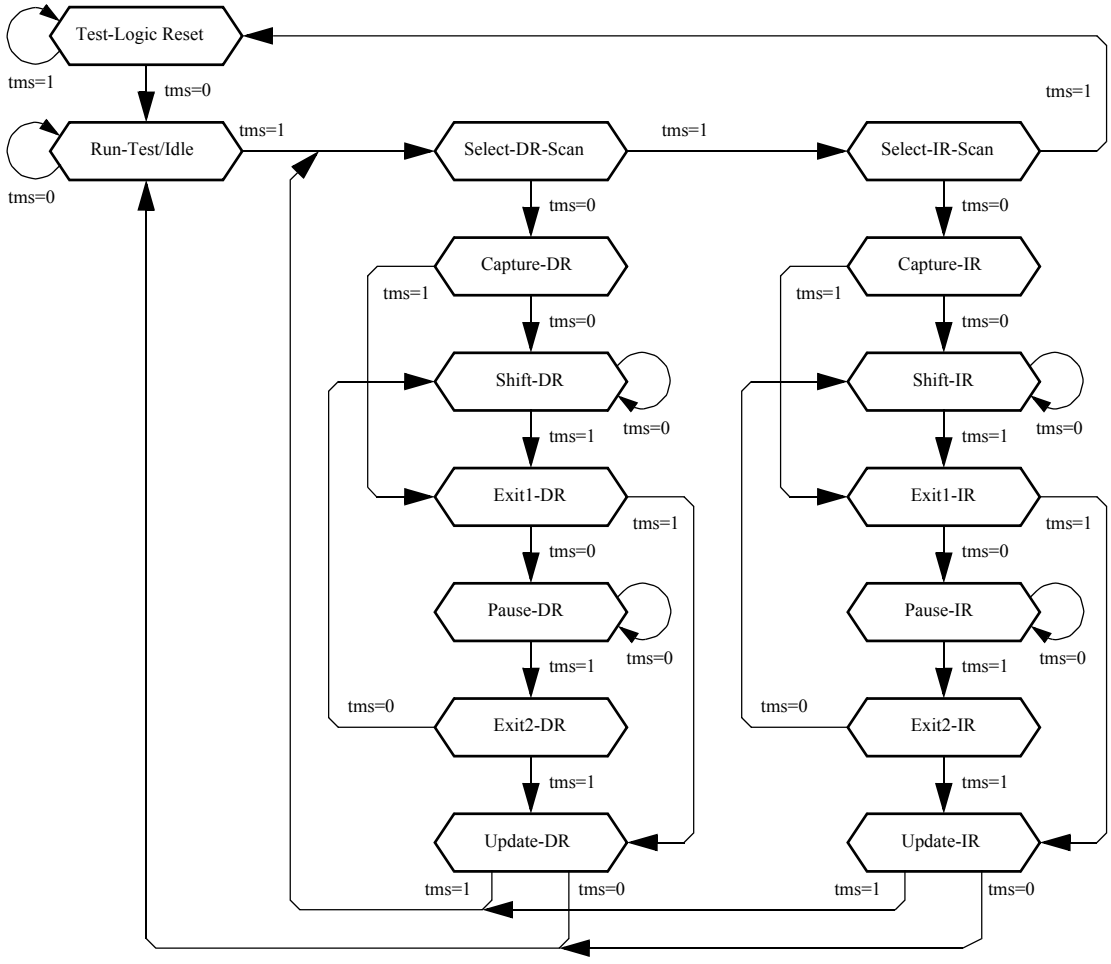


Figure 14-1: Test Access Port (TAP) Controller State Transitions

14.3.1 Reset

The boundary-scan interface includes a state-machine controller (the TAP controller). A pulldown resistor is included in the **nTRST** pad which holds the TAP controller state machine in a safe state after power up. In order to use the boundary scan interface, **nTRST** should be driven HIGH to take the TAP state machine out of reset.

The action of reset (either a pulse or a DC level) is as follows:

- System mode is selected (i.e. the boundary scan chain does NOT intercept any of the signals passing between the pads and the core).
- IDcode mode is selected. If **TCK** is pulsed, the contents of the ID register will be clocked out of **TDO**.

Note *The TAP controller inside GMS30C7201 contains a scan chip register which is reset to the value b0011 thus selecting the boundary scan chain. If this register is programmed to any value other than b0011, then it must be reprogrammed with b0011 or a reset applied before boundary scan operation can be attempted.*

14.3.2 Pullup Resistors

The IEEE 1149.1 standard requires pullup resistors in the input pins. However, to ensure safe operation an internal pulldown is present in the **nTRST** pin and therefore will have to be driven HIGH when using this interface.

Pin	Internal Resistor
TCLK	Pullup
nTRST	Pulldown
TMS	Pullup
TDI	Pullup

Table 14-1: Internal Resistors for Input Pins

14.3.3 Instruction Register

The instruction register is 4 bits in length.

There is no parity bit. The fixed value loaded into the instruction register during the CAPTURE-IR controller state is: 0001.

14.3.4 Public Instructions

The following public instructions are supported:

Instruction	Binary Code
EXTEST	0000
SAMPLE/PRELOAD	0011
CLAMP	0101
HIGHZ	0111

Table 14-2: Supported Public Instructions

Instruction	Binary Code
CLAMPZ	1001
INTEST	1100
IDCODE	1110
BYPASS	1111

Table 14-2: Supported Public Instructions

In the descriptions that follow, **TDI** and **TMS** are sampled on the rising edge of **TCK** and all output transitions on **TDO** occur as a result of the falling edge of **TCK**.

EXTEST (0000)

The BS (boundary-scan) register is placed in test mode by the EXTEST instruction.The EXTEST instruction connects the BS register between **TDI** and **TDO**.When the instruction register is loaded with the EXTEST instruction, all the boundary-scan cells are placed in their test mode of operation.

In the CAPTURE-DR state, inputs from the system pins and outputs from the boundary-scan output cells to the system pins are captured by the boundary-scan cells. In the SHIFT-DR state, the previously captured test data is shifted out of the BS register via the **TDO** pin, whilst new test data is shifted in via the **TDI** pin to the BS register parallel input latch. In the UPDATE-DR state, the new test data is transferred into the BS register parallel output latch. Note that this data is applied immediately to the system logic and system pins. The first EXTEST vector should be clocked into the boundary-scan register, using the SAMPLE/PRELOAD instruction, prior to selecting EXTEST to ensure that known data is applied to the system logic.

SAMPLE/PRELOAD (0011)

The BS (boundary-scan) register is placed in normal (system) mode by the SAMPLE/PRELOAD instruction.

The SAMPLE/PRELOAD instruction connects the BS register between **TDI** and **TDO**.

When the instruction register is loaded with the SAMPLE/PRELOAD instruction, all the boundary-scan cells are placed in their normal system mode of operation.

In the CAPTURE-DR state, a snapshot of the signals at the boundary-scan cells is taken on the rising edge of **TCK**. Normal system operation is unaffected. In the SHIFT-DR state, the sampled test data is shifted out of the BS register via the **TDO** pin, whilst new data is shifted in via the **TDI** pin to preload the BS register parallel input latch. In the UPDATE-DR state, the preloaded data is transferred into the BS register parallel output latch. Note that this data is not applied to the system logic or system pins while the SAMPLE/PRELOAD instruction is active. This instruction should be used to preload the boundary-scan register with known data prior to selecting the INTEST or EXTEST instructions.

CLAMP (0101)

The CLAMP instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**. When the CLAMP instruction is loaded into the instruction register, the state of all output signals is defined by the values previously loaded into the boundary-scan register. A guarding pattern should be pre-loaded into the boundary-scan register using the SAMPLE/PRELOAD instruction prior to selecting the CLAMP instruction. In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

HIGHZ (0111)

The HIGHZ instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**. When the HIGHZ instruction is loaded into the instruction register, all outputs are placed in an inactive drive state. In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

CLAMPZ (1001)

The CLAMPZ instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**.When the CLAMPZ instruction is loaded into the instruction register, all outputs are placed in an inactive drive state, but the data supplied to the disabled output drivers is derived from the boundary-scan cells. The purpose of this instruction is to ensure, during production testing, that each output driver can be disabled when its data input is either a 0 or a 1.A guarding pattern (specified for this device at the end of this section) should be pre-loaded into the boundary-scan register using the SAMPLE/PRELOAD instruction prior to selecting the CLAMPZ instruction. In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

INTEST (1100)

The BS (boundary-scan) register is placed in test mode by the INTEST instruction. The INTEST instruction connects the BS register between **TDI** and **TDO**. When the instruction register is loaded with the INTEST instruction, all the boundary-scan cells are placed in their test mode of operation. In the CAPTURE-DR state, the complement of the data supplied to the core logic from input boundary-scan cells is captured, while the true value of the data that is output from the core logic to output boundary- scan cells is captured. Note that CAPTURE-DR captures the complemented value of the input cells for testability reasons. In the SHIFT-DR state, the previously captured test data is shifted out of the BS register via the **TDO** pin, whilst new test data is shifted in via the **TDI** pin to the BS register parallel input latch. In the UPDATE-DR state, the new test data is transferred into the BS register parallel output latch. Note that this data is applied immediately to the system logic and system pins. The first INTEST vector should be clocked into the boundary-scan register, using the SAMPLE/PRELOAD instruction, prior to selecting INTEST to ensure that known data is applied to the system logic. Single-step operation is possible using the INTEST instruction.

IDCODE (1110)

The IDCODE instruction connects the device identification register (or ID register) between **TDI** and **TDO**. The ID register is a 32-bit register that allows the manufacturer, part number and version of a component to be determined through the TAP. The IDCODE returned will be that for the ARM720T core. When the instruction register is loaded with the IDCODE instruction, all the boundary-scan cells are placed in their normal (system) mode of operation. In the CAPTURE-DR state, the device identification code (specified at the end of this section) is captured by the ID register. In the SHIFT-DR state, the previously captured device identification code is shifted out of the ID register via the **TDO** pin, whilst data is shifted in via the **TDI** pin into the ID register. In the UPDATE-DR state, the ID register is unaffected.

BYPASS (1111)

The BYPASS instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**. When the BYPASS instruction is loaded into the instruction register, all the boundary-scan cells are placed in their normal (system) mode of operation. This instruction has no effect on the system pins. In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In

the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

14.3.5 Test Data Registers

Figure 14-2: Boundary Scan Block Diagram illustrates the structure of the boundary scan logic

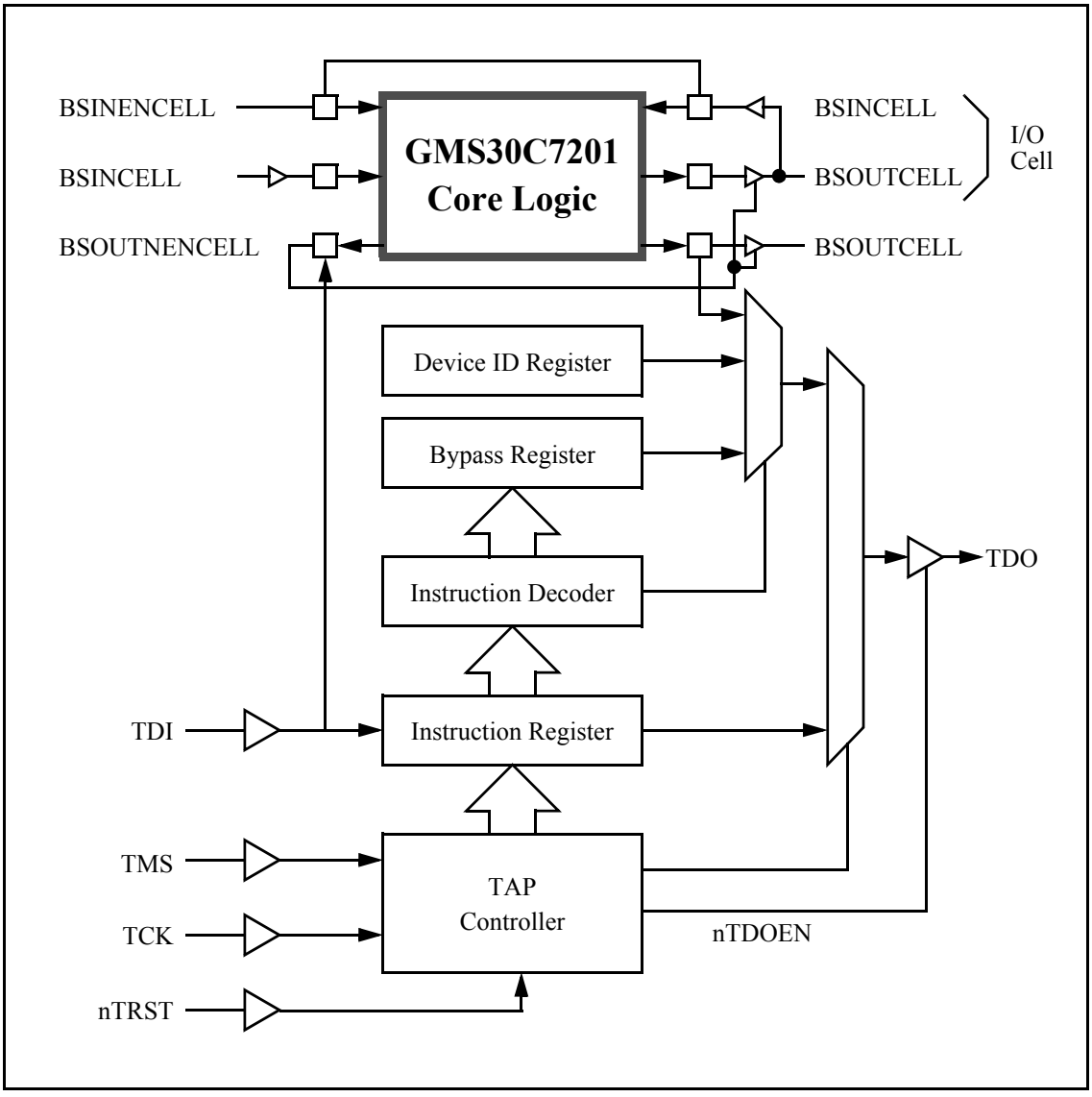


Figure 14-2: Boundary Scan Block Diagram

Bypass Register

Purpose: This is a single bit register which can be selected as the path between **TDI** and **TDO** to allow the device to be bypassed during boundary-scan testing.

Length: 1 bit

Operating Mode: When the BYPASS instruction is the current instruction in the instruction register, serial data is transferred from **TDI** to **TDO** in the SHIFT-DR state with a delay of one **TCK** cycle.

There is no parallel output from the bypass register.

A logic 0 is loaded from the parallel input of the bypass register in the CAPTURE-DR state.

Boundary Scan (BS) Register

Purpose: The BS register consists of a serially connected set of cells around the periphery of the device, at the interface between the core logic and the system input/output pads. This register can be used to isolate the core logic from the pins and then apply tests to the core logic, or conversely to isolate the pins from the core logic and then drive or monitor the system pins. Operating modes: The BS register is selected as the register to be connected between **TDI** and **TDO** only during the SAMPLE/PRELOAD, EXTEST and INTEST instructions. Values in the BS register are used, but are not changed, during the CLAMP and CLAMPZ instructions. In the normal (system) mode of operation, straight-through connections between the core logic and pins are maintained and normal system operation is unaffected. In TEST mode (i.e. when either EXTEST or INTEST is the currently selected instruction), values can be applied to the core logic or output pins independently of the actual values on the input pins and core logic outputs respectively. On the GMS30C7201 all of the boundary scan cells include an update register and thus all of the pins can be controlled in the above manner. Additional boundary-scan cells are interposed in the scan chain in order to control the enabling of tristateable buses. The values stored in the BS register after power-up are not defined. Similarly, the values previously clocked into the BS register are not guaranteed to be maintained across a Boundary Scan reset (from forcing **nTRST** LOW or entering the Test Logic Reset state).

Single-step Operation

GMS30C7201 is a static design and there is no minimum clock speed. It can therefore be single-stepped while the INTEST instruction is selected and the PLLs are bypassed. This can be achieved by serializing a parallel stimulus and clocking the resulting serial vectors into the boundary-scan register. When the boundary-scan register is updated, new test stimuli are applied to the core logic inputs; the effect of these stimuli can then be observed on the core logic outputs by capturing them in the boundary-scan register.

14.3.6 Boundary Scan Interface Signals

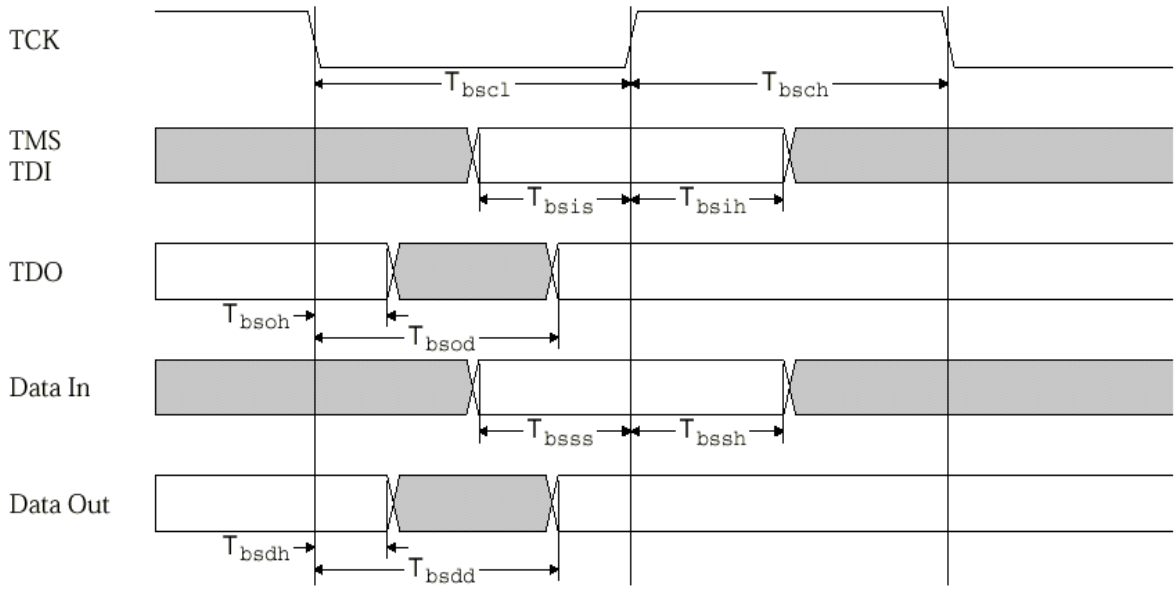


Figure 14-3: Boundary Scan General Timing

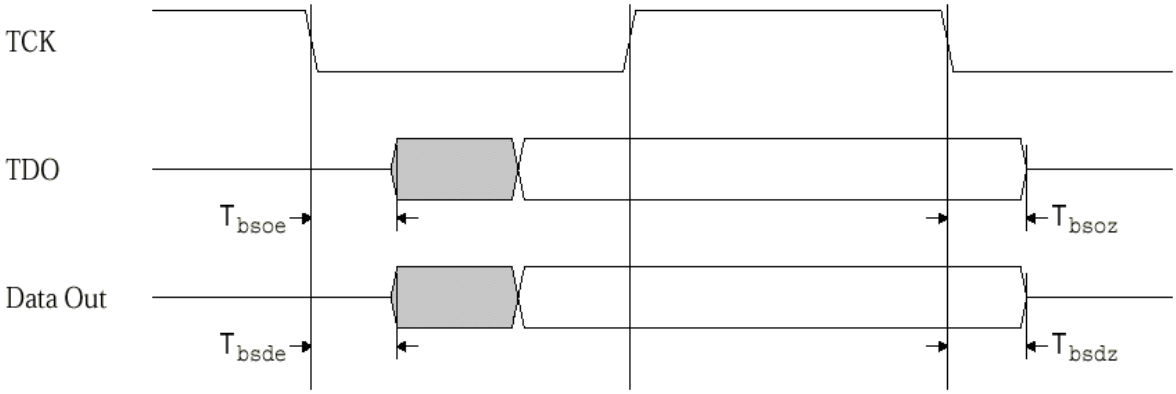


Figure 14-4: Boundary Scan Tristate Timing

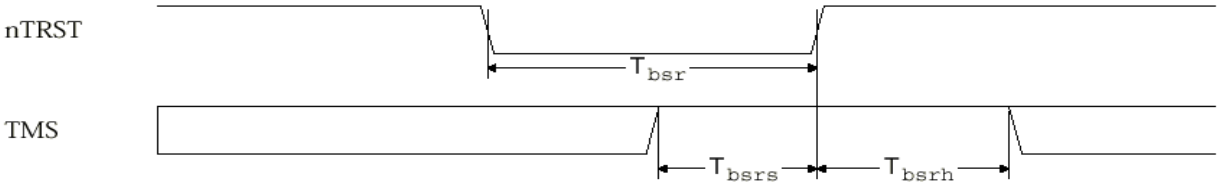


Figure 14-5: Boundary Scan Reset Timing

Symbol	Parameter	Min	Max
Tbscl	TCK low period	50	-
Tbsch	TCK high period	50	-
Tbsis	TMS, TDI setup to TCKr	0	-
Tbsih	TMS, TDI hold from TCKr	2	-
Tbsoh	TDO output hold from TCKf	3	-
Tbsod	TDO output delay from TCKf	-	20
Tbsss	Test mode Data in setup to TCKr	2	-
Tbssh	Test mode Data in hold from TCKf	5	-
Tbsdh	Test mode Data out hold from TCKf	3	-
Tbsdd	Test mode Data out delay from TCKf	-	20
Tbsoe	TDO output enable delay from TCKf	2	15
Tbsoz	Test mode Data enable delay from TCKf	2	15
Tbsde	TDO output disable delay from TCKf	2	15
Tbsdz	Test mode Data disable delay from TCKf	2	15
Tbsr	nTRST minimum pulse width	25	-
Tbsrs	TMS setup to nTRSTr	20	-
Tbsrh	TMS hold from nTRSTr	20	-

Table 14-3: Provisional Boundary scan AC parameters (units in ns)

The AC parameters are based on simulation results using 0.0pf circuit signal loads. Delays should be calculated using manufacturers output derating values for the actual circuit capacitance loading.

The correspondence between boundary-scan cells and system pins, system direction controls and system output enables is shown below. The cells are listed in the order in which they are connected in the boundary-scan register, starting with the cell closest to TDI. All outputs are three-state outputs. All boundary-scan register cells at input pins can apply tests to the on-chip system logic.

EXTEST/CLAMP guard values specified in the table below should be clocked into the boundary-scan register (using the SAMPLE/PRELOAD instruction) before the EXTEST, CLAMP or CLAMPZ instructions are selected to ensure that known data is applied to the system logic during the test. The INTEST guard values shown in the table below should be clocked into the boundary-scan register (using the SAMPLE/PRELOAD instruction) before the INTEST instruction is selected to ensure that all outputs are disabled. An asterisk in the guard value column indicates that any value can be submitted (as test requires), but ones and zeros should always be placed as shown.

Number	Cell Name	Pin	Type	Output Enable BS Cell	Guard Value	
					IN	EX
1	uOSCJTAG	OSCIN	IN	-	*	0
2	uLBLEN	LBLEN	OUT	-	0	*
3	uIRDIN	IRDIN	IN	-	*	0
4	uLD11	LD[11]	IN	-	*	*
5	uLD11	LD[11]	OUT	jnLDEn=0	*	*
6	jnLDEn	-	OUTEN0	-	1	*
7	uLLP	LLP	OUT	-	0	*
8	uLAC	LAC	OUT	-	0	*
9	uLCDEN	LCDEN	OUT	-	0	*
10	uLD7	LD[7]	IN	-	*	*
11	uLD7	LD[7]	OUT	jnLDEn=0	*	*
12	uLFP	LFP	OUT	-	0	*
13	uLCP	LCP	OUT	-	0	*
14	uLD9	LD[9]	IN	-	*	*
15	uLD9	LD[9]	OUT	jnLDEn=0	*	*
16	uLD3	LD[3]	IN	-	*	*
17	uLD3	LD[3]	OUT	jnLDEn=0	*	*
18	uLD2	LD[2]	IN	-	*	*
19	uLD2	LD[2]	OUT	jnLDEn=0	*	*
20	uLD10	LD[10]	IN	-	*	*
21	uLD10	LD[10]	OUT	jnLDEn=0	*	*
22	uLD8	LD[8]	IN	-	*	*
23	uLD8	LD[8]	OUT	jnLDEn=0	*	*
24	uLD5	LD[5]	IN	-	*	*
25	uLD5	LD[5]	OUT	jnLDEn=0	*	*
26	uVGAHS	VGAHS	OUT	-	0	*
27	uLD6	LD[6]	IN	-	*	*
28	uLD6	LD[6]	OUT	jnLDEn=0	*	*

Table 14-4: Boundary Scan Signals and Pins

29	uLD4	LD[4]	IN	-	*	*
30	uLD4	LD[4]	OUT	jnLDEn=0	*	*
31	uLD0	LD[0]	IN	-	*	*
32	uLD0	LD[0]	OUT	jnLDEn=0	*	*
33	uLD1	LD[1]	IN	-	*	*
34	uLD1	LD[1]	OUT	jnLDEn=0	*	*
35	uVGAVS	VGAVS	OUT	-	0	*
36	uSSDIN	SSDIN	IN	-	*	0
37	uSSOUT	SSOUT	OUT	-	0	*
38	unSSCS	nSSCS	OUT	-	1	*
39	uPORTD5	PORTD[5]	IN	-	*	0
40	uPORTD5	PORTD[5]	OUT	jMuxnPORTDEn[5]=0	0	*
41	jMuxnPORTDEn[5]	-	OUTEN0	-	1	*
42	uSSCLK	SSCLK	OUT	-	0	*
43	uATSYP	ATSYP	OUT	-	0	*
44	uATSYM	ATSYM	OUT	-	0	*
45	uATSXP	ATSXP	OUT	-	0	*
46	uMCLK	MCLK	IN	-	*	0
47	uMDOUT	MDOUT	OUT	-	0	*
48	uMRLY	MRLY	OUT	-	0	*
49	uATSXM	ATSXM	OUT	-	0	*
50	uMDFR	MDFR	IN	-	*	0
51	uMRING	MRING	IN	-	*	0
52	uMDIN	MDIN	IN	-	*	0
53	uRTOSCJTAG	RTCOSCIN	IN	-	*	0
54	uUSOUT0	USOUT[0]	OUT	-	0	*
55	unMCON	nMCON	OUT	-	1	*
56	uUSIN0	USIN[0]	IN	-	*	0
57	unUDTR1	nUDTR[1]	OUT	-	1	*
58	unUCTS1	nUCTS[1]	IN	-	*	0
59	unUCTS0	nUCTS[0]	IN	-	*	0
60	unURTS0	nURTS[0]	OUT	-	1	*

Table 14-4: Boundary Scan Signals and Pins

61	uUSIN1	USIN[1]	IN	-	*	0
62	unURTS1	nURTS[1]	OUT	-	1	*
63	uUSOUT1	USOUT[1]	OUT	-	0	*
64	unUDSR0	nUDSR[0]	IN	-	*	0
65	unUDTR0	nUDTR[0]	OUT	-	1	*
66	unUDCD1	nUDCD[1]	IN	-	*	0
67	uKSCANO10	KSCANO[10]	OUT	-	0	*
68	unUDCD0	nUDCD[0]	IN	-	*	0
69	unUDSR1	nUDSR[1]	IN	-	*	0
70	uKSCANO8	KSCANO[8]	OUT	-	0	*
71	uKSCANO5	KSCANO[5]	OUT	-	0	*
72	uKSCANO6	KSCANO[6]	OUT	-	0	*
73	uKSCANO7	KSCANO[7]	OUT	-	0	*
74	uKSCANO9	KSCANO[9]	OUT	-	0	*
75	uKSCANO3	KSCANO[3]	OUT	-	0	*
76	uKSCANO1	KSCANO[1]	OUT	-	0	*
77	uKSCANO2	KSCANO[2]	OUT	-	0	*
78	uKSCANO4	KSCANO[4]	OUT	-	0	*
79	uKSCANI7	KSCANI[7]	IN	-	*	0
80	uKSCANI6	KSCANI[6]	IN	-	*	0
81	uKSCANI4	KSCANI[4]	IN	-	*	0
82	uKSCANO0	KSCANO[0]	OUT	-	0	*
83	uKSCANI5	KSCANI[5]	IN	-	*	0
84	uKSCANI2	KSCANI[2]	IN	-	*	0
85	uKSCANI1	KSCANI[1]	IN	-	*	0
86	uKSCANI0	KSCANI[0]	IN	-	*	0
87	uKSCANI3	KSCANI[3]	IN	-	*	0
88	unRCS4	nRCS[4]	IN	-	*	0
89	unRCS4	nRCS[4]	OUT	jMuxnPORTDEn[2]=0	1	*
90	jMuxnPORTDEn[2]	-	OUTEN0	-	1	*
91	unRCS3	nRCS[3]	IN	-	*	0
92	unRCS3	nRCS[3]	OUT	jMuxnPORTDEn[1]=0	1	*

Table 14-4: Boundary Scan Signals and Pins

93	jMuxnPORTDEn[1]	-	OUTEN0	-	1	*
94	unRCS5	nRCS[5]	IN	-	*	0
95	unRCS5	nRCS[5]	OUT	jMuxnPORTDEn[3]=0	1	*
96	jMuxnPORTDEn[3]	-	OUTEN0	-	1	*
97	uEXPRDY	EXPRDY	IN	-	*	0
98	uBOOTBIT0	BOOTBIT[0]	IN	-	*	0
99	uRA25	RA[25]	OUT	-	*	*
100	uEXBCLK	EXBCLK	IN	-	*	0
101	uEXBCLK	EXBCLK	OUT	jMuxnPORTDEn[0]=0	0	*
102	jMuxnPORTDEn[0]	-	OUTEN0	-	1	*
103	uBOOTBIT1	BOOTBIT[1]	IN	-	*	0
104	uRA22	RA[22]	OUT	-	*	*
105	uRA23	RA[23]	OUT	-	*	*
106	uRA17	RA[17]	OUT	-	*	*
107	uRA24	RA[24]	OUT	-	*	*
108	uRA19	RA[19]	OUT	-	*	*
109	uPORTD4	PORTD[4]	IN	-	*	0
110	uPORTD4	PORTD[4]	OUT	jMuxnPORTDEn[4]=0	0	*
111	jMuxnPORTDEn[4]	-	OUTEN0	-	1	*
112	uRA20	RA[20]	OUT	-	*	*
113	uRA12	RA[12]	OUT	-	*	*
114	uRA21	RA[21]	OUT	-	*	*
115	uRA15	RA[15]	OUT	-	*	*
116	uRA13	RA[13]	OUT	-	*	*
117	uRA16	RA[16]	OUT	-	*	*
118	uRA18	RA[18]	OUT	-	*	*
119	uRA8	RA[8]	OUT	-	*	*
120	uRA10	RA[10]	OUT	-	*	*
121	uRA11	RA[11]	OUT	-	*	*
122	uRA14	RA[14]	OUT	-	*	*
123	uRA3	RA[3]	OUT	-	*	*
124	uRA6	RA[6]	OUT	-	*	*

Table 14-4: Boundary Scan Signals and Pins

125	uRA7	RA[7]	OUT	-	*	*
126	uRA9	RA[9]	OUT	-	*	*
127	uRA4	RA[4]	OUT	-	*	*
128	unROE	nROE	OUT	-	1	*
129	uRA1	RA[1]	OUT	-	*	*
130	uRA2	RA[2]	OUT	-	*	*
131	uRA5	RA[5]	OUT	-	*	*
132	unRWE3	nRWE[3]	OUT	-	1	*
133	unRWE2	nRWE[2]	OUT	-	1	*
134	unRCS1	nRCS[1]	OUT	-	1	*
135	unRCS2	nRCS[2]	OUT	-	1	*
136	uRA0	RA[0]	OUT	-	*	*
137	unRWE0	nRWE[0]	OUT	-	1	*
138	unRWE1	nRWE[1]	OUT	-	1	*
139	unRCS0	nRCS[0]	OUT	-	1	*
140	uRD0	RD[0]	IN	-	*	*
141	uRD0	RD[0]	OUT	jnRDEn[0]=0	*	*
142	uRD1	RD[1]	IN	-	*	*
143	uRD1	RD[1]	OUT	jnRDEn[0]=0	*	*
144	uRD2	RD[2]	IN	-	*	*
145	uRD2	RD[2]	OUT	jnRDEn[0]=0	*	*
146	uRD6	RD[6]	IN	-	*	*
147	uRD6	RD[6]	OUT	jnRDEn[0]=0	*	*
148	uRD4	RD[4]	IN	-	*	*
149	uRD4	RD[4]	OUT	jnRDEn[0]=0	*	*
150	uRD7	RD[7]	IN	-	*	*
151	uRD7	RD[7]	OUT	-	*	*
152	jnRDEn[0]	-	OUTEN0	-	1	*
153	uRD10	RD[10]	IN	-	*	*
154	uRD10	RD[10]	OUT	jnRDEn[1]=0	*	*
155	uRD11	RD[11]	IN	-	*	*
156	uRD11	RD[11]	OUT	jnRDEn[1]=0	*	*

Table 14-4: Boundary Scan Signals and Pins

157	uRD3	RD[3]	IN	-	*	*
158	uRD3	RD[3]	OUT	jnRDEn[0]=0	*	*
159	uRD5	RD[5]	IN	-	*	*
160	uRD5	RD[5]	OUT	jnRDEn[0]=0	*	*
161	uRD9	RD[9]	IN	-	*	*
162	uRD9	RD[9]	OUT	jnRDEn[1]=0	*	*
163	uRD15	RD[15]	IN	-	*	*
164	uRD15	RD[15]	OUT	-	*	*
165	jnRDEn[1]	-	OUTEN0	-	1	*
166	uRD16	RD[16]	IN	-	*	*
167	uRD16	RD[16]	OUT	jnRDEn[2]=0	*	*
168	uRD8	RD[8]	IN	-	*	*
169	uRD8	RD[8]	OUT	jnRDEn[1]=0	*	*
170	uRD13	RD[13]	IN	-	*	*
171	uRD13	RD[13]	OUT	jnRDEn[1]=0	*	*
172	uRD19	RD[19]	IN	-	*	*
173	uRD19	RD[19]	OUT	jnRDEn[2]=0	*	*
174	uRD12	RD[12]	IN	-	*	*
175	uRD12	RD[12]	OUT	jnRDEn[1]=0	*	*
176	uRD20	RD[20]	IN	-	*	*
177	uRD20	RD[20]	OUT	jnRDEn[2]=0	*	*
178	uRD18	RD[18]	IN	-	*	*
179	uRD18	RD[18]	OUT	jnRDEn[2]=0	*	*
180	uRD14	RD[14]	IN	-	*	*
181	uRD14	RD[14]	OUT	jnRDEn[1]=0	*	*
182	uRD24	RD[24]	IN	-	*	*
183	uRD24	RD[24]	OUT	jnRDEn[3]=0	*	*
184	uRD17	RD[17]	IN	-	*	*
185	uRD17	RD[17]	OUT	jnRDEn[2]=0	*	*
186	uRD25	RD[25]	IN	-	*	*
187	uRD25	RD[25]	OUT	jnRDEn[3]=0	*	*
188	uRD22	RD[22]	IN	-	*	*

Table 14-4: Boundary Scan Signals and Pins

189	uRD22	RD[22]	OUT	jnRDEn[2]=0	*	*
190	uRD28	RD[28]	IN	-	*	*
191	uRD28	RD[28]	OUT	jnRDEn[3]=0	*	*
192	uRD21	RD[21]	IN	-	*	*
193	uRD21	RD[21]	OUT	jnRDEn[2]=0	*	*
194	uRD29	RD[29]	IN	-	*	*
195	uRD29	RD[29]	OUT	jnRDEn[3]=0	*	*
196	uRD31	RD[31]	IN	-	*	*
197	uRD31	RD[31]	OUT	-	*	*
198	jnRDEn[3]	-	OUTEN0	-	1	*
199	uRD27	RD[27]	IN	-	*	*
200	uRD27	RD[27]	OUT	jnRDEn[3]=0	*	*
201	uRD26	RD[26]	IN	-	*	*
202	uRD26	RD[26]	OUT	jnRDEn[3]=0	*	*
203	uRD23	RD[23]	IN	-	*	*
204	uRD23	RD[23]	OUT	-	*	*
205	jnRDEn[2]	-	OUTEN0	-	1	*
206	uPCBVCCEN0	PCBVCCEN[0]	OUT	-	0	*
207	uPCBVCCEN1	PCBVCCEN[1]	OUT	-	0	*
208	uPCBVPPEN0	PCBVPPEN[0]	OUT	-	0	*
209	uRD30	RD[30]	IN	-	*	*
210	uRD30	RD[30]	OUT	jnRDEn[3]=0	*	*
211	uPCAVCCEN0	PCAVCCEN[0]	OUT	-	0	*
212	uPCAVCCEN1	PCAVCCEN[1]	OUT	-	0	*
213	uPCBVPPEN1	PCBVPPEN[1]	OUT	-	0	*
214	uPCBVS0	PCBVS[0]	IN	-	*	0
215	uPCBVS1	PCBVS[1]	IN	-	*	0
216	uPCAVS0	PCAVS[0]	IN	-	*	0
217	uPCAVPPEN1	PCAVPPEN[1]	OUT	-	0	*
218	uPCAVS1	PCAVS[1]	IN	-	*	0
219	unPCBCD0	nPCBCD[0]	IN	-	*	1
220	uPCAVPPEN0	PCAVPPEN[0]	OUT	-	0	*

Table 14-4: Boundary Scan Signals and Pins

221	unPCBOE	nPCBOE	OUT	-	1	*
222	unPCBCD1	nPCBCD[1]	IN	-	*	1
223	unPCACD0	nPCACD[0]	IN	-	*	1
224	uPORTD7	PORTD[7]	IN	-	*	0
225	uPORTD7	PORTD[7]	OUT	jnPORTDEn[7]=0	0	*
226	jnPORTDEn[7]	-	OUTEN0	-	1	*
227	uPCBWP	PCBWP	IN	-	*	0
228	unPCAOE	nPCAOE	OUT	-	1	*
229	unPCACD1	nPCACD[1]	IN	-	*	1
230	uPCCBDRV	PCCBDRV	OUT	-	0	*
231	unPCBWE	nPCBWE	OUT	-	1	*
232	uPCAWP	PCAWP	IN	-	*	0
233	uPCBREADY	PCBREADY	IN	-	*	0
234	uPCBBVD0	PCBBVD[0]	IN	-	*	0
235	uPCCADRV	PCCADRV	OUT	-	0	*
236	uPCBBVD1	PCBBVD[1]	IN	-	*	0
237	uPCAREADY	PCAREADY	IN	-	*	0
238	uPCABVD0	PCABVD[0]	IN	-	*	0
239	uPCBRESET	PCBRESET	OUT	-	0	*
240	uPCABVD1	PCABVD[1]	IN	-	*	0
241	unPCBWAIT	nPCBWAIT	IN	-	*	1
242	unPCAWE	nPCAWE	OUT	-	1	*
243	unPCBIORD	nPCBIORD	OUT	-	1	*
244	uPCARESET	PCARESET	OUT	-	0	*
245	unPCBIOWR	nPCBIOWR	OUT	-	1	*
246	unPCAWAIT	nPCAWAIT	IN	-	*	1
247	uPCBIPORTE	PCBIPORTE	OUT	-	0	*
248	unPCAIORD	nPCAIORD	OUT	-	1	*
249	unPCBCE0	nPCBCE[0]	OUT	-	1	*
250	unPCAIOWR	nPCAIOWR	OUT	-	1	*
251	unPCBCE1	nPCBCE[1]	OUT	-	1	*
252	unPCACE0	nPCACE[0]	OUT	-	1	*

Table 14-4: Boundary Scan Signals and Pins

253	unPCREG	nPCREG	OUT	-	1	*
254	unPCACE1	nPCACE[1]	OUT	-	1	*
255	uUSUSPEND	USUSPEND	OUT	-	0	*
256	uUSBOE	nUSBOE	IN	-	*	1
257	uUSBOE	nUSBOE	OUT	jMuxnPORTBEn[2]=0	1	*
258	jMuxnPORTBEn[2]	-	OUTEN0	-	1	*
259	uUVP	UVP	IN	-	*	0
260	uUVP	UVP	OUT	jMuxnPORTBEn[4]=0	0	*
261	jMuxnPORTBEn[4]	-	OUTEN0	-	1	*
262	uUVMO	UVMO	IN	-	*	0
263	uUVMO	UVMO	OUT	jMuxnPORTBEn[1]=0	0	*
264	jMuxnPORTBEn[1]	-	OUTEN0	-	1	*
265	uPORTC7	PORTC[7]	IN	-	*	0
266	uPORTC7	PORTC[7]	OUT	jnPORTCEn[7]=0	0	*
267	jnPORTCEn[7]	-	OUTEN0	-	1	*
268	uURCVIN	URCVIN	IN	-	*	0
269	uURCVIN	URCVIN	OUT	jMuxnPORTBEn[3]=0	0	*
270	jMuxnPORTBEn[3]	-	OUTEN0	-	1	*
271	uUVPO	UVPO	IN	-	*	0
272	uUVPO	UVPO	OUT	jMuxnPORTBEn[0]=0	0	*
273	jMuxnPORTBEn[0]	-	OUTEN0	-	1	*
274	uPORTA7	PORTA[7]	IN	-	*	0
275	uPORTA7	PORTA[7]	OUT	jnPORTAEn[7]=0	0	*
276	jnPORTAEn[7]	-	OUTEN0	-	1	*
277	uUVM	UVM	IN	-	*	0
278	uUVM	UVM	OUT	jMuxnPORTBEn[5]=0	0	*
279	jMuxnPORTBEn[5]	-	OUTEN0	-	1	*
280	uPORTB6	PORTB[6]	IN	-	*	0
281	uPORTB6	PORTB[6]	OUT	jnPORTBEn[6]=0	0	*
282	jnPORTBEn[6]	-	OUTEN0	-	1	*
283	uPORTC6	PORTC[6]	IN	-	*	0
284	uPORTC6	PORTC[6]	OUT	jnPORTCEn[6]=0	0	*

Table 14-4: Boundary Scan Signals and Pins

285	jnPORTCEn[6]	-	OUTEN0	-	1	*
286	uPORTA3	PORTA[3]	IN	-	*	0
287	uPORTA3	PORTA[3]	OUT	jnPORTAEn[3]=0	0	*
288	jnPORTAEn[3]	-	OUTEN0	-	1	*
289	uPORTA6	PORTA[6]	IN	-	*	0
290	uPORTA6	PORTA[6]	OUT	jnPORTAEn[6]=0	0	*
291	jnPORTAEn[6]	-	OUTEN0	-	1	*
292	uPORTC3	PORTC[3]	IN	-	*	0
293	uPORTC3	PORTC[3]	OUT	jMuxnPORTCEn[3]=0	0	*
294	jMuxnPORTCEn[3]	-	OUTEN0	-	1	*
295	uPORTC5	PORTC[5]	IN	-	*	0
296	uPORTC5	PORTC[5]	OUT	jnPORTCEn[5]=0	0	*
297	jnPORTCEn[5]	-	OUTEN0	-	1	*
298	uPORTA2	PORTA[2]	IN	-	*	0
299	uPORTA2	PORTA[2]	OUT	jnPORTAEn[2]=0	0	*
300	jnPORTAEn[2]	-	OUTEN0	-	1	*
301	uPORTC2	PORTC[2]	IN	-	*	0
302	uPORTC2	PORTC[2]	OUT	jMuxnPORTCEn[2]=0	0	*
303	jMuxnPORTCEn[2]	-	OUTEN0	-	1	*
304	uPORTA5	PORTA[5]	IN	-	*	0
305	uPORTA5	PORTA[5]	OUT	jnPORTAEn[5]=0	0	*
306	jnPORTAEn[5]	-	OUTEN0	-	1	*
307	uPORTC4	PORTC[4]	IN	-	*	0
308	uPORTC4	PORTC[4]	OUT	jMuxnPORTCEn[4]=0	0	*
309	jMuxnPORTCEn[4]	-	OUTEN0	-	1	*
310	uPORTB7	PORTB[7]	IN	-	*	0
311	uPORTB7	PORTB[7]	OUT	jnPORTBEn[7]=0	0	*
312	jnPORTBEn[7]	-	OUTEN0	-	1	*
313	uPORTA1	PORTA[1]	IN	-	*	0
314	uPORTA1	PORTA[1]	OUT	jnPORTAEn[1]=0	0	*
315	jnPORTAEn[1]	-	OUTEN0	-	1	*
316	uPORTC1	PORTC[1]	IN	-	*	0

Table 14-4: Boundary Scan Signals and Pins

317	uPORTC1	PORTC[1]	OUT	jMuxnPORTCEn[1]=0	0	*
318	jMuxnPORTCEn[1]	-	OUTEN0	-	1	*
319	uPORTA4	PORTA[4]	IN	-	*	0
320	uPORTA4	PORTA[4]	OUT	jnPORTAEn[4]=0	0	*
321	jnPORTAEn[4]	-	OUTEN0	-	1	*
322	uSD0	SD[0]	IN	-	*	*
323	uSD0	SD[0]	OUT	jnSDEn=0	*	*
324	uPORTA0	PORTA[0]	IN	-	*	0
325	uPORTA0	PORTA[0]	OUT	jnPORTAEn[0]=0	0	*
326	jnPORTAEn[0]	-	OUTEN0	-	1	*
327	uPORTC0	PORTC[0]	IN	-	*	0
328	uPORTC0	PORTC[0]	OUT	jMuxnPORTCEn[0]=0	0	*
329	jMuxnPORTCEn[0]	-	OUTEN0	-	1	*
330	uSD1	SD[1]	IN	-	*	*
331	uSD1	SD[1]	OUT	jnSDEn=0	*	*
332	uSD14	SD[14]	IN	-	*	*
333	uSD14	SD[14]	OUT	jnSDEn=0	*	*
334	uSD2	SD[2]	IN	-	*	*
335	uSD2	SD[2]	OUT	jnSDEn=0	*	*
336	uSD15	SD[15]	IN	-	*	*
337	uSD15	SD[15]	OUT	-	*	*
338	jnSDEn	-	OUTEN0	-	1	*
339	uPORTD6	PORTD[6]	IN	-	*	0
340	uPORTD6	PORTD[6]	OUT	jMuxnPORTDEn[6]=0	0	*
341	jMuxnPORTDEn[6]	-	OUTEN0	-	1	*
342	uSD3	SD[3]	IN	-	*	*
343	uSD3	SD[3]	OUT	jnSDEn=0	*	*
344	uSD12	SD[12]	IN	-	*	*
345	uSD12	SD[12]	OUT	jnSDEn=0	*	*
346	uSD4	SD[4]	IN	-	*	*
347	uSD4	SD[4]	OUT	jnSDEn=0	*	*
348	uSD13	SD[13]	IN	-	*	*

Table 14-4: Boundary Scan Signals and Pins

349	uSD13	SD[13]	OUT	jnSDEn=0	*	*
350	uSD5	SD[5]	IN	-	*	*
351	uSD5	SD[5]	OUT	jnSDEn=0	*	*
352	uSD10	SD[10]	IN	-	*	*
353	uSD10	SD[10]	OUT	jnSDEn=0	*	*
354	uSD11	SD[11]	IN	-	*	*
355	uSD11	SD[11]	OUT	jnSDEn=0	*	*
356	uSD6	SD[6]	IN	-	*	*
357	uSD6	SD[6]	OUT	jnSDEn=0	*	*
358	uSD9	SD[9]	IN	-	*	*
359	uSD9	SD[9]	OUT	jnSDEn=0	*	*
360	uSD7	SD[7]	IN	-	*	*
361	uSD7	SD[7]	OUT	jnSDEn=0	*	*
362	uPMBATOK	PMBATOK	IN	-	*	0
363	uSD8	SD[8]	IN	-	*	*
364	uSD8	SD[8]	OUT	jnSDEn=0	*	*
365	unSWE	nSWE	OUT	-	1	*
366	uPMADAPOK	PMADAPOK	IN	-	*	0
367	unSCAS	nSCAS	OUT	-	1	*
368	uSDQML	SDQML	OUT	-	0	*
369	uSCKE2	SCKE[2]	OUT	-	0	*
370	unPMWAKEUP	nPMWAKEUP	IN	-	*	1
371	uSDQMU	SDQMU	OUT	-	0	*
372	uSCKE0	SCKE[0]	OUT	-	0	*
373	uSCKE3	SCKE[3]	OUT	-	0	*
374	unSCS2	nSCS[2]	OUT	-	1	*
375	uSCLK	SCLK	IN	-	*	0
376	uSCLK	SCLK	OUT	jnSCLKEn=0	0	*
377	jnSCLKEn	-	OUTEN0	-	1	*
378	uSCKE1	SCKE[1]	OUT	-	0	*
379	unSCS3	nSCS[3]	OUT	-	1	*
380	unSCS0	nSCS[0]	OUT	-	1	*

Table 14-4: Boundary Scan Signals and Pins

381	uSA9	SA[9]	OUT	-	*	*
382	unSCS1	nSCS[1]	OUT	-	1	*
383	uSA12	SA[12]	OUT	-	*	*
384	uSA11	SA[11]	OUT	-	*	*
385	uSA7	SA[7]	OUT	-	*	*
386	uSA13	SA[13]	OUT	-	*	*
387	uSA0	SA[0]	OUT	-	*	*
388	unSRAS	nSRAS	OUT	-	1	*
389	uSA8	SA[8]	OUT	-	*	*
390	uSA1	SA[1]	OUT	-	*	*
391	uSA10	SA[10]	OUT	-	*	*
392	uSA5	SA[5]	OUT	-	*	*
393	uIRDOUT	IRDOUT	OUT	-	0	*
394	unRESET	nRESET	IN	-	*	1
395	unRESET	nRESET	OUT	jnRESETEn=0	1	*
396	jnRESETEn	-	OUTEN0	-	1	*
397	uSA2	SA[2]	OUT	-	*	*
398	uSA6	SA[6]	OUT	-	*	*
399	uSA4	SA[4]	OUT	-	*	*
400	unPOR	nPOR	IN	-	*	1
401	unPLLENABLE	nPLLENABLE	IN	-	*	1
402	uSA3	SA[3]	OUT	-	*	*
403	unTEST	nTEST	IN	-	*	1

Table 14-4: Boundary Scan Signals and Pins

14.4 Production Test Features

In order to generate test vectors suitable for use on a production tester by the chip manufacturer, some special test modes have been introduced. These modes come into operation whenever the pin **nTEST** is forced LOW.

Full details of these modes are available from ARM in a special Test Document on request.

15 Electrical Characteristics

Electrical Characteristics

15.1 Absolute Maximum Ratings

Symbol	Parameter	Min	Typ	Max	Units
V_{DD}	Power Supply voltage	-0.3		5.5	V
V_I	DC Input voltage	-0.3		Vdd+0.5	V
V_O	DC Output voltage	-0.3		Vdd+0.5	V
T_{STG}	Storage temperature	-55		150	^oC
T_{JCT}	Junction Temperature	-55		150	^oC
P_D	Power Dissipation		277		mW
P_{DWN}	Power Dissipation at Power Down mode			132	uW

Table 15-1: Maximum Ratings

Note: Permanent damage can occur if maximum ratings are exceeded.
Device modules may not operate normally while being exposed to electrical extremes.
Although sections of the device contain circuitry to protect against damage from high static voltages or electical fields, take normal precautions to avoid exposure to voltages higher than maximum-rated voltage.

Recommended Operating Range

Symbol	Parameter	Min	Max	Units
V_{DD}	Power Supply voltage	3.0	3.6	V
T_{OPR}	Operating Temperature	-40	85	^oC

Table 15-2: Operating Range

15.2 DC Characteristics

All characteristics are specified at Vdd = 3.0 to 3.6 volts and Vss = 0 volts, over an operating temperature range of -40 to 85^{o}C

CMOS pins

Symbol	Parameter	Minimum	Maximum	Conditions
V_{IL}	Low Level input voltage	-0.5V	0.8V	Guaranteed Input Low Voltage
V_{IH}	High level input voltage	0.7 x V_{DD}	5.5V	Guaranteed Input High Voltage
V_{OL}	Low level output voltage		V_{SS} + 0.4V	IOL = 0.8 mA
V_{OH}	High level output voltage	V_{DD} - 0.4V		IOH = 0.8 mA
I_I	Input current at maximum voltage		1mA	Input = 5.5V

Table 15-3: CMOS signal pin characteristics

TTL pins

Symbol	Parameter	Minimum	Maximum	Conditions
V_{IL}	Low Level Input voltage	-0.5V	0.8V	
V_{IH}	High Level Input voltage	2.0 V	5.5V	
V_{OL}	Low Level output voltage		0.4V	IOL = 8mA depending on Cell
V_{OH}	High Level output voltage	2.4 V		IOH = 8mA depending on Cell
I_I	Input current at maximum voltage		1mA	Input = 5.5V

Table 15-4: TTL signal pin characteristics

Note : TTL pins are as follows and the drive capability of these is 8mA.
VGAHS, VGAVS, SD[15:0], SA[14:0], SCLK, SKE[3:0], nSRAS, nSCAS, nSW, nSCS[3:0], SDQMU, SDQML

I/O CIrcuit Pullups

The following current values are used for I/Os with internal pullup devices.

Symbol	Parameter	Minimum(at Pad = 0V)	Maximum(at Pad = 0V)
I_{PU}	3.3V Pullup	-30uA	-146uA
R_{ER}	Equivalent resistance	88.3kOhms	24.7kOhms

Note : The following pins are used with internal pullup devices.
nRESET, nPOR, nPMWAKEUP, TCK, TMS, TDI, nTEST, nPCACD[1:0], nPCBCD[1:0], PCAVS[1:0], PCBVS[1:0], nUDCD[1:0], nUDSR[1:0], nUCTS[1:0], USIN[1:0], KSCANI[7:0]

I/O CIrcuit Pulldowns

The following current values are used for I/Os with internal pulldown devices.

Symbol	Parameter	Minimum(at Pad = 2.6V)	Maximum(at Pad = 3.6V)
I_{PD}	Pulldown	31uA	159uA
R_{ER}	Equivalent resistance	85.5kOhms	22.6kOhms

Note : The following pins are used with internal pulldown devices.
nTRST, nPLLENABLE

15.3 A/D converter Electrical Characteristics

ADVcc = AAVcc = 3.3V, Fconv = 25Ksps unless otherwise specified

Item	Symbol	Test Condition	Min.	Typ.	Max.	Unit
Resolution				12		bit
Signal to Noise Distortion Ratio	SNR	Average 4 Fsinal=3.125KHz	50		65	dB
Analog Reference Voltage	AVref				AAVcc	V
Differential Non-linearity	DNL	10bit resolution	-0.5		0.5	LSB
Integral Non-linearity	INL	10bit resolution	-1.0		1.0	LSB
Junction Temperature	Tj		-20		130	^{o}C
Digital Power Supply	ADVcc		3	3.3	3.6	V
Analog Power Supply	AAVcc		3	3.3	3.6	V
Current Consumption	Idd	Operating			6.0	mA
	Idn	Power save			2.0	uA

Table 15-5: A/D characteristics

15.4 D/A Converter characteristics

15.4.1 Audio DAC

Item	Symbol	Test Condition	Min.	Typ.	Max.	Unit
Resolution				8		bit
Conversion speed	F_{CON}			50		KHz
D/A Output Voltage Range	V_{DAO}	Vref=3.3V	1.025		2.675	V
Differential Non-linearity	DNL	AC	-1.0		1.0	LSB
Integral Non-linearity	INL	AC	-1.5		1.5	LSB
Junction Temperature	Tj		-20		130	^oC
Digital Power Supply	ADVcc		3	3.3	3.6	V
Analog Power Supply	AAVcc		3	3.3	3.6	V
Comsumption Current	Idd	fclk=50KHz	2	3	5	mA

Table 15-6: Audio DAC characteristics

15.4.2 Video DAC

f

Item	Symbol	Test Condition	Min.	Typ.	Max.	Unit
Resolution				12		bit
Signal to Noise Distortion Ratio	SNR	Fcon=330KHz	40		50	dB
Differential Non-linearity	DNL	10bit resolution	-1.5		1.5	LSB
Integral Non-linearity	INL	10bit resolution	-1.5		1.5	LSB
Full scale current	Ifull	with +/-5% error		26.6		mA
rise/fall time	Tr/Tf	with +/-10%error			5	ns
Setup time	Ts		0.0		3.33	ns
Hold time	Th		0.4		0.5	ns
Output Delay time	Td				3	ns
Current Consumption	Idd				60	mA

Table 15-7: Video DAC characteristics

15.5 AC Characteristics

All characteristics are specified at Vdd = 3.0 to 3.6 volts and Vss = 0 volts over an operating temperature of -40 Deg C to +85 Deg C.

15.5.1 Expansion and ROM Interface

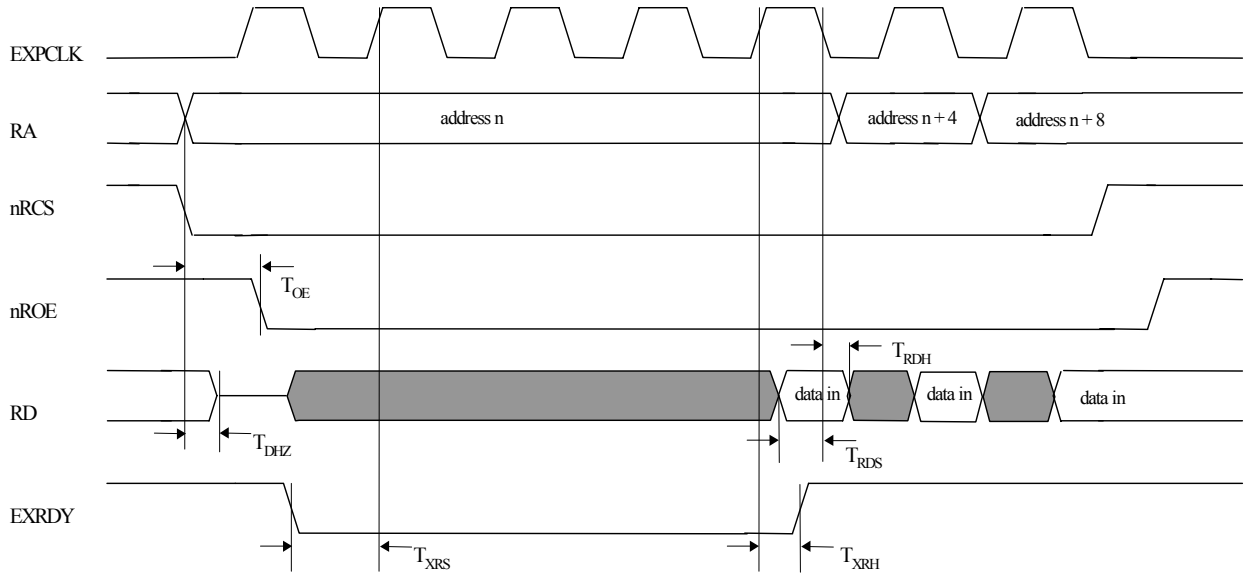


Figure 15-1: External Bus Read Timing

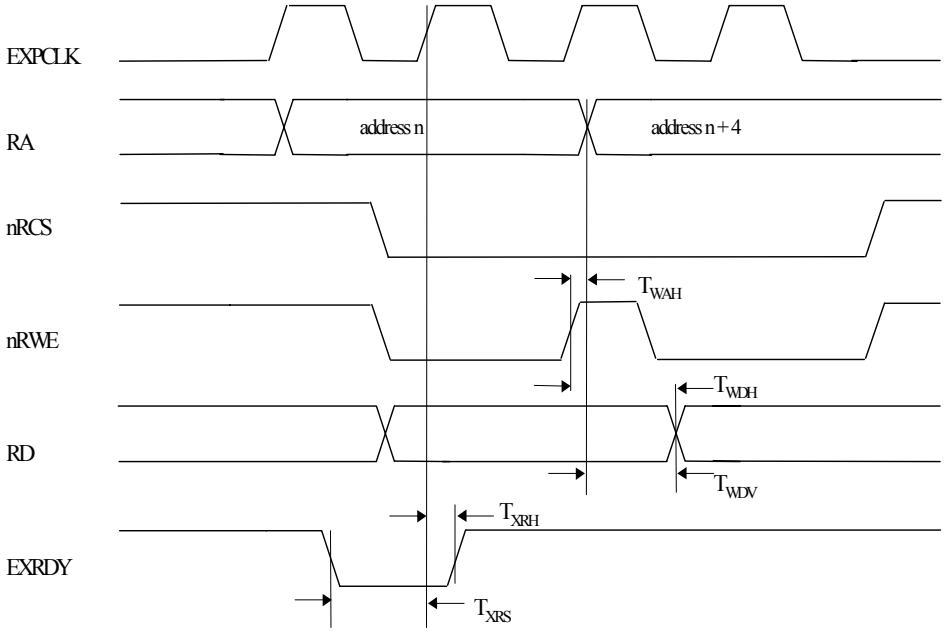


Figure 15-2: External Bus WriteTiming

Name	Description	Min	Max	Unit
TDHZ	Falling CS to data bus Hi-Z	0	15	ns
TRDH	Read data in to falling EXPCLK hold time	0		ns
TRDS	Read data in to falling EXPCLK setup time	16		ns
TWDH	Rising NNW to write data valid hold time	20		ns
TWAH	Rising owner to valid write address hold time	3		ns
TWDV	Address change to valid write data	18		ns
TOE	Falling CS to output enable Low	8	22	ns
TXRS	EXPRDY to rising EXPCLK setup time	10		ns
TXRH	Rising EXPCLK to EXPRDY hold time		22	ns

Table 15-8: External bus AC timing values

15.5.2 SDRAM interface signals

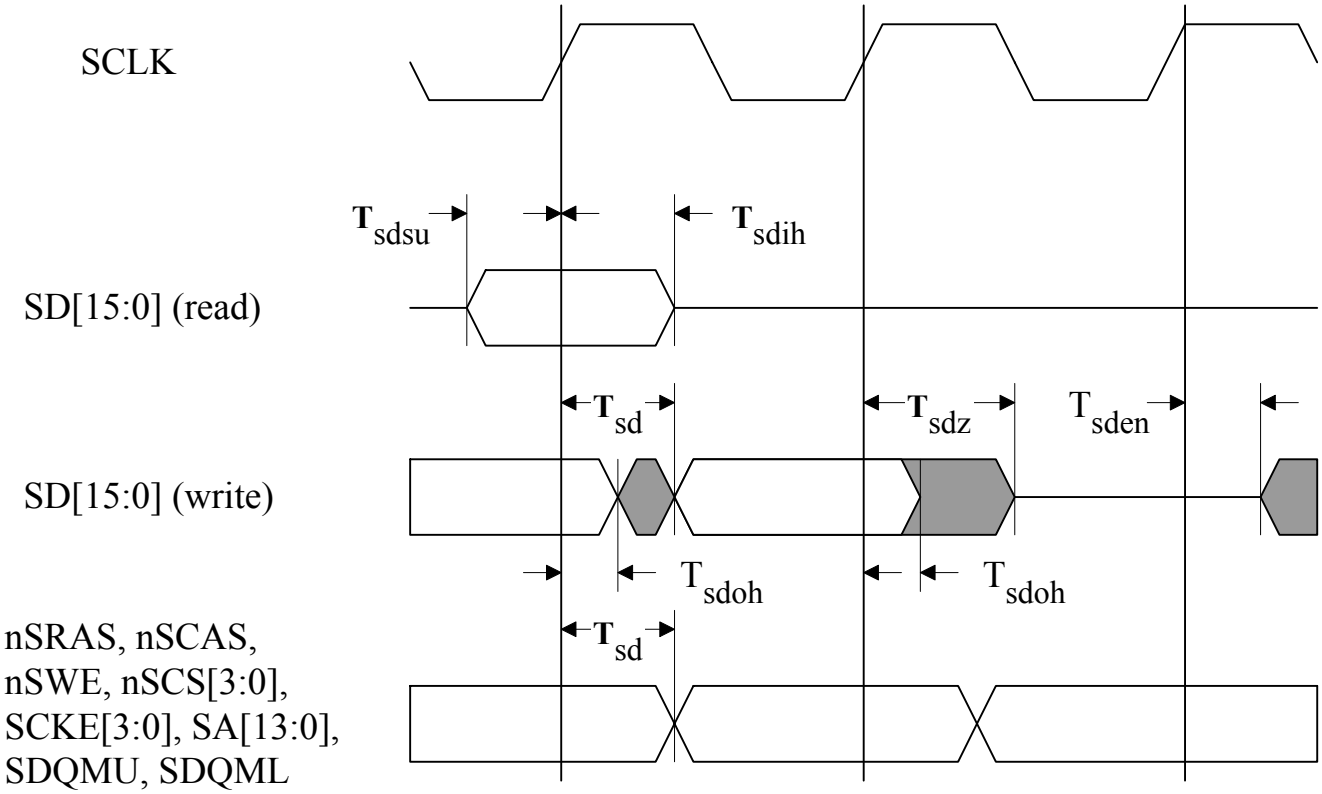


Figure 15-3: SDRAM interface timing

Symbol	Parameter	Min	Max
Tsdsu	SDRAM data in setup time to SCLKr	0	-
Tsdih	SDRAM data in hold from SCLKr	2	-
Tsd	Signal delay from SCLKr	2	7
Tsdoh	SDRAM data output hold time from SCLKr	2	-
Tsdz	SDRAM data bus disable time from SCLKr	2	11
Tsden	SDRAM data bus enable time from SCLKr	2	9

Table 15-9: Provisional SDRAM interface AC parameters (units ns)

Timing values are derived from simulations using 0pF signal loading. Actual circuit output delays should be calculated by adding manufacturers signal load de-rating delay values.

15.5.3 LCD interface, signal stimings

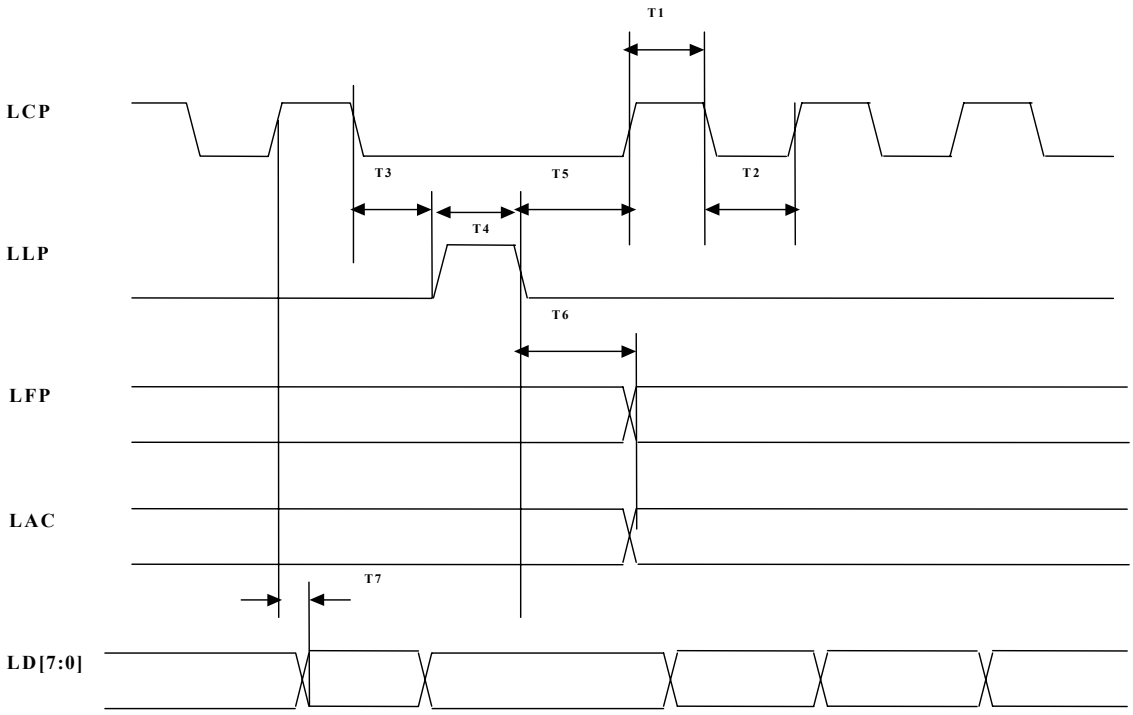


Figure 15-4: LCD Controller Timing (STN Mode)

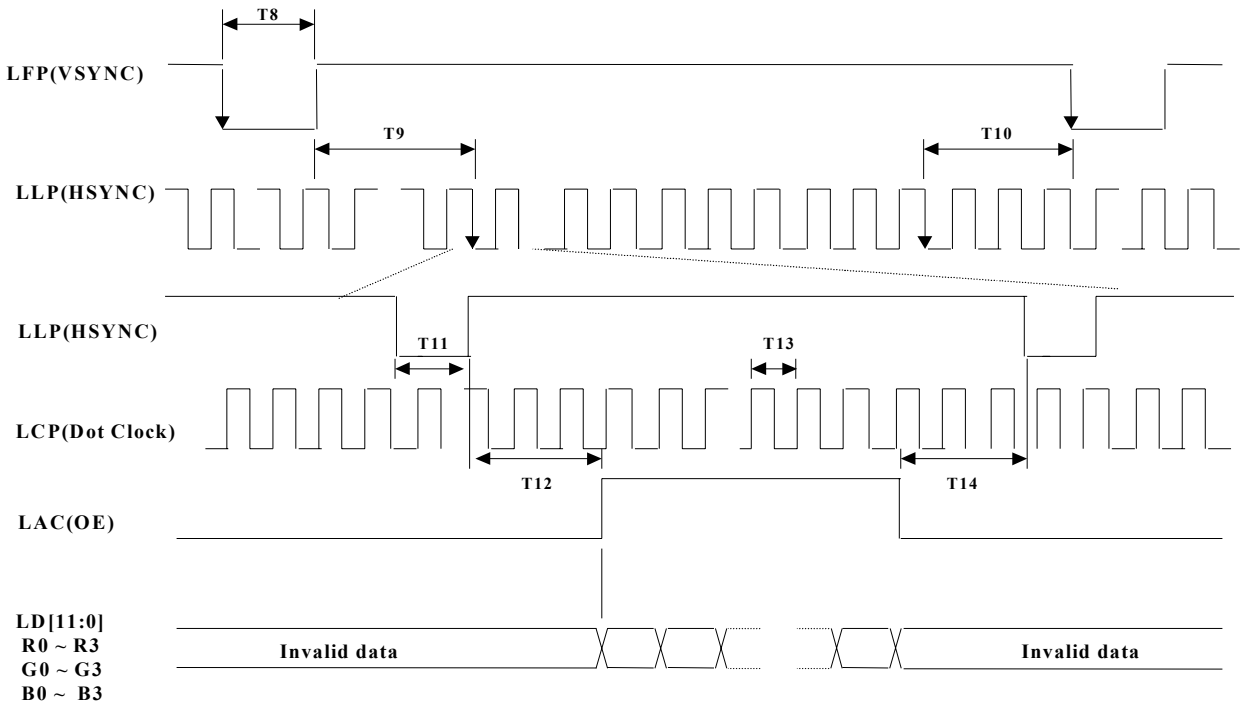


Figure 15-5: LCD Controller Timing (Active-TFT mode)

Symbol	Parameters	Min.	Typ.	Max.	Unit
T1	LCP high time	1	-	16	tCLK(Notes)
T2	LCP low time	1	-	17	tCLK
T3	LLP Front-Porch	1	-	256	tCLK
T4	LLP Pulse width	1	-	256	tCLK
T5	LLP Back-Porch	1	-	256	tCLK
T6	Falling LLP to LFP(LAC) toggle	1	-	256	tCLK
T7	Rising LCP to Display data change	TBD		TBD	ns
T8	VSYNC width	1		64	tHperiod(Notes)
T9	VSYNC back-porch	1		256	tHperiod
T10	VSYNC front-porch	1		256	tHperiod
T11	HSYNC width	1		256	tCLK
T12	HSYNC back-porch	1	-	256	tCLK
T13	HSYNC front-porch	1	-	256	tCLK
T14	Dot Clock Period	1	-	-	tCLK

Table 15-10: LCD interface Signal Timing Parameters

Notes: tCLK is BCLK or VCLK (LCDC internal clock source, 31.5 or 40MHz)
tHperiod Max = 1408 tCLK

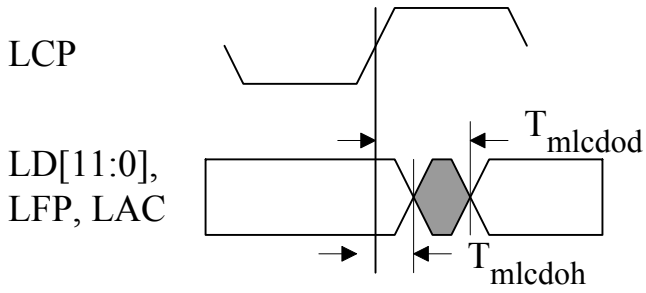


Figure 15-6: STN mode Signal delay

Symbol	Parameter	Min	Max
Tmlcdod	Output delay time from LCP rising	-	5
Tmlcdoh	Output hold time from LCP rising	-	-5

Table 15-11: STN mode Signal delay Parameters

Timing values are derived from simulations using 0pF signal loading. Actual circuit output delays should be calculated by adding manufacturers signal load de-rating delay values.

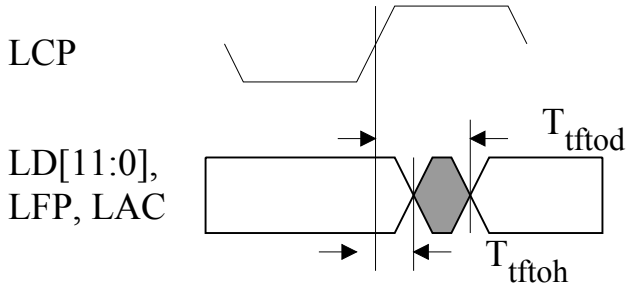


Figure 15-7: TFT mode Signal delay

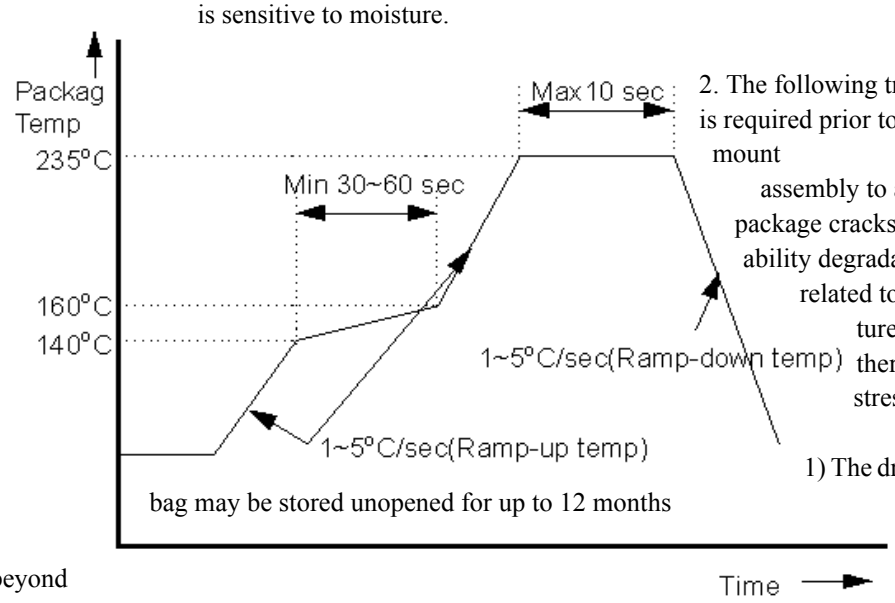
Symbol	Parameter	Min	Max
Ttftod	Output delay time from LCP rising	-	3
Ttftoh	Output hold time from LCP rising	-	-3

Table 15-12: TFT mode Signal delay Parameters

Timing values are derived from simulations using 0pF signal loading. Actual circuit output delays should be calculated by adding manufacturers signal load de-rating delay values.

15.6 Recommended soldering conditions

Recommended Soldering Conditions



1. Bake at 125°C for 4 hours just before soldering
2. After soldering, devices must cool in the room temp minimum 5
3. And then proceed cleaning process (If need)

1. If the product packed in the moisture-proof bag (Dry-pack) that is sensitive to moisture.

2. The following treatment is required prior to surface mount assembly to avoid package cracks or reliability degradation related to moisture and thermal stress.

1) The dry packed bag may be stored unopened for up to 12 months beyond the seal date at < 40°C and 90% relative humidity (RH).

2) The contents of the bag may be stored indefinitely at < 20% RH.

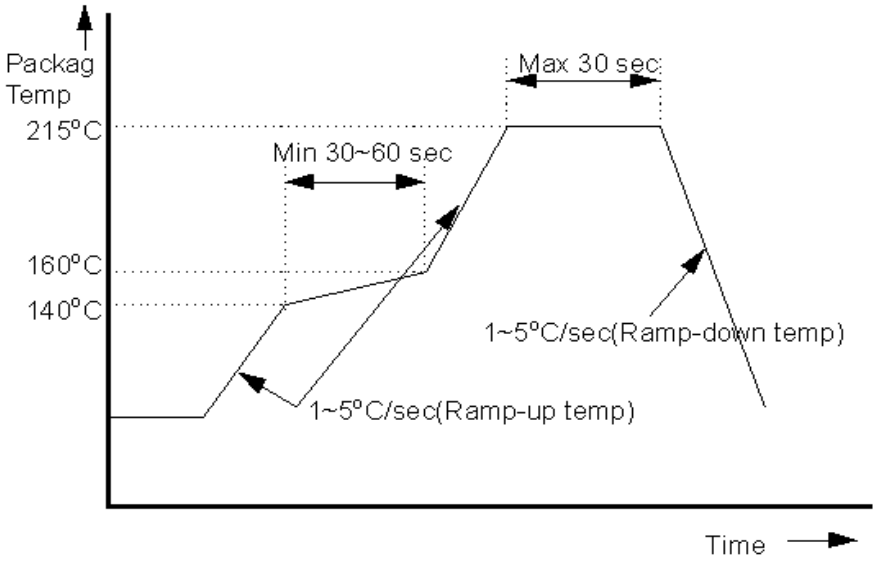
3) If upon opening, the humidity indicator card show humidity above 20%, the contents have expired and required re-baking at 125°C for 20 hours.

4) The contents of the bag must be surface mounted within 48 hours of opening the dry bag. It is recommended the dry bag is not opened until just prior to soldering.

5) Desiccant and humidity indicator card (HIC) contained within dry bag.

Electrical Characteristics

Recommend VPS machine temp

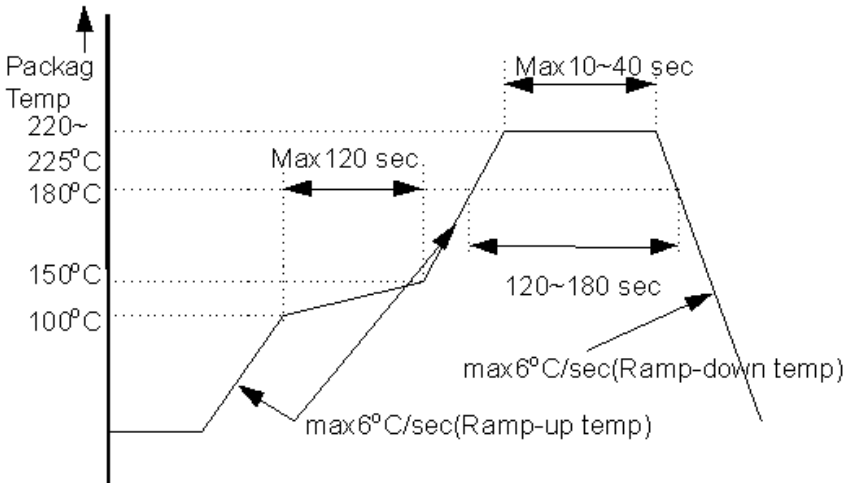


1.Bake at 125°C for 4 hours just before soldering (recommend)

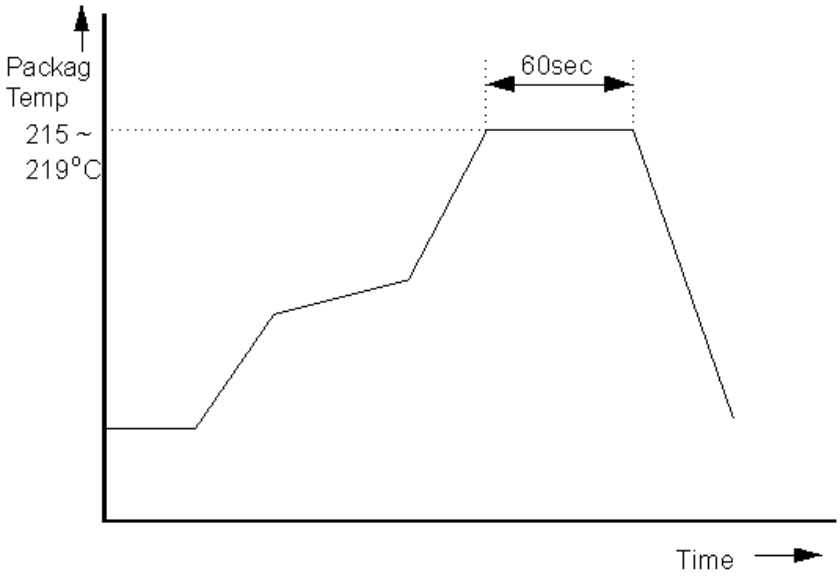
2. After soldering, devices must cool in the room temp minimum 5

3. And then proceed cleaning process (If need)

IR-reflow solder machine temp profile for JESD22-A112-A



VPS solder machine temp profile for JESD22-A112-A



JESD22-A112-ATEST temp profile for VPS.

1. Test method and procondtioning : refer JESD22-A112-A
(free download site - http://www.eia.org/jedec/)