USER MANUAL

LPC2119/2129/2194/2292/2294 USER MANUAL

Preliminary
Supersedes data of 2004 Feb 03

2004 May 03

Philips
Semiconductors

PHILIPS

Table of Contents

List of Figures

List of Tables

DOCUMENT REVISION HISTORY

2003 Dec 03:

- Prototype LPC2119/2129/2194/2292/2294 User Manual created from the design specification.

2003 Dec 09:

- External Memory Controller and Pin Connect Block chapters updated.

2003 Dec 15/16:

- System Control Block chapter updated.

2003 Dec 18:

- A/D Converter Block chapter updated.

2004 Jan 07:

- CAN and PLL related material updated.

2004 Jan 26:

- System Control Block (Crystal Oscillator section - new frequencies added) and CAN Chapter updated.

2004 Feb 03:

- Introduction chapter (register list) updated.

2004 May 03:

- P0.16 description in "Pin Connect Block" chapter corrected from "Reserved" to "Capture 0.2 (TIMER0)".
- LPC2212 Flash size corrected in "Introduction" chapter corrected from 256 to 128 kB.
- Interrupt source #17 in "Vectored Interrupt Controller (VIC)" corrected from "EINT2" to "EINT3".
- Parallel ports 2 and 3 related registers added to "Introduction" and "GPIO" chapters
- Trigger levels determined by bits 7 and 6 in U0FCR and U1FCR ("UART0" and "UART1" chapters) now showed in both decimal and hexadecimal notations
- References to DBGSEL pin removed from entire document (pin does not exist in this family of microcontrollers)
- Pin 20 in figure showing 64-pin package ("Pin Configuration" chapter) corrected from "1.3" to "1.31"
- V_{ddA} replaced with V_{3A} in "A/D Converter" chapter and V_{3A} description updated in "Pin Configuration" chapter
- Warning on analog input levels added to "A/D Converter" chapter
- On-chip upper RAM boundary corrected from 0x4000 1FFF to 0x4000 3FFF in "LPC2119/2129/2292/2294 Memory Addressing" chapter
- Port pin tolerance, pull-up presence and voltage considerations added in "Pin Configuration" and "A/D Converter" chapter
- Baudrates in "Flash Memory System and Programming" corrected: 115200 and 230400 instead of 115000 and 230000
- CAN related interrupt sources fixed in "Vectored Interrupt Controller (VIC)" chapter
- ERRBIT field in CANICR CAN register ("Vectored Interrupt Controller (VIC)" chapter) updated
- Number of the on-chip Flash erase and write cycles added into "Introduction" and "Flash Memory System and Programming" chapters
- Pins capable of providing an External Interrupt functionality are acounted and listed in "System Control Block" chapter
- Access to ports with respect to GPIO configured pins clarified in "GPIO" and "Pin Connect Block" chapters
- Description of Code Read Protection feature added in "Flash Memory System and Programming" chapter

- IOPIN0 and IOPIN1 tyopografic errors corrected in "System Control Block" chapter
- PINSEL2 added to to "Introduction" chapter
- T0IR, T0CCR, T0TCR, T1TCR, T0EMR and PCONP updated in "Introduction" chapter
- EXTMODE and EXTPOLAR registers added in "Introduction" chapter and updated in "System Control Block" chapter
- Power Control Usage Notes for reducing the total power added to "System Control Block" chapter
- PINSEL2 register as well as booting procedure updated in "Pin Connect Block" and "Watchdog" chapters
- references to the pclk in "External Memory Controller (EMC)" chapter corrected to the cclk
- LPC2292/2294 PINSEL2 table in "Pin Connect Block" chapter corrected
- A/D pin description in "A/D Converter" chapter rephrased
- Information on Spurious Interrupts added into "Vectored Interrupt Controller (VIC)" chapter
- Details on the checksum generation in case of Read Memory and Write to RAM ISP commands added in "Flash Memory System and Programming" chapter

1. INTRODUCTION

GENERAL DESCRIPTION

The LPC2119/2129/2194/2292/2294 are based on a 16/32 bit ARM7TDMI-S™ CPU with real-time emulation and embedded trace support, together with 128/256 kilobytes (kB) of embedded high speed flash memory. A 128-bit wide internal memory interface and a unique accelerator architecture enable 32-bit code execution at maximum clock rate. For critical code size applications, the alternative 16-bit Thumb Mode reduces code by more than 30% with minimal performance penalty.

With their comapct 64 and 144 pin packages, low power consumption, various 32-bit timers, combination of 4-channel 10-bit ADC and 2/4 advanced CAN channels or 8-channel 10-bit ADC and 2/4 advanced CAN channels (64 and 144 pin packages respectively), and up to 9 external interrupt pins these microcontrollers are particularly suitable for industrial control, medical systems, access control and point-of-sale.

Number of available GPIOs goes up to 46 in 64 pin package. In 144 pin packages number of available GPIOs tops 76 (with external memory in use) through 112 (single-chip application). Being equipped wide range of serial communications interfaces, they are also very well suited for communication gateways, protocol converters and embedded soft modems as well as many other general-purpose applications.

FEATURES

- 16/32-bit ARM7TDMI-S microcontroller in a 64 or 144 pin package.
- 16 kB on-chip Static RAM
- 128/256 kB on-chip Flash Program Memory (at least 10,000 erate/write cycles over the whole temperature range). 128-bit wide interface/accelerator enables high speed 60 MHz operation.
- External 8, 16 or 32-bit bus (144 pin package only)
- In-System Programming (ISP) and In-Application Programming (IAP) via on-chip boot-loader software. Flash programming takes 1 ms per 512 byte line. Single sector or full chip erase takes 400 ms.
- EmbeddedICE-RT interface enables breakpoints and watch points. Interrupt service routines can continue to execute whilst the foreground task is debugged with the on-chip RealMonitor software.
- Embedded Trace Macrocell enables non-intrusive high speed real-time tracing of instruction execution.
- Two/four interconnected CAN interfaces with advanced acceptance filters.
- Four/eight channel (64/144 pin package) 10-bit A/D converter with conversion time as low as 2.44 ms.
- Two 32-bit timers (with 4 capture and 4 compare channels), PWM unit (6 outputs), Real Time Clock and Watchdog.
- Multiple serial interfaces including two UARTs (16C550), Fast I^2C (400 kbits/s) and two SPIs™.
- 60 MHz maximum CPU clock available from programmable on-chip Phase-Locked Loop.
- Vectored Interrupt Controller with configurable priorities and vector addresses.
- Up to forty-six (64 pin) and hundred-twelve (144 pin package) 5 V tolerant general purpose I/O pins. Up to 12 independent external interrupt pins available (EIN and CAP functions).
- On-chip crystal oscillator with an operating range of 1 MHz to 30 MHz.
- Two low power modes, Idle and Power-down.
- Processor wake-up from Power-down mode via external interrupt.
- Individual enable/disable of peripheral functions for power optimization.
- Dual power supply.
  - CPU operating voltage range of 1.65V to 1.95V (1.8V +/- 8.3%).
  - I/O power supply range of 3.0V to 3.6V (3.3V +/- 10%).

APPLICATIONS

- Industrial control
- Medical systems
- Access control
- Point-of-sale
- Communication gateway
- Embedded soft modem
- general purpose applications

DEVICE INFORMATION

Table 1: LPC2119/2129/2194/2292/2294 device information

Device	No. of pins	On-chip RAM	On-chip FLASH	No. of CAN channels	No. of 10-bit AD Channels	Note
LPC2119	64	16 kB	128 kB	2	4	-
LPC2129	64	16 kB	256 kB	2	4	-
LPC2194	64	16 kB	256 kB	4	4	-
LPC2292	144	16 kB	256 kB	2	8	with external memory interface
LPC2294	144	16 kB	256 kB	4	8	with external memory interface

ARCHITECTURAL OVERVIEW

The LPC2119/2129/2194/2292/2294 consists of an ARM7TDMI-S CPU with emulation support, the ARM7 Local Bus for interface to on-chip memory controllers, the AMBA Advanced High-performance Bus (AHB) for interface to the interrupt controller, and the VLSI Peripheral Bus (VPB, a compatible superset of ARM's AMBA Advanced Peripheral Bus) for connection to on-chip peripheral functions. The LPC2119/2129/2194/2292/2294 configures the ARM7TDMI-S processor in little-endian byte order.

AHB peripherals are allocated a 2 megabyte range of addresses at the very top of the 4 gigabyte ARM memory space. Each AHB peripheral is allocated a 16 kilobyte address space within the AHB address space. LPC2119/2129/2194/2292/2294 peripheral functions (other than the interrupt controller) are connected to the VPB bus. The AHB to VPB bridge interfaces the VPB bus to the AHB bus. VPB peripherals are also allocated a 2 megabyte range of addresses, beginning at the 3.5 gigabyte address point. Each VPB peripheral is allocated a 16 kilobyte address space within the VPB address space.

The connection of on-chip peripherals to device pins is controlled by a Pin Connection Block. This must be configured by software to fit specific application requirements for the use of peripheral functions and pins.

ARM7TDMI-S PROCESSOR

The ARM7TDMI-S is a general purpose 32-bit microprocessor, which offers high performance and very low power consumption. The ARM architecture is based on Reduced Instruction Set Computer (RISC) principles, and the instruction set and related decode mechanism are much simpler than those of microprogrammed Complex Instruction Set Computers. This simplicity results in a high instruction throughput and impressive real-time interrupt response from a small and cost-effective processor core.

Pipeline techniques are employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory.

The ARM7TDMI-S processor also employs a unique architectural strategy known as THUMB, which makes it ideally suited to high-volume applications with memory restrictions, or applications where code density is an issue.

The key idea behind THUMB is that of a super-reduced instruction set. Essentially, the ARM7TDMI-S processor has two instruction sets:

- The standard 32-bit ARM instruction set.
- A 16-bit THUMB instruction set.

The THUMB set's 16-bit instruction length allows it to approach twice the density of standard ARM code while retaining most of the ARM's performance advantage over a traditional 16-bit processor using 16-bit registers. This is possible because THUMB code operates on the same 32-bit register set as ARM code.

THUMB code is able to provide up to 65% of the code size of ARM, and 160% of the performance of an equivalent ARM processor connected to a 16-bit memory system.

The ARM7TDMI-S processor is described in detail in the ARM7TDMI-S Datasheet that can be found on official ARM website.

ON-CHIP FLASH MEMORY SYSTEM

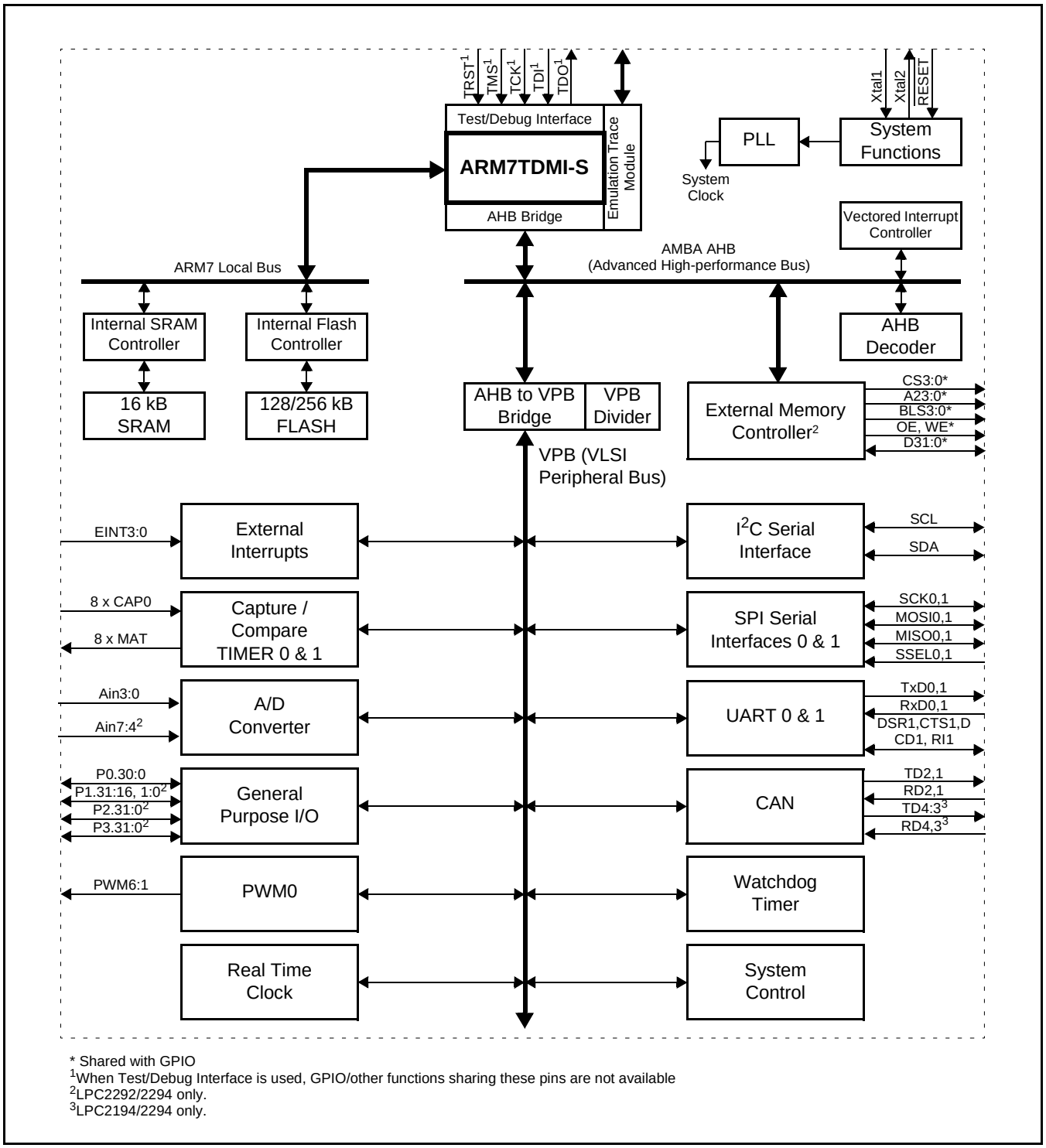
The LPC2219 incorporate a 128 kB Flash memory system, while LPC2129/2194/2292/2294 incorporate a 256 kB Flash memory system. This memory may be used for both code and data storage. Programming of the Flash memory may be accomplished in several ways: over the serial built-in JTAG interface, using In System Programming (ISP) and UART0, or by means of In Application Programming (IAP) capabilities. The application program, using the In Application Programming (IAP) functions, may also erase and/or program the Flash while the application is running, allowing a great degree of flexibility for data storage field firmware upgrades, etc.

ON-CHIP STATIC RAM

The LPC2119/2129/2194/2292/2294 provide a 16 kB static RAM memory that may be used for code and/or data storage. The SRAM supports 8-bit, 16-bit, and 32-bit accesses.

The SRAM controller incorporates a write-back buffer in order to prevent CPU stalls during back-to-back writes. The write-back buffer always holds the last data sent by software to the SRAM. This data is only written to the SRAM when another write is requested by software (the data is only written to the SRAM when software does another write). If a chip reset occurs, actual SRAM contents will not reflect the most recent write request (i.e. after a "warm" chip reset, the SRAM does not reflect the last write operation). Any software that checks SRAM contents after reset must take this into account. Two identical writes to a location guarantee that the data will be present after a Reset. Alternatively, a dummy write operation before entering idle or power-down mode will similarly guarantee that the last data written will be present in SRAM after a subsequent Reset.

BLOCK DIAGRAM



* Shared with GPIO
[1]When Test/Debug Interface is used, GPIO/other functions sharing these pins are not available
[2]LPC2292/2294 only.
[3]LPC2194/2294 only.

Figure 1: LPC2119/2129/2194/2292/2294 Block Diagram

LPC2119/2129/2292/2194/2294 REGISTERS

Accesses to registers in LPC2119/2129/2194/2292/2294 is restricted in the following ways:

1) user must NOT attempt to access any register locations not defined.

2) Access to any defined register locations must be strictly for the functions for the registers.

3) Register bits labeled '-', '0' or '1' can ONLY be written and read as follows:

- '-' MUST be written with '0', but can return any value when read (even if it was written with '0'). It is a reserved bit and may be used in future derivatives.
- '0' MUST be written with '0', and will return a '0' when read.
- '1' MUST be written with '1', and will return a '1' when read.

The following table shows all registers available in LPC2119/2129/2194/2292/2294 microcontroller sorted according to the address.

Access to the specific one can be categorized as either read/write, read only or write only (R/W, RO and WO respectively).

"Reset Value" field refers to the data stored in used/accessible bits only. It does not include reserved bits content. Some registers may contain undetermined data upon reset. In this case, reset value is categorized as "undefined". Classification as "NA" is used in case reset value is not applicable. Some registers in RTC are not affected by the chip reset. Their reset value is marked as * and these registers must be initialized by software if the RTC is enabled.

Registers in LPC2119/2129/2194/2292/2294 are 8, 16 or 32 bits wide. For 8 bit registers shown in Table 2, bit residing in the MSB (The Most Significant Bit) column corresponds to the bit 7 of that register, while bit in the LSB (The Least Significant Bit) column corresponds to the bit 0 of the same register.

If a register is 16/32 bit wide, the bit residing in the top left corner of its description, is the bit corresponding to the bit 15/31 of the register, while the bit in the bottom right corner corresponds to bit 0 of this register.

Examples: bit "ENA6" in PWMPCR register (address 0xE001404C) represents the bit at position 14 in this register; bits 15, 8, 7 and 0 in the same register are reserved. Bit "Stop on MR6" in PWMMCR register (0xE0014014) corresponds to the bit at position 20; bits 31 to 21 of the same register are reserved.

Unused (reserved) bits are marked with "-" and represented as gray fields. Access to them is restricted as already described.

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
WD												
0xE0000000	WD MOD	Watchdog mode register	-	-	-	-	WD INT	WD TOF	WDRESET	WDEN	R/W	0
0xE0000004	WDTC	Watchdog timer constant register	32 bit data								R/W	0xFF
0xE0000008	WD FEED	Watchdog feed sequence register	8 bit data (0xAA fallowed by 0x55)								WO	NA
0xE000000C	WDTV	Watchdog timer value register	32 bit data								RO	0xFF
TIMER0												
0xE0004000	T0IR	T0 Interrupt Register	CR3 Int.	CR2 Int.	CR1 Int.	CR0 Int.	MR3 Int.	MR2 Int.	MR1 Int.	MR0 Int.	R/W	0
0xE0004004	T0TCR	T0 Control Register	-	-	-	-	-	-	CTR Reset	CTR Enable	R/W	0
0xE0004008	T0TC	T0 Counter	32 bit data								RW	0
0xE000400C	T0PR	T0 Prescale Register	32 bit data								R/W	0
0xE0004010	T0PC	T0 Prescale Counter	32 bit data								R/W	0
0xE0004014	T0MCR	T0 Match Control Register	4 reserved (-) bits				Stop on MR3	Reset on MR3	Int. on MR3	Stop on MR2	R/W	0
			Reset on MR2	Int. on MR2	Stop on MR1	Reset on MR1	Int. on MR1	Stop on MR0	Reset on MR0	Int. on MR0		
0xE0004018	T0MR0	T0 Match Register 0	32 bit data								R/W	0
0xE000401C	T0MR1	T0 Match Register 1	32 bit data								R/W	0
0xE0004020	T0MR2	T0 Match Register 2	32 bit data								R/W	0
0xE0004024	T0MR3	T0 Match Register 3	32 bit data								R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0004028	T0CCR	T0 Capture Control Register	4 reserved (-) bits				Int. on Cpt.3	Int. on Cpt.3 falling	Int. on Cpt.3 rising	Int. on Cpt.2	R/W	0
			Int. on Cpt.2 falling	Int. on Cpt.2 rising	Int. on Cpt.1	Int. on Cpt.1 falling	Int. on Cpt.1 rising	Int. on Cpt.0	Int. on Cpt.0 falling	Int. on Cpt.0 rising		
0xE000402C	T0CR0	T0 Capture Register 0	32 bit data								RO	0
0xE0004030	T0CR1	T0 Capture Register 1	32 bit data								RO	0
0xE0004034	T0CR2	T0 Capture Register 2	32 bit data								RO	0
0xE000403C	T0EMR	T0 External Match Register	4 reserved (-) bits				External Match Control 3		External Match Control 2		R/W	0
			External Match Control 1		External Match Control 0		Ext. Mtch3.	Ext. Mtch2.	Ext. Mtch.1	Ext. Mtch.0		
TIMER1												
0xE0008000	T1IR	T1 Interrupt Register	CR3 Int.	CR2 Int.	CR1 Int.	CR0 Int.	MR3 Int.	MR2 Int.	MR1 Int.	MR0 Int.	R/W	0
0xE0008004	T1TCR	T1 Control Register	-	-	-	-	-	-	CTR Reset	CTR Enable	R/W	0
0xE0008008	T1TC	T1 Counter	32 bit data								RW	0
0xE000800C	T1PR	T1 Prescale Register	32 bit data								R/W	0
0xE0008010	T1PC	T1 Prescale Counter	32 bit data								R/W	0
0xE0008014	T1MCR	T1 Match Control Register	4 reserved (-) bits				Stop on MR3	Reset on MR3	Int. on MR3	Stop on MR2	R/W	0
			Reset on MR2	Int. on MR2	Stop on MR1	Reset on MR1	Int. on MR1	Stop on MR0	Reset on MR0	Int. on MR0		
0xE0008018	T1MR0	T1 Match Register 0	32 bit data								R/W	0
0xE000801C	T1MR1	T1 Match Register 1	32 bit data								R/W	0
0xE0008020	T1MR2	T1 Match Register 2	32 bit data								R/W	0
0xE0008024	T1MR3	T1 Match Register 3	32 bit data								R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0008028	T1CCR	T1 Capture Control Register	4 reserved (-) bits				Int. on Cpt.3	Int. on Cpt.3 falling	Int. on Cpt.3 rising	Int. on Cpt.2	R/W	0
			Int. on Cpt.2 falling	Int. on Cpt.2 rising	Int. on Cpt.1	Int. on Cpt.1 falling	Int. on Cpt.1 rising	Int. on Cpt.0	Int. on Cpt.0 falling	Int. on Cpt.0 rising		
0xE000802C	T1CR0	T1 Capture Register 0	32 bit data								RO	0
0xE0008030	T1CR1	T1 Capture Register 1	32 bit data								RO	0
0xE0008034	T1CR2	T1 Capture Register 2	32 bit data								RO	0
0xE0008038	T1CR3	T1 Capture Register 3	32 bit data								RO	0
0xE000803C	T1EMR	T1 External Match Register	4 reserved (-) bits				External Match Control 3		External Match Control 2		R/W	0
			External Match Control 1		External Match Control 0		Ext. Mtch.3	Ext. Mtch2.	Ext. Mtch.1	Ext. Mtch.0		
UART0												
0xE000C000	U0RBR (DLAB=0)	U0 Receiver Buffer Register		8 bit data							RO	un-defined
	U0THR (DLAB=0)	U0 Transmit Holding Register		8 bit data							WO	NA
	U0DLL (DLAB=1)	U0 Divisor Latch LSB		8 bit data							R/W	0x01
0xE000C004	U0IER (DLAB=0)	U0 Interrupt Enable Register	0	0	0	0	0	En. Rx Line Status Int.	Enable THRE Int.	En. Rx Data Av.Int.	R/W	0
	U0DLM (DLAB=1)	U0 Divisor Latch MSB		8 bit data							R/W	0
0xE000C008	U0IIR	U0 Interrupt ID Register	FIFOs Enabled		0	0	IIR3	IIR2	IIR1	IIR0	RO	0x01
	U0FCR	U0 FIFO Control Register	Rx Trigger		-	-	-	U0 Tx FIFO Reset	U0 Rx FIFO Reset	U0 FIFO Enable	WO	0
0xE000C00C	U0LCR	U0 Line Control Register	DLAB	Set Break	Stick Parity	Even Parity Select	Parity Enable	Nm. of Stop Bits	Word Length Select		R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE000C014	U0LSR	U0 Line Status Register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60
0xE000C01C	U0SCR	U0 Scratch Pad Register	8 bit data								R/W	0
UART1												
0xE0010000	U1RBR (DLAB=0)	U1 Receiver Buffer Register	8 bit data								RO	un-defined
	U1THR (DLAB=0)	U1 Transmit Holding Register	8 bit data								WO	NA
	U1DLL (DLAB=1)	U1 Divisor Latch LSB	8 bit data								R/W	0x01
0xE0010004	U1IER (DLAB=0)	U1 Interrupt Enable Register	0	0	0	0	En. Mdem Satus Int.	En. Rx Line Status Int.	Enable THRE Int.	En. Rx Data Av.Int.	R/W	0
	U1DLM (DLAB=1)	U1 Divisor Latch MSB	8 bit data								R/W	0
0xE0010008	U1IIR	U1 Interrupt ID Register	FIFOs Enabled		0	0	IIR3	IIR2	IIR1	IIR0	RO	0x01
	U1FCR	U1 FIFO Control Register	Rx Trigger		-	-	-	U0 Tx FIFO Reset	U0 Rx FIFO Reset	U0 FIFO Enable	WO	0
0xE001000C	U1LCR	U1 Line Control Register	DLAB	Set Break	Stick Parity	Even Parity Select	Parity Enable	Nm. of Stop Bits	Word Length Select		R/W	0
0xE0010010	U1 MCR	U1 Modem Control Register	0	0	0	Loop Back	0	0	RTS	DTR	R/W	0
0xE0010014	U1LSR	U1 Line Status Register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60
0xE001001C	U1SCR	U1 Scratch Pad Register	8 bit data								R/W	0
0xE0010018	U1 MSR	U1 Modem Status Register	DCD	RI	DSR	CTS	Delta DCD	Trailing Edge RI	Delta DSR	Delta CTS	RO	0
PWM												

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0014000	PWM IR	PWM Interrupt Register	-	-	-	-	-	MR6 Int.	MR5 Int.	MR4 Int.	R/W	0
			-	-	-	-	MR3 Int.	MR2 Int.	MR1 Int.	MR0 Int.		
0xE0014004	PWM TCR	PWM Timer Control Register	-	-	-	-	PWM Enable	-	CTR Reset	CTR Enable	R/W	0
0xE0014008	PWM TC	PWM Timer Counter	32 bit data								RW	0
0xE001400C	PWM PR	PWM Prescale Register	32 bit data								R/W	0
0xE0014010	PWM PC	PWM Prescale Counter	32 bit data								R/W	0
0xE0014014	PWM MCR	PWM Match Control Register	11 reserved (-) bits			Stop on MR6	Reset on MR6	Int. on MR6	Stop on MR5	Reset on MR5	R/W	0
			Int. on MR5	Stop on MR4	Reset on MR4	Int. on MR4	Stop on MR3	Reset on MR3	Int. on MR3	Stop on MR2		
			Reset on MR2	Int. on MR2	Stop on MR1	Reset on MR1	Int. on MR1	Stop on MR0	Reset on MR0	Int. on MR0		
0xE0014018	PWM MR0	PWM Match Register 0	32 bit data								R/W	0
0xE001401C	PWM MR1	PWM Match Register 1	32 bit data								R/W	0
0xE0014020	PWM MR2	PWM Match Register 2	32 bit data								R/W	0
0xE0014024	PWM MR3	PWM Match Register 3	32 bit data								R/W	0
0xE0014040	PWM MR4	PWM Match Register 4	32 bit data								R/W	0
0xE0014044	PWM MR5	PWM Match Register 5	32 bit data								R/W	0
0xE0014048	PWM MR6	PWM Match Register 6	32 bit data								R/W	0
0xE001404C	PWM PCR	PWM Control Register	-	ENA6	ENA5	ENA4	ENA3	ENA2	ENA1	-	R/W	0
			-	SEL6	SEL5	SEL4	SEL3	SEL2	SEL1	-		

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0014050	PWM LER	PWM Latch Enable Register	-	Ena. PWM M6 Latch	Ena. PWM M5 Latch	Ena. PWM M4 Latch	Ena. PWM M3 Latch	Ena. PWM M2 Latch	Ena. PWM M1 Latch	Ena. PWM M0 Latch	R/W	0
I^2C												
0xE001C000	I2CONSET	I^2C Control Set Register	-	I2EN	STA	STO	SI	AA	-	-	R/W	0
0xE001C004	I2STAT	I^2C Status Register	5 bit Status					0	0	0	RO	0xF8
0xE001C008	I2DAT	I^2C Data Register	8 bit data								R/W	0
0xE001C00C	I2 ADR	I^2C Slave Address Register	7 bit data							GC	R/W	0
0xE001C010	I2 SCLH	SCL Duty Cycle Register High Half Word	16 bit data								R/W	0x04
0xE001C014	I2 SCLL	SCL Duty Cycle Register Low Half Word	16 bit data								R/W	0x04
0xE001C018	I2CONCLR	I^2C Control Clear Register	-	I2ENC	STAC	-	SIC	AAC	-	-	WO	NA
SPI0												
0xE0020000	S0 SPCR	SPI0 Control Register	SPIE	LSBF	MSTR	CPOL	CPHA	-	-	-	R/W	0
0xE0020004	S0 SPSR	SPI0 Status Register	SPIF	WCOL	ROVR	MODF	ABRT	-	-	-	RO	0
0xE0020008	S0 SPDR	SPI0 Data Register	8 bit data								R/W	0
0xE002000C	S0 SPCCR	SPI0 Clock Counter Register	8 bit data								R/W	0
0xE002001C	S0 SPINT	SPI0 Interrupt Flag	-	-	-	-	-	-	-	SPI Int.	R/W	0
SPI1												
0xE0030000	S1 SPCR	SPI1 Control Register	SPIE	LSBF	MSTR	CPOL	CPHA	-	-	-	R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0030004	S1 SPSR	SPI1 Status Register	SPIF	WCOL	ROVR	MODF	ABRT	-	-	-	RO	0
0xE0030008	S1 SPDR	SPI1 Data Register	8 bit data								R/W	0
0xE003000C	S1 SPCCR	SPI1 Clock Counter Register	8 bit data								R/W	0
0xE003001C	S1 SPINT	SPI1 Interrupt Flag	-	-	-	-	-	-	-	SPI Int.	R/W	0
RTC												
0xE0024000	ILR	Interrupt Location Register	-	-	-	-	-	-	RTC ALF	RTC CIF	R/W	*
0xE0024004	CTC	Clock Tick Counter	15 bit data							-	RO	*
0xE0024008	CCR	Clock Control Register	-	-	-	-	CTTEST		CTC RST	CLK EN	R/W	*
0xE002400C	CIIR	Counter Increment Interrupt Register	IM YEAR	IM MON	IM DOY	IM DOW	IM DOM	IM HOUR	IM MIN	IM SEC	R/W	*
0xE0024010	AMR	Alarm Mask Register	AMR YEAR	AMR MON	AMR DOY	AMR DOW	AMR DOM	AMR HOUR	AMR MIN	AMR SEC	R/W	*
0xE0024014	CTIME0	Consolidated Time Register 0	-	-	-	-	-	3 bit Day of Week			RO	*
			-	-	-	5 bit Hours						
			-	-	6 bit Minutes							
			-	-	6 bit Seconds							
0xE0024018	CTIME1	Consolidated Time Register 1	-	-	-	-					RO	*
			12 bit Year									
			-	-	-	-	4 bit Month					
			-	-	-	5 bit Day of Month						
0xE002401C	CTIME2	Consolidated Time Register 2	reserved (-) 20 bits				12 bit Day of Year				RO	*
0xE0024020	SEC	Seconds Register	-	-	6 bit data						R/W	*
0xE0024024	MIN	Minutes Register	-	-	6 bit data						R/W	*
0xE0024028	HOUR	Hours Register	-	-	-	5 bit data					R/W	*

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE002402C	DOM	Day of Month Register	-	-	-	5 bit data					R/W	*
0xE0024030	DOW	Day of Week Register	-	-	-	-	-	3 bit data			R/W	*
0xE0024034	DOY	Day of Year Register	reserved (-) 7 bits				9 bit data				R/W	*
0xE0024038	MONTH	Months Register	-	-	-	-	4 bit data				R/W	*
0xE002403C	YEAR	Year Register	reserved (-) 4 bits			12 bit data					R/W	*
0xE0024060	AL SEC	Alarm value for Seconds	-	-	6 bit data						R/W	*
0xE0024064	AL MIN	Alarm value for Minutes	-	-	6 bit data						R/W	*
0xE0024068	AL HOUR	Alarm value for Hours	-	-	-	5 bit data					R/W	*
0xE002406C	AL DOM	Alarm value for Day of Month	-	-	-	5 bit data					R/W	*
0xE0024070	AL DOW	Alarm value for Day of Week	-	-	-	-	-	3 bit data			R/W	*
0xE0024074	AL DOY	Alarm value for Day of Year	reserved (-) 7 bits			9 bit data					R/W	*
0xE0024078	AL MON	Alarm value for Months	-	-	-	-	4 bit data				R/W	*
0xE002407C	AL YEAR	Alarm value for Year	reserved (-) 4 bits		12 bit data						R/W	*
0xE0024080	PRE INT	Prescale value, integer portion	reserved (-) 3 bits		13 bit data						R/W	0
0xE0024084	PRE FRAC	Prescale value, fractional portion	-	15 bit data							R/W	0
GPIO PORT0												
0xE0028000	IO0PIN	GPIO 0 Pin Value reg.	32 bit data								RO	NA
0xE0028004	IO0SET	GPIO 0 Out. Set register	32 bit data								R/W	0
0xE0028008	IO0DIR	GPIO 0 Dir. control reg.	32 bit data								R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE002800C	IO0CLR	GPIO 0 Out. Clear register	32 bit data								WO	0
GPIO PORT1												
0xE0028010	IO1PIN	GPIO 1 Pin Value reg.	32 bit data								RO	NA
0xE0028014	IO1SET	GPIO 1 Out. Set register	32 bit data								R/W	0
0xE0028018	IO1DIR	GPIO 1 Dir. control reg.	32 bit data								R/W	0
0xE002801C	IO1CLR	GPIO 1 Out. Clear register	32 bit data								WO	0
GPIO PORT2												
0xE0028020	IO2PIN	GPIO 2 Pin Value reg.	32 bit data								RO	NA
0xE0028024	IO2SET	GPIO 2 Out. Set register	32 bit data								R/W	0
0xE0028028	IO2DIR	GPIO 2 Dir. control reg.	32 bit data								R/W	0
0xE002802C	IO2CLR	GPIO 2 Out. Clear register	32 bit data								WO	0
GPIO PORT3												
0xE0028030	IO3PIN	GPIO 3 Pin Value reg.	32 bit data								RO	NA
0xE0028034	IO3SET	GPIO 3 Out. Set register	32 bit data								R/W	0
0xE0028038	IO3DIR	GPIO 3 Dir. control reg.	32 bit data								R/W	0
0xE002803C	IO3CLR	GPIO 3 Out. Clear register	32 bit data								WO	0
Pin Connet Block												
0xE002C000	PIN SEL0	Pin function select register 0	32 bit data								R/W	0
0xE002C004	PIN SEL1	Pin function select register 1	32 bit data								R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE002C014	PIN SEL2	Pin function select register 2	-								R/W	0
			24-bit pin configuration data (144 package case)									
			Reserved bits (64 package case)									
							configuration data			-		
ADC												
0xE0034000	ADCR	ADC Control register	-				EDGE	START			RW	01
			TEST1:0		PDN	-	CLKS			BURST		
			8 bit data									
			8 bit data									
0xE0034004	ADDR	ADC Data register	DONE	OVER RUN	-		CHN				RW	x
			-									
					10 bit data							
					-							
CAN												
0xE0038000-0xE00387FF		2K RAM (512 x 32) of lookup receive identifiers.									RW	0
0xE003C000	AFMR	Acceptance Filter Reg.	-	-	-	-	-	eFCAN	AccBP	AccOff	RW	1
0xE003C004	SFF_sa	Standard Frame Indiv. Start. Addr.	-	-	-	-	-	9 - bit data			RW	0
			9 - bit data						-	-		
0xE003C008	SFF_ GRP_sa	Standard Frame Group Start. Addr.	-	-	-	-	10 - bit data >>>				RW	0
			<<< 10 - bit data						-	-		
0xE003C00C	EFF_sa	Extended Frame Indiv. Start. Addr.	-	-	-	-	-	9 - bit data			RW	0
			9 - bit data						-	-		
0xE003C010	EFF_ GRP_sa	Extended Frame Group Start. Addr.	-	-	-	-	10 - bit data >>>				RW	0
			<<< 10 - bit data						-	-		
0xE003C014	ENDof Table	End of AF Tables Reg.	-	-	-	-	10 - bit data >>>				RW	0
			<<< 10 - bit data						-	-		
0xE003C018	LUT errAd	LUT Error Address Reg.	-	-	-	-	-	9 - bit data			RO	0
			9 - bit data						-	-		

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE003C01C	LUTerr	LUT Error Register	-	-	-	-	-	-	-	Error	RO	0
0xE0040000	CAN TxSR	CAN Central Transmit Status Register	-	-	-	-	-	-	-	-	R/O	0x003F 3F00
			-	-	-	-	TCS4:1					
			-	-	-	-	TBS4:1					
			-	-	-	-	TS4:1					
0xE0040004	CAN RxSR	CAN Central Receive Status Register	-	-	-	-	-	-	-	-	R/O	0
			-	-	-	-	DOS4:1					
			-	-	-	-	RBS4:1					
			-	-	-	-	RS4:1					
0xE0040008	CAN MSR	CAN Central Miscellaneous Register	-	-	-	-	-	-	-	-	RO	0
			-	-	-	-	-	-	-	-		
			-	-	-	-	BS4:1					
			-	-	-	-	ES4:1					
CAN1 Interface												
0xE0044000	C1MOD	CAN1 Mode Register	-	-	-	-	-	-	-	-	R/W	0x0000 0001
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			TM	-	RPM	SM	TPM	STM	LOM	RM		
0xE0044004	C1CMR	CAN1 Command Register	-	-	-	-	-	-	-	-	W0	NA
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			STB3	STB2	STB1	SRR	CDO	RRB	AT	TR		
0xE0044008	C1GSR	CAN1 Global Status Register	8-bit data TXERR								RO	0x0000 000C
			8-bit data RXERR									
			-	-	-	-	-	-	-	-		
			BS	ES	TS	RS	TCS	TBS	DOS	RBS		
0xE004400C	C1ICR	CAN1 Interrupt and Capture Register	-	-	-	ALCBIT					RO	0x0000 0000
			ERRC		ERR DIR	ERRBIT						
			-	-	-	-	-	TI3	TI2	IDI		
			BEI	ALI	EPI	WUI	DOI	EI	TI1	RI		

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0044010	C1IER	CAN1 Interrupt Enable Register	-	-	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	TIE3	TIE2	IDIE		
			BEIE	ALIE	EPIE	WUIE	DOIE	EIE	TIE1	RIE		
0xE0044014	C1BTR	CAN1 Bus Timing Register	-	-	-	-	-	-	-	-	R/W	0x001C 0000
			SAM	TSEG2			TSEG1					
			SJW		-	-	-	-	10-bit data>>>			
			<<< 10-bit data BRP									
0xE0044018	C1EWL	CAN1 Error Warning Limit Register	-	-	-	-	-	-	-	-	R/W	0x0000 0060
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			EWL									
0xE004401C	C1SR	CAN1 Status Register	-	-	-	-	-	-	-	-	RO	0x000C 0C0C
			BS	ES	TS3	RS	TCS3	TBS3	DOS	RBS		
			BS	ES	TS2	RS	TCS2	TBS2	DOS	RBS		
			BS	ES	TS1	RS	TCS1	TBS1	DOS	RBS		
0xE0044020	C1RFS	CAN1 Rx Frame Status Register	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	BP	10-bit data>>>			
			<<< 10-bit data ID Index									
0xE0044024	C1RID	CAN1 Rx identifier Register	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0044028	C1RDA	CAN1 Rx Data Register A	Data 4								R/W	0x0000 0000
			Data 3									
			Data 2									
			Data 1									
0xE004402C	C1RDB	CAN1 Rx Data Register B	Data 8								R/W	0x0000 0000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0044030	C1TFI1	CAN1 Tx Frame Information Register (buffer 1)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0044034	C1TID1	CAN1 Tx Identifier Register (buffer 1)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0044038	C1TDA1	CAN1 Tx Data Register A (buffer 1)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004403C	C1TDB1	CAN1 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
0xE0044040	C1TFI2	CAN1 Tx Frame Information Register (buffer 2)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0044044	C1TID2	CAN1 Tx Identifier Register (buffer 2)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0044048	C1TDA2	CAN1 Tx Data Register A (buffer 2)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004404C	C1TDB2	CAN1 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0044050	C1TFI3	CAN1 Tx Frame Information Register (buffer 3)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0044054	C1TID3	CAN1 Tx Identifier Register (buffer 3)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0044058	C1TDA3	CAN1 Tx Data Register A (buffer 3)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004405C	C1TDB3	CAN1 Tx Data Register B (buffer 3)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
CAN2 Interface												
0xE0048000	C2MOD	CAN2 Mode Register	-	-	-	-	-	-	-	-	R/W	0x0000 0001
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			TM	-	RPM	SM	TPM	STM	LOM	RM		
0xE0048004	C2CMR	CAN2 Command Register	-	-	-	-	-	-	-	-	W0	NA
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			STB3	STB2	STB1	SRR	CDO	RRB	AT	TR		
0xE0048008	C2GSR	CAN2 Global Status Register	8-bit data TXERR								RO	0x0000 000C
			8-bit data RXERR									
			-	-	-	-	-	-	-	-		
			BS	ES	TS	RS	TCS	TBS	DOS	RBS		
0xE004800C	C2ICR	CAN2 Interrupt and Capture Register	-	-	-	ALCBIT					RO	0x0000 0000
			ERRC		ERR DIR	ERRBIT						
			-	-	-	-	-	TI3	TI2	IDI		
			BEI	ALI	EPI	WUI	DOI	EI	TI1	RI		

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0048010	C2IER	CAN2 Interrupt Enable Register	-	-	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	TIE3	TIE2	IDIE		
			BEIE	ALIE	EPIE	WUIE	DOIE	EIE	TIE1	RIE		
0xE0048014	C2BTR	CAN2 Bus Timing Register	-	-	-	-	-	-	-	-	R/W	0x001C 0000
			SAM	TSEG2			TSEG1					
			SJW		-	-	-	-	10-bit data>>>			
			<<< 10-bit data BRP									
0xE0048018	C2EWL	CAN2 Error Warning Limit Register	-	-	-	-	-	-	-	-	R/W	0x0000 0060
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			EWL									
0xE004801C	C2SR	CAN2 Status Register	-	-	-	-	-	-	-	-	RO	0x000C 0C0C
			BS	ES	TS3	RS	TCS3	TBS3	DOS	RBS		
			BS	ES	TS2	RS	TCS2	TBS2	DOS	RBS		
			BS	ES	TS1	RS	TCS1	TBS1	DOS	RBS		
0xE0048020	C2RFS	CAN2 Rx Frame Status Register	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	BP	10-bit data>>>			
			<<< 10-bit data ID Index									
0xE0048024	C2RID	CAN2 Rx identifier Register	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0048028	C2RDA	CAN2 Rx Data Register A	Data 4								R/W	0x0000 0000
			Data 3									
			Data 2									
			Data 1									
0xE004802C	C2RDB	CAN2 Rx Data Register B	Data 8								R/W	0x0000 0000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0048030	C2TFI1	CAN2 Tx Frame Information Register (buffer 1)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0048034	C2TID1	CAN2 Tx Identifier Register (buffer 1)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0048038	C2TDA1	CAN2 Tx Data Register A (buffer 1)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004803C	C2TDB1	CAN2 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
0xE0048040	C2TFI2	CAN2 Tx Frame Information Register (buffer 2)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0048044	C2TID2	CAN2 Tx Identifier Register (buffer 2)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0048048	C2TDA2	CAN2 Tx Data Register A (buffer 2)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004804C	C2TDB2	CAN2 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0048050	C2TFI3	CAN2 Tx Frame Information Register (buffer 3)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0048054	C2TID3	CAN2 Tx Identifier Register (buffer 3)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0048058	C2TDA3	CAN2 Tx Data Register A (buffer 3)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004805C	C2TDB3	CAN2 Tx Data Register B (buffer 3)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
CAN3 Interface (LPC2194/2294 only)												
0xE004C000	C3MOD	CAN3 Mode Register	-	-	-	-	-	-	-	-	R/W	0x0000 0001
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			TM	-	RPM	SM	TPM	STM	LOM	RM		
0xE004C004	C3CMR	CAN3 Command Register	-	-	-	-	-	-	-	-	W0	NA
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			STB3	STB2	STB1	SRR	CDO	RRB	AT	TR		
0xE004C008	C3GSR	CAN3 Global Status Register	8-bit data TXERR								RO	0x0000 000C
			8-bit data RXERR									
			-	-	-	-	-	-	-	-		
			BS	ES	TS	RS	TCS	TBS	DOS	RBS		
0xE004C00C	C3ICR	CAN3 Interrupt and Capture Register	-	-	-	ALCBIT					RO	0x0000 0000
			ERRC		ERR DIR	ERRBIT						
			-	-	-	-	-	TI3	TI2	IDI		
			BEI	ALI	EPI	WUI	DOI	EI	TI1	RI		

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE004C010	C3IER	CAN3 Interrupt Enable Register	-	-	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	TIE3	TIE2	IDIE		
			BEIE	ALIE	EPIE	WUIE	DOIE	EIE	TIE1	RIE		
0xE004C014	C3BTR	CAN3 Bus Timing Register	-	-	-	-	-	-	-	-	R/W	0x001C 0000
			SAM	TSEG2			TSEG1					
			SJW		-	-	-	-	10-bit data>>>			
			<<< 10-bit data BRP									
0xE004C018	C3EWL	CAN3 Error Warning Limit Register	-	-	-	-	-	-	-	-	R/W	0x0000 0060
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			EWL									
0xE004C01C	C3SR	CAN3 Status Register	-	-	-	-	-	-	-	-	RO	0x000C 0C0C
			BS	ES	TS3	RS	TCS3	TBS3	DOS	RBS		
			BS	ES	TS2	RS	TCS2	TBS2	DOS	RBS		
			BS	ES	TS1	RS	TCS1	TBS1	DOS	RBS		
0xE004C020	C3RFS	CAN3 Rx Frame Status Register	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	BP	10-bit data>>>			
			<<< 10-bit data ID Index									
0xE004C024	C3RID	CAN3 Rx identifier Register	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE004C028	C3RDA	CAN3 Rx Data Register A	Data 4								R/W	0x0000 0000
			Data 3									
			Data 2									
			Data 1									
0xE004C02C	C3RDB	CAN3 Rx Data Register B	Data 8								R/W	0x0000 0000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE004C030	C3TFI1	CAN3 Tx Frame Information Register (buffer 1)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE004C034	C3TID1	CAN3 Tx Identifier Register (buffer 1)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE004C038	C3TDA1	CAN3 Tx Data Register A (buffer 1)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004C03C	C3TDB1	CAN3 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
0xE004C040	C3TFI2	CAN3 Tx Frame Information Register (buffer 2)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE004C044	C3TID2	CAN3 Tx Identifier Register (buffer 2)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE004C048	C3TDA2	CAN3 Tx Data Register A (buffer 2)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004C04C	C3TDB2	CAN3 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE004C050	C3TFI3	CAN3 Tx Frame Information Register (buffer 3)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE004C054	C3TID3	CAN3 Tx Identifier Register (buffer 3)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE004C058	C3TDA3	CAN3 Tx Data Register A (buffer 3)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE004C05C	C3TDB3	CAN3 Tx Data Register B (buffer 3)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
CAN4 Interface (LPC2194/2294 only)												
0xE0050000	C4MOD	CAN4 Mode Register	-	-	-	-	-	-	-	-	R/W	0x0000 0001
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			TM	-	RPM	SM	TPM	STM	LOM	RM		
0xE0050004	C4CMR	CAN4 Command Register	-	-	-	-	-	-	-	-	W0	NA
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			STB3	STB2	STB1	SRR	CDO	RRB	AT	TR		
0xE0050008	C4GSR	CAN4 Global Status Register	8-bit data TXERR								RO	0x0000 000C
			8-bit data RXERR									
			-	-	-	-	-	-	-	-		
			BS	ES	TS	RS	TCS	TBS	DOS	RBS		
0xE005000C	C4ICR	CAN4 Interrupt and Capture Register	-	-	-	ALCBIT					RO	0x0000 0000
			ERRC		ERR DIR	ERRBIT						
			-	-	-	-	-	TI3	TI2	IDI		
			BEI	ALI	EPI	WUI	DOI	EI	TI1	RI		

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0050010	C4IER	CAN4 Interrupt Enable Register	-	-	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	TIE3	TIE2	IDIE		
			BEIE	ALIE	EPIE	WUIE	DOIE	EIE	TIE1	RIE		
0xE0050014	C4BTR	CAN4 Bus Timing Register	-	-	-	-	-	-	-	-	R/W	0x001C 0000
			SAM	TSEG2			TSEG1					
			SJW		-	-	-	-	10-bit data>>>			
			<<< 10-bit data BRP									
0xE0050018	C4EWL	CAN4 Error Warning Limit Register	-	-	-	-	-	-	-	-	R/W	0x0000 0060
			-	-	-	-	-	-	-	-		
			-	-	-	-	-	-	-	-		
			EWL									
0xE005001C	C4SR	CAN4 Status Register	-	-	-	-	-	-	-	-	RO	0x000C 0C0C
			BS	ES	TS3	RS	TCS3	TBS3	DOS	RBS		
			BS	ES	TS2	RS	TCS2	TBS2	DOS	RBS		
			BS	ES	TS1	RS	TCS1	TBS1	DOS	RBS		
0xE0050020	C4RFS	CAN4 Rx Frame Status Register	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	BP	10-bit data>>>			
			<<< 10-bit data ID Index									
0xE0050024	C4RID	CAN4 Rx identifier Register	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0050028	C4RDA	CAN4 Rx Data Register A	Data 4								R/W	0x0000 0000
			Data 3									
			Data 2									
			Data 1									
0xE005002C	C4RDB	CAN4 Rx Data Register B	Data 8								R/W	0x0000 0000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0050030	C4TFI1	CAN4 Tx Frame Information Register (buffer 1)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0050034	C4TID1	CAN4 Tx Identifier Register (buffer 1)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0050038	C4TDA1	CAN4 Tx Data Register A (buffer 1)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE005003C	C4TDB1	CAN4 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
0xE0050040	C4TFI2	CAN4 Tx Frame Information Register (buffer 2)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0050044	C4TID2	CAN4 Tx Identifier Register (buffer 2)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0050048	C4TDA2	CAN4 Tx Data Register A (buffer 2)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE005004C	C4TDB2	CAN4 Tx Data Register B (buffer 2)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE0050050	C4TFI3	CAN4 Tx Frame Information Register (buffer 3)	FF	RTR	-	-	-	-	-	-	R/W	0x0000 0000
			-	-	-	-	DLC					
			-	-	-	-	-	-	-	-		
			PRIO									
0xE0050054	C4TID3	CAN4 Tx Identifier Register (buffer 3)	-	-	-						R/W	0x0000 0000
			29-bit (FF=1) or									
			11-bit data (FF=0) ID									
0xE0050058	C4TDA3	CAN4 Tx Data Register A (buffer 3)	Data 4								R/W	0x0000 000
			Data 3									
			Data 2									
			Data 1									
0xE005005C	C4TDB3	CAN4 Tx Data Register B (buffer 3)	Data 8								R/W	0x0000 000
			Data 7									
			Data 6									
			Data 5									
System Control Block												
0xE01FC000	MAM CR	MAM control register	-	-	-	-	-	-	2 bit data		R/W	0
0xE01FC004	MAM TIM	MAM timing control	-	-	-	-	-	3 bit data			R/W	0x07
0xE01FC040	MEM MAP	Memory mapping control	-	-	-	-	-	-	2 bit data		R/W	0
0xE01FC080	PLL CON	PLL control register	-	-	-	-	-	-	PLLC	PLLE	R/W	0
0xE01FC084	PLL CFG	PLL configuration register	-	2bit data PSEL		5 bit data MSEL					R/W	0
0xE01FC088	PLL STAT	PLL status register	-	-	-	-	-	PLOCK	PLLC	PLLE	RO	0
			-	2bit data PSEL		5 bit data MSEL						
0xE01FC08C	PLL FEED	PLL feed register	8 bit data								WO	NA
0xE01FC0C0	PCON	Power control register	-	-	-	-	-	-	PD	IDL	R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xE01FC0C4	PCONP	Power control for peripherals	reserved (-) 19 bits			PCAD	-	PC SPI1	PC RTC	PC SPI0	R/W	0x3BE
			PC I2C	-	PC PWM0	PC URT1	PC URT0	PC TIM1	PC TIM0	-		
0xE01FC100	VPB DIV	VPB divider control	-	-	-	-	-	-	2 bit data		R/W	0
0xE01FC140	EXT INT	External interrupt flag register	-	-	-	-	EINT3	EINT2	EINT1	EINT0	R/W	0
0xE01FC144	EXT WAKE	External interrupt wakeup register	-	-	-	-	EXT WAKE 3	EXT WAKE 2	EXT WAKE 1	EXT WAKE 0	R/W	0
0xE01FC148	EXT MODE	External interrupt mode register	-	-	-	-	EXT MODE 3	EXT MODE 2	EXT MODE 1	EXT MODE 0	R/W	0
0xE01FC14C	EXT POLAR	External interrupt polarity register	-	-	-	-	EXT POLAR 3	EXT POLAR 2	EXT POLAR 1	EXT POLAR 0	R/W	0
External memory Controller - EMC												
0xFFE00000	BCFG0	Conf. Reg. for mem bank 0	AT		MW ($\overline{\text{BOOT1:0}}$)		BM	WP	WP ERR	BUS ERR	R/W	0x0000 FBEF
			-	-	-	-	-	-	-	-		
			WST2					RBLE	WST1			
			WST1			-	IDCY					
0xFFE00004	BCFG1	Conf. Reg. for mem bank 1	AT		MW (0x2)		BM	WP	WP ERR	BUS ERR	R/W	0x2000 FBEF
			-	-	-	-	-	-	-	-		
			WST2					RBLE	WST1			
			WST1			-	IDCY					
0xFFE00008	BCFG2	Conf. Reg. for mem bank 2	AT		MW (0x1)		BM	WP	WP ERR	BUS ERR	R/W	0x1000 FBEF
			-	-	-	-	-	-	-	-		
			WST2					RBLE	WST1			
			WST1			-	IDCY					

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xFFE0000C	BCFG3	Conf. Reg. for mem bank 3	AT		MW (0x0)		BM	WP	WP ERR	BUS ERR	R/W	0x0000 FBEF
			-	-	-	-	-	-	-	-		
			WST2					RBLE	WST1			
			WST1			-	IDCY					
Vectored Interrupt Controller - VIC												
0xFFFFF000	VICIRQ Status	IRQ Status Register	32-bit data								RO	0
0xFFFFF004	VICFIQ Status	FIQ Status Register	32-bit data								RO	0
0xFFFFF008	VIC RawIntr	Raw Interrupt Status Reg.	32-bit data								RO	0
0xFFFFF00C	VICInt Select	Interrupt Select Reg.	32-bit data								R/W	0
0xFFFFF010	VICInt Enable	Interrupt Enable Reg.	32-bit data								R/W	0
0xFFFFF014	VICInt EnClear	Int. Enable Clear Reg.	32-bit data								WO	0
0xFFFFF018	VICSoft Int	Software Interrupt Reg.	32-bit data								R/W	0
0xFFFFF01C	VICSoftI ntClear	Software Int. Clear Reg.	32-bit data								W	0
0xFFFFF020	VIC Protection	Protection Enable Reg.	32-bit data								R/W	0
0xFFFFF030	VICVect Addr	Vector Address Reg.	32-bit data								R/W	0
0xFFFFF034	VICDefv ectAddr	Default Vec. Addr.Reg.	32-bit data								R/W	0
0xFFFFF100	VICVect Addr0	Vector adr. 0 reg.	32-bit data								R/W	0
0xFFFFF104	VICVect Addr1	Vector adr. 1 reg.	32-bit data								R/W	0
⋮												
0xFFFFF13C	VICVect Addr15	Vector adr. 15 reg.	32-bit data								R/W	0

Table 2: LPC2119/2129/2194/2292/2294 Registers

Address Offset	Name	Description	MSB							LSB	Access	Reset Value
0xFFFFF200	VICVect Cntl0	Vect. Control 0 Reg.	-	-	1-bit data	5-bit data					R/W	0
0xFFFFF204	VICVect Cntl1	Vect. Control 1 Reg.	-	-	1-bit data	5-bit data					R/W	0
⋮												
0xFFFFF23C	VICVect Cntl15	Vect. Control 15 Reg.	-	-	1-bit data	5-bit data					R/W	0

2. LPC2119/2129/2292/2294 MEMORY ADDRESSING

MEMORY MAPS

The LPC2119/2129/2194/2292/2294 incorporates several distinct memory regions, shown in the following figures. Figure 2 shows the overall map of the entire address space from the user program viewpoint following reset. The interrupt vector area supports address re-mapping, which is described later in this section.

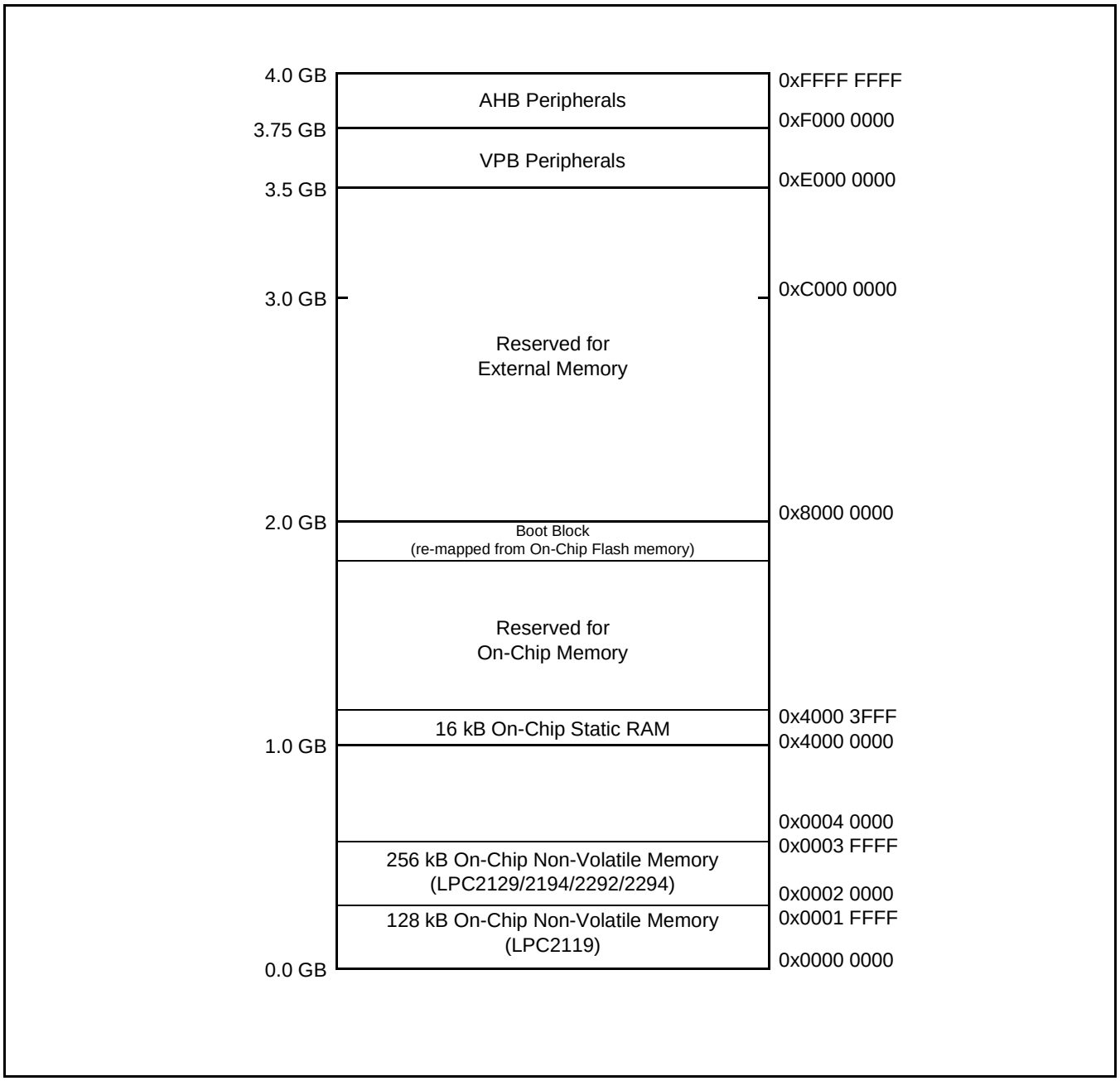


Figure 2: System Memory Map

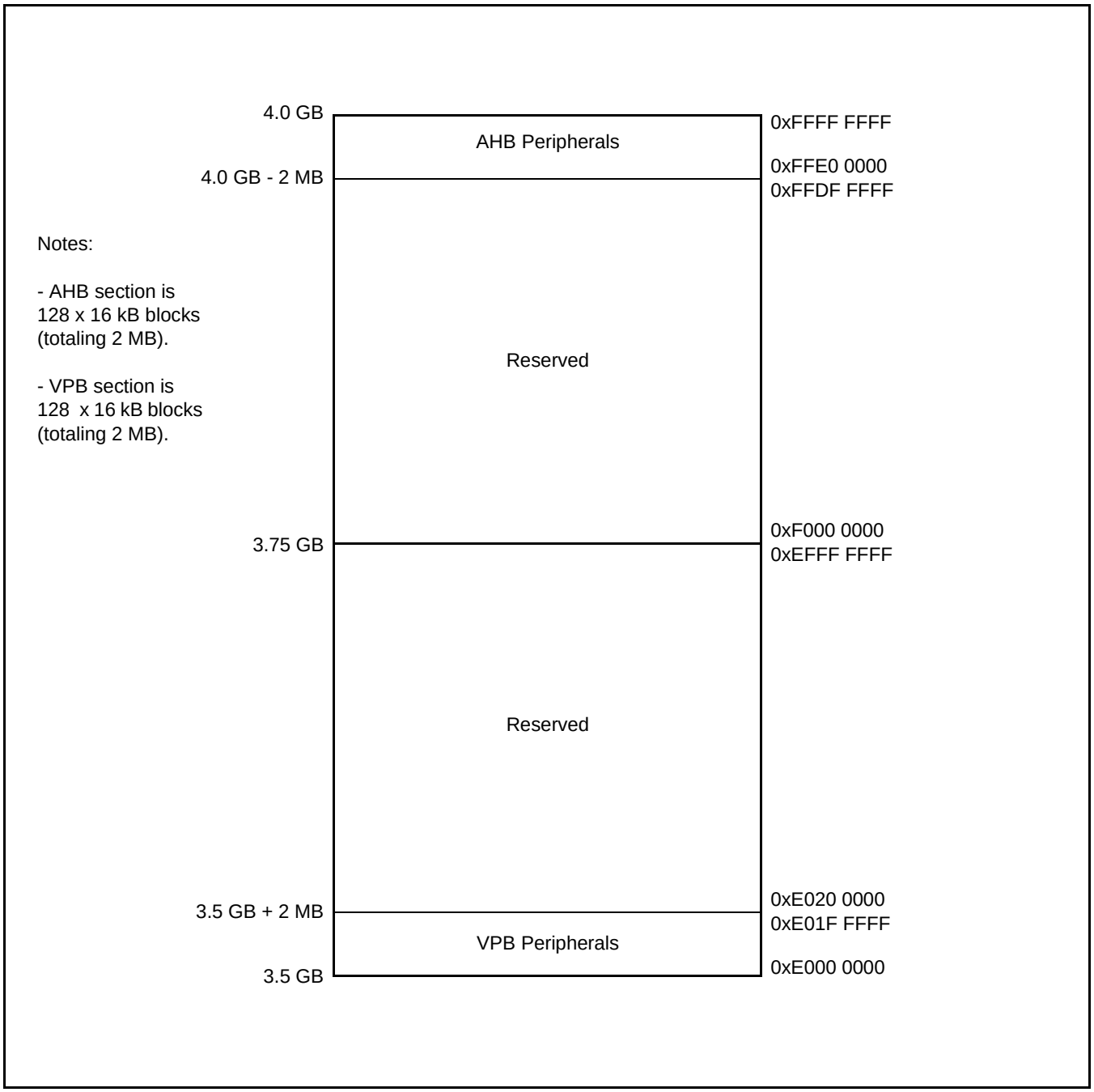


Figure 3: Peripheral Memory Map

Figures 3 through 5 show different views of the peripheral address space. Both the AHB and VPB peripheral areas are 2 megabyte spaces which are divided up into 128 peripherals. Each peripheral space is 16 kilobytes in size. This allows simplifying the address decoding for each peripheral. All peripheral register addresses are word aligned (to 32-bit boundaries) regardless of their size. This eliminates the need for byte lane mapping hardware that would be required to allow byte (8-bit) or half-word (16-bit) accesses to occur at smaller boundaries. An implication of this is that word and half-word registers must be accessed all at once. For example, it is not possible to read or write the upper byte of a word register separately.

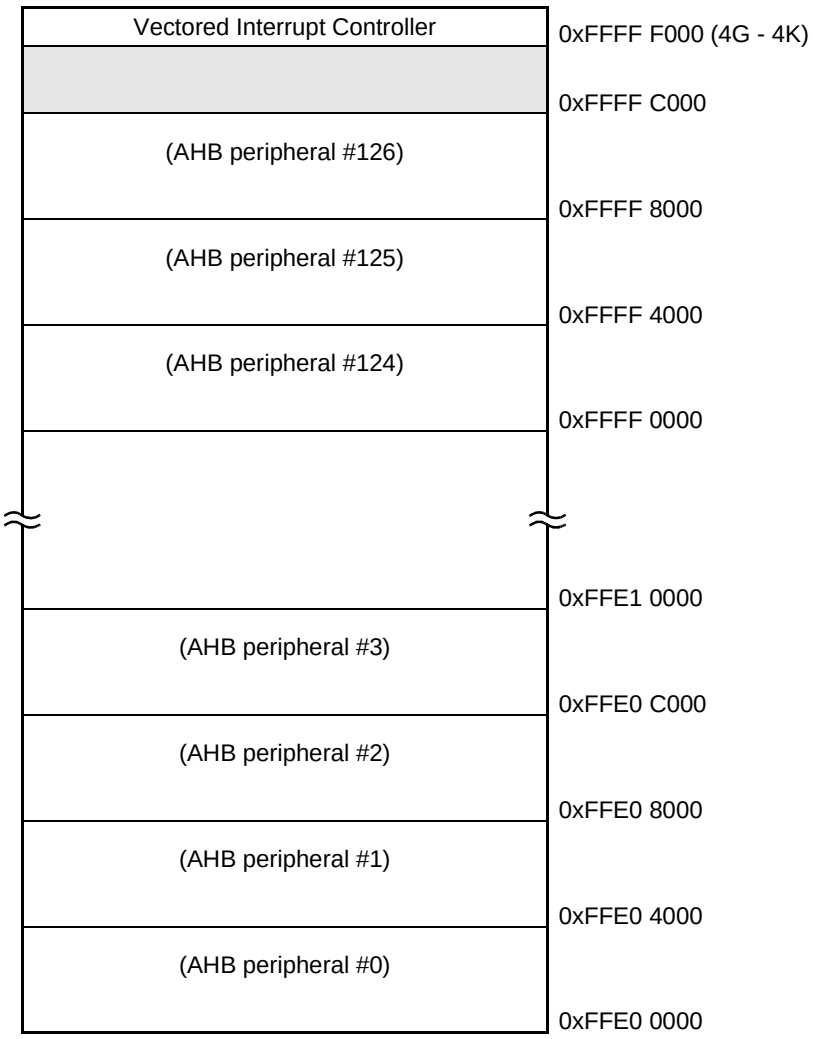


Figure 4: AHB Peripheral Map

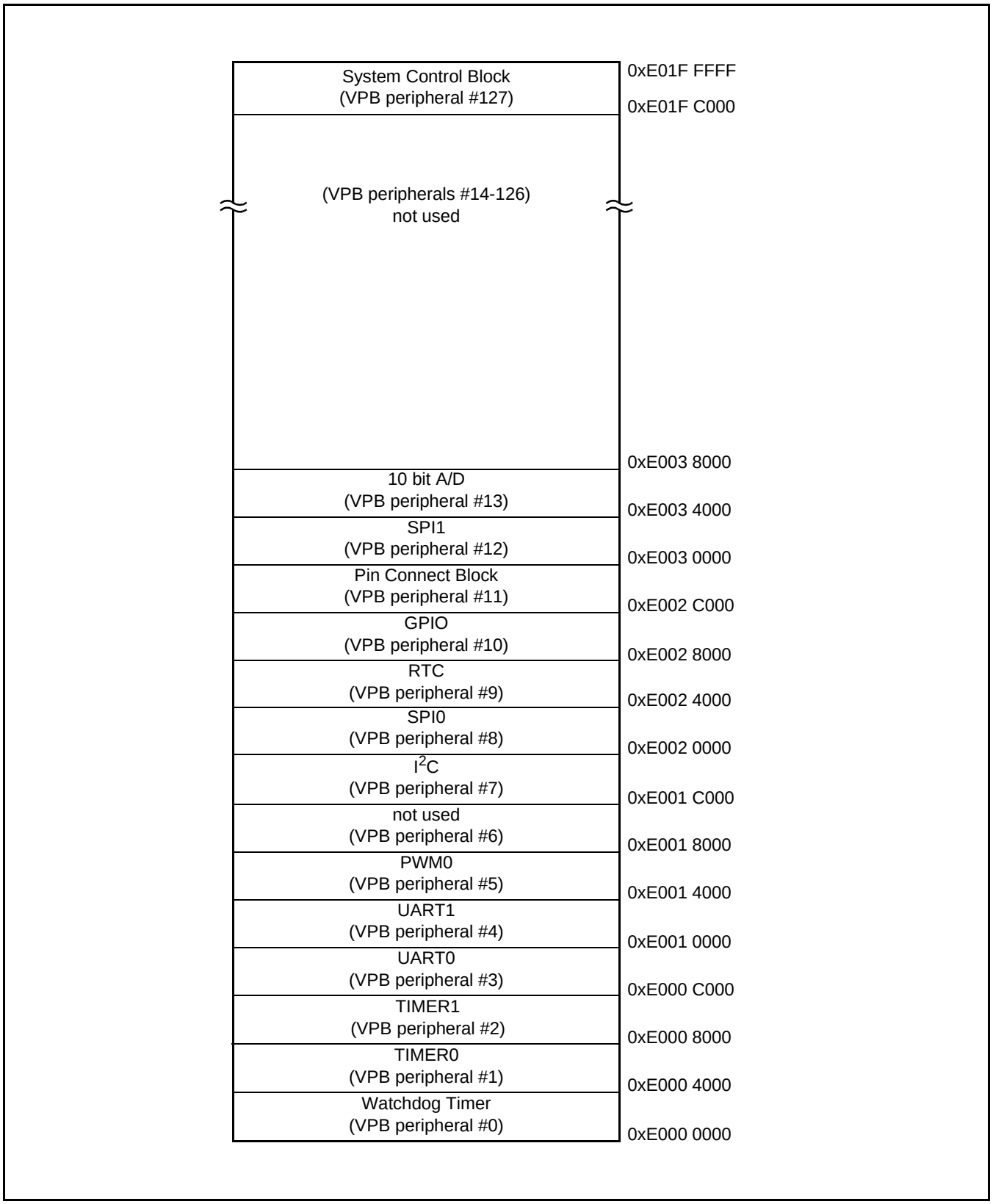


Figure 5: VPB Peripheral Map

LPC2119/2129/2194/2292/2294 MEMORY RE-MAPPING AND BOOT BLOCK

Memory Map Concepts and Operating Modes

The basic concept on the LPC2119/2129/2194/2292/2294 is that each memory area has a "natural" location in the memory map. This is the address range for which code residing in that area is written. The bulk of each memory space remains permanently fixed in the same location, eliminating the need to have portions of the code designed to run in different address ranges.

Because of the location of the interrupt vectors on the ARM7 processor (at addresses 0x0000 0000 through 0x0000 001C, as shown in Table 3 below), a small portion of the Boot Block and SRAM spaces need to be re-mapped in order to allow alternative uses of interrupts in the different operating modes described in Table 4. Re-mapping of the interrupts is accomplished via the Memory Mapping Control feature described in the System Control Block section.

Table 3: ARM Exception Vector Locations

Address	Exception
0x0000 0000	Reset
0x0000 0004	Undefined Instruction
0x0000 0008	Software Interrupt
0x0000 000C	Prefetch Abort (instruction fetch memory fault)
0x0000 0010	Data Abort (data access memory fault)
0x0000 0014	Reserved *
0x0000 0018	IRQ
0x0000 001C	FIQ

*: Identified as reserved in ARM documentation, this location is used by the Boot Loader as the Valid User Program key. This is descibed in detail in Flash Memory System and Programming on page 262.

Table 4: LPC2119/2129/2194/2292/2294 Memory Mapping Modes

Mode	Activation	Usage
Boot Loader mode	Hardware activation by any Reset	The Boot Loader always executes after any reset. The Boot Block interrupt vectors are mapped to the bottom of memory to allow handling exceptions and using interrupts during the Boot Loading process.
User Flash mode	Software activation by Boot code	Activated by Boot Loader when a valid User Program Signature is recognized in memory and Boot Loader operation is not forced. Interrupt vectors are not re-mapped and are found in the bottom of the Flash memory.
User RAM mode	Software activation by User program	Activated by a User Program as desired. Interrupt vectors are re-mapped to the bottom of the Static RAM.
User External mode	Activated by BOOT1:0 pins not 11 at Reset	Activated by the Boot Loader when either or both BOOT pins are low at the end of RESET low. Interrupt vectors are re-mapped from the bottom of the external memory map. **Note:** This mode is available in LPC2292/2294 only!

Memory Re-Mapping

In order to allow for compatibility with future derivatives, the entire Boot Block is mapped to the top of the on-chip memory space. In this manner, the use of larger or smaller flash modules will not require changing the location of the Boot Block (which would require changing the Boot Loader code itself) or changing the mapping of the Boot Block interrupt vectors. Memory spaces other than the interrupt vectors remain in fixed locations. Figure 6 shows the on-chip memory mapping in the modes defined above.

The portion of memory that is re-mapped to allow interrupt processing in different modes includes the interrupt vector area (32 bytes) and an additional 32 bytes, for a total of 64 bytes. The re-mapped code locations overlay addresses 0x0000 0000 through 0x0000 003F. A typical user program in the Flash memory can place the entire FIQ handler at address 0x0000 001C without any need to consider memory boundaries. The vector contained in the SRAM, external memory, and Boot Block must contain branches to the actual interrupt handlers, or to other instructions that accomplish the branch to the interrupt handlers.

There are three reasons this configuration was chosen:

1. To give the FIQ handler in the Flash memory the advantage of not having to take a memory boundary caused by the re-mapping into account.
2. Minimize the need to for the SRAM and Boot Block vectors to deal with arbitrary boundaries in the middle of code space.
3. To provide space to store constants for jumping beyond the range of single word branch instructions.

Re-mapped memory areas, including the Boot Block and interrupt vectors, continue to appear in their original location in addition to the re-mapped address.

Details on re-mapping and examples can be found in System Control Block on page 64.

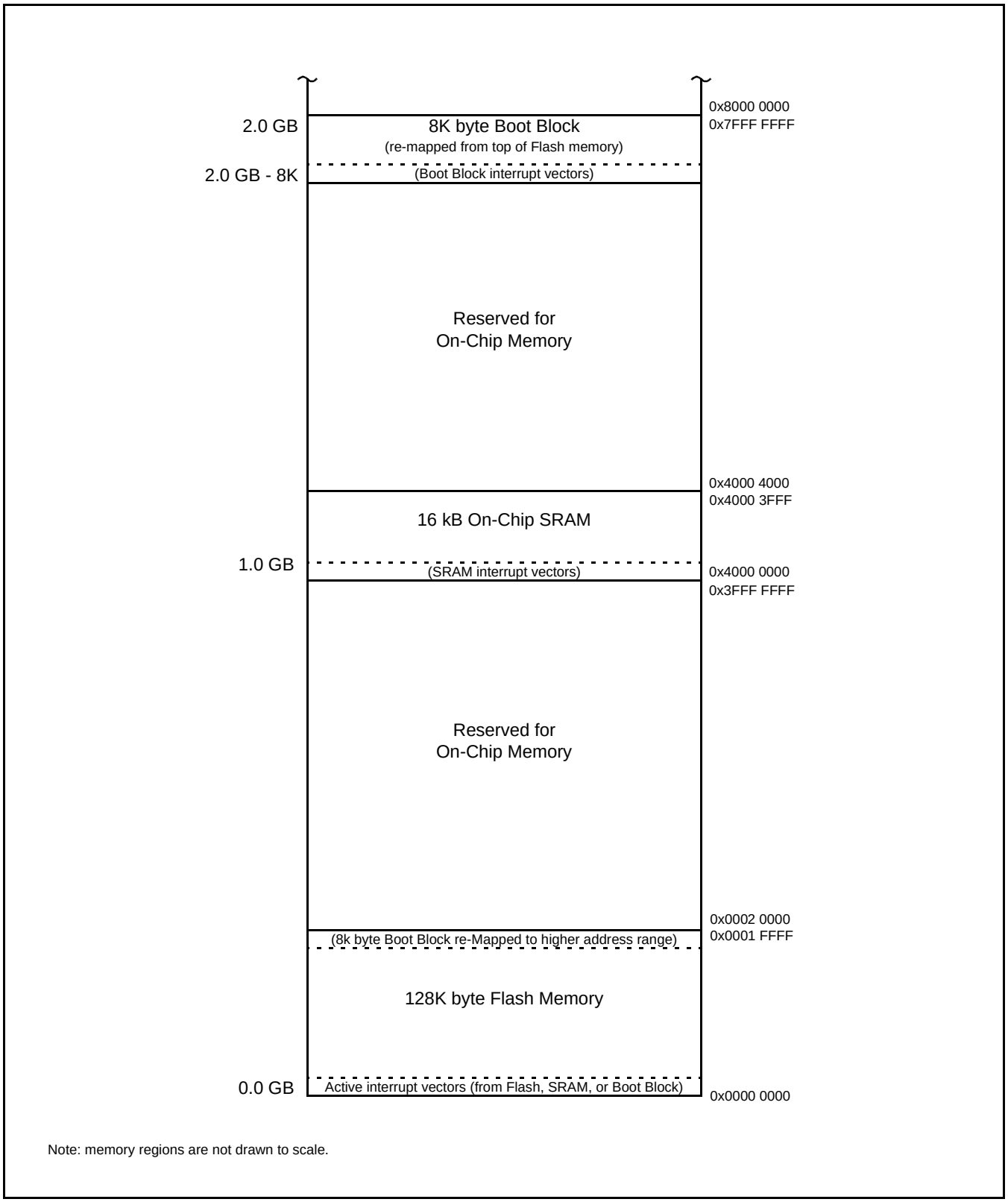


Figure 6: Map of lower memory is showing re-mapped and re-mappable areas (128 kB Flash).

PREFETCH ABORT AND DATA ABORT EXCEPTIONS

The LPC2119/2129/2194/2292/2294 generates the appropriate bus cycle abort exception if an access is attempted for an address that is in a reserved or unassigned address region. The regions are:

- Areas of the memory map that are not implemented for a specific ARM derivative. For the LPC2119/2129/2194/2292/2294, this is:
  - Address space between On-Chip Non-Volatile Memory and On-Chip SRAM, labelled "Reserved for On-Chip Memory" in Figure 2 and Figure 6. For 128 kB Flash device, this is memory address range from 0x0002 0000 to 0x3FFF FFFF, while for 256 kB Flash device this range is from 0x0004 0000 to 0x3FFF FFFF.
  - Address space between On-Chip Static RAM and External Memory. Labelled "Reserved for On-Chip Memory" in Figure 2. This is an address range from 0x4000 3FFF to 0x7FFF DFFF.
  - External Memory other than that provided by the EMC in the 144-pin package.
  - Reserved regions of the AHB and VPB spaces. See Figure 3.
- Unassigned AHB peripheral spaces. See Figure 4.
- Unassigned VPB peripheral spaces. See Figure 5.

For these areas, both attempted data access and instruction fetch generate an exception. In addition, a Prefetch Abort exception is generated for any instruction fetch that maps to an AHB or VPB peripheral address.

Within the address space of an existing VPB peripheral, a data abort exception is not generated in response to an access to an undefined address. Address decoding within each peripheral is limited to that needed to distinguish defined registers within the peripheral itself. For example, an access to address 0xE000D000 (an undefined address within the UART0 space) may result in an access to the register defined at address 0xE000C000. Details of such address aliasing within a peripheral space are not defined in the LPC2119/2129/2194/2292/2294 documentation and are not a supported feature.

Note that the ARM core stores the Prefetch Abort flag along with the associated instruction (which will be meaningless) in the pipeline and processes the abort only if an attempt is made to execute the instruction fetched from the illegal address. This prevents accidental aborts that could be caused by prefetches that occur when code is executed very near a memory boundary.

3. EXTERNAL MEMORY CONTROLLER (EMC)

This module is available in LPC2292 and LPC2294 only.

FEATURES

- Supports static memory-mapped devices including RAM, ROM, flash, burst ROM, and some external I/O devices.
- Asynchronous page mode read operation in non-clocked memory subsystems
- Asynchronous burst mode read access to burst mode ROM devices
- Independent configuration for up to four banks, each up to 16M Bytes
- Programmable bus turnaround (idle) cycles (1 to 16)
- Programmable read and write WAIT states (up to 32), for static RAM devices
- Programmable initial and subsequent burst read WAIT state, for burst ROM devices
- Programmable write protection
- Programmable burst mode operation
- Programmable external data width, 8, 16, or 32 bits
- Programmable read byte lane enable control

DESCRIPTION

The external Static Memory Controller is an AMBA AHB slave module which provides an interface between an AMBA AHB system bus and external (off-chip) memory devices. It provides support for up to four independently configurable memory banks simultaneously. Each memory bank is capable of supporting SRAM, ROM, Flash EPROM, Burst ROM memory, or some external I/O devices

Each memory bank may be 8, 16, or 32 bits wide.

This module is available in LPC2219 and LPC2294only. Since this 144 pin package pins out address lines A[23:0], the decoding among the four banks uses address bits A[25:24]. The native location of the four banks is at the start of the External Memory area identified in Figure 2 on page 48, but Bank 0 can be used for initial booting under control of the state of the BOOT[1:0] pins.

Bank	Address Range	Configuration Register
0	8000 0000 - 80FF FFFF	BCFG0
1	8100 0000 - 81FF FFFF	BCFG1
2	8200 0000 - 82FF FFFF	BCFG2
3	8300 0000 - 83FF FFFF	BCFG3

Table 5: Address Ranges of External Memory Banks (LPC2292/2294only)

PIN DESCRIPTION

Pin Name	Type	Pin Description
D[31:0]	Input/ Output	External memory data lines.
A[23:0]	Output	External memory address lines.
OE	Output	Low-active Output Enable signal.
BLS[3:0]	Output	Low-active Byte Lane Select signals.
WE	Output	Low-active Write Enable signal.
CS[3:0]	Output	Low-active Chip Select signals.

Table 6: External Memory Controller Pin Description

REGISTER DESCRIPTION

The external memory controller contains 4 registers as shown in Table 7.

Name	Description	Access	Reset Value (see Table 9)	Address
BCFG0	Configuration register for memory bank 0	Read/Write	0x0000 FBEF	0xFFE00000
BCFG1	Configuration register for memory bank 1	Read/Write	0x2000 FBEF	0xFFE00004
BCFG2	Configuration register for memory bank 2	Read/Write	0x1000 FBEF	0xFFE00008
BCFG3	Configuration register for memory bank 3	Read/Write	0x0000 FBEF	0xFFE0000C

Table 7: External Memory Controller Register Map

Each register selects the following options for its memory bank:

- The number of idle clock cycles inserted between between read and write accesses in this bank, and between an access in another bank and an access in this bank, to avoid bus contention between devices (1 to 17 clocks)
- the length of read accesses, except for subsequent reads from a burst ROM (3 to 35 clocks)
- the length of write accesses (3 to 19 clocks)
- whether the bank is write-protected
- whether the bank is 8, 16, or 32 bits wide

Bank Configuration Registers 0 - 3 (BCFG0-3 - 0xFFE00000-0C).

BCFG0-3	Name	Function	Reset Value
3:0	IDCY	This field controls the minimum number of “idle” CCLK cycles that the EMC maintains between read and write accesses in this bank, and between an access in another bank and an access in this bank, to avoid bus contention between devices. The number of idle CCLK cycles between such accesses is the value in this field plus 1.	1111
4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
9:5	WST1	This field controls the length of read accesses, except for subsequent reads from a burst ROM. The length of such read accesses, in CCLK cycles, is the value in this field plus 3.	11111
10	RBLE	This bit should be 0 for banks composed of byte-wide or non-byte-partitioned devices, so that the EMC drives the BLS3:0 lines High during read accesses. This bit should be 1 for banks composed of 16-bit and 32-bit wide devices that include byte select inputs, so that the EMC drives the BLS3:0 lines Low during read accesses.	0
15:11	WST2	For SRAM banks, this field controls the length of write accesses, which consist of: • one CCLK cycle of address setup with CS, BLS, and WE high, • (this value plus 1) CCLK cycles with address valid and CS, BLS, and WE low, and • one CCLK cycle with address valid, CS low, BLS and WE high. For burst ROM banks, this field controls the length of subsequent accesses, which are (this value plus 1) CCLK cycles long.	11111
16:23	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
24	BUSERR	The only known case in which this bit is set is if the EMC detects an AMBA request for more than 32 bits of data. The ARM7TDMI-S will not make such a request.	0
25	WPERR	This bit is set if software attempts to write to a bank that has the WP bit 1. Write a 1 to this bit to clear it.	0
26	WP	A 1 in this bit write-protects the bank.	0
27	BM	A 1 in this bit identifies a burst-ROM bank.	0
29:28	MW	This field controls the width of the data bus for this bank: 00=8 bit, 01=16 bit, 10=32 bit, 11=reserved	see Table 9
31:30	AT	Always write 00 to this field.	00

Table 8: Bank Configuration Registers 0-3 (BCFG0-3 - 0xFFE00000-0C)

The table below shows the state of BCFG0[29:28] after the Boot Loader has run. The hardware reset state of these bits is 10.

Bank	BOOT[1:0] during Reset	BCFG[29:28] Reset value	Memory Width
0	LL	00	8 bits
0	LH	01	16 bits
0	HL	10	32 bits
1	XX	10	32 bits
2	XX	01	16 bits
3	XX	00	8 bits

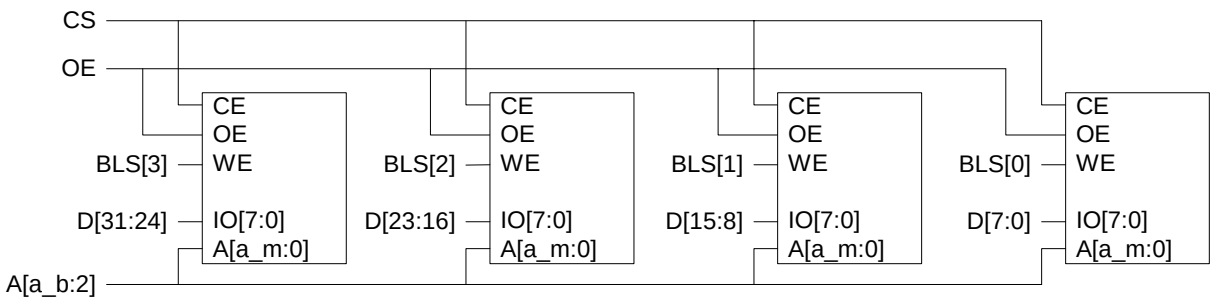
Table 9: Default memory widths at Reset

EXTERNAL MEMORY INTERFACE

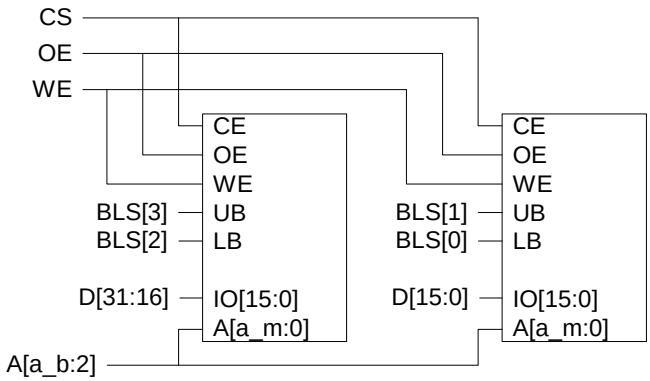
External memory interface depends on the bank width (32, 16 or 8 bit selected via MW bits in corresponding BCFG register). Furthermore, choice of the memory chip(s) will require an adequate setup of RBLE bit in BCFG register, too. RBLE = 0 in case of 8-bit based external memories, while memory chips capable of accepting 16 or 32 bit wide data will work with RBLE = 1.

If a memory bank is configured to be 32 bits wide, address lines A0 and A1 can be used as non-address lines. Memory bank configured to 16 bits wide will not require A0, while 8 bit wide memory bank will require address lines down to A0. Configuring A1 and/or A0 line(s) to provide address or non-address function is acomplished using bits 23 and 24 in Pin Function Select Register 2 (PINSEL2 register).

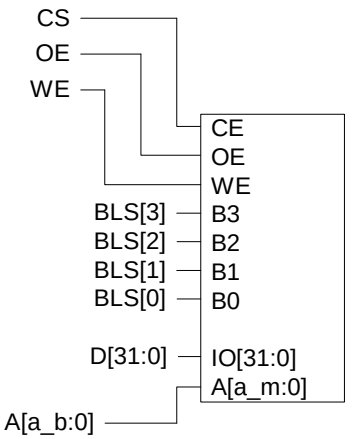
Symbol "a_b" in following figures refers to the highest order address line in the data bus. Symbol "a_m" refers to the highest order address line of the memory chip used in the external memory interface



a) 32 bit wide memory bank interfaced to 8 bit memory chips



b) 32 bit wide memory bank interfaced to 16 bit memory chips



c) 32 bit wide memory bank interfaced to 32 bit memory chip

Figure 7: 32 Bit Bank External Memory Interfaces

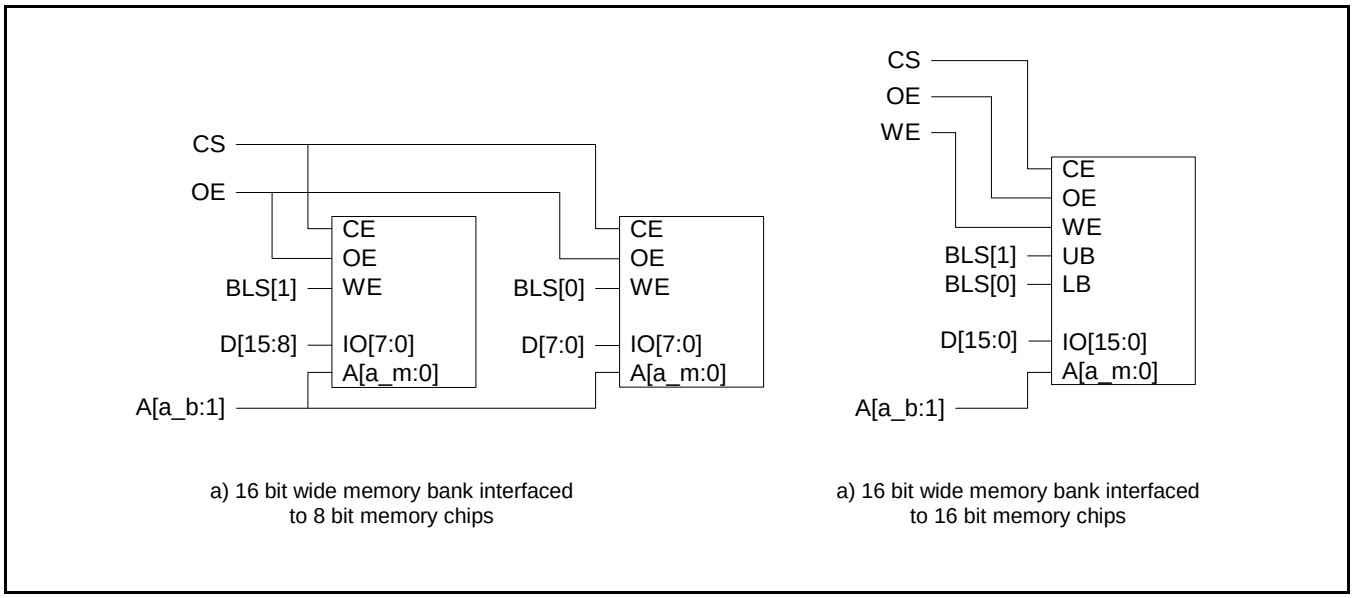


Figure 8: 16 Bit Bank External Memory Interfaces

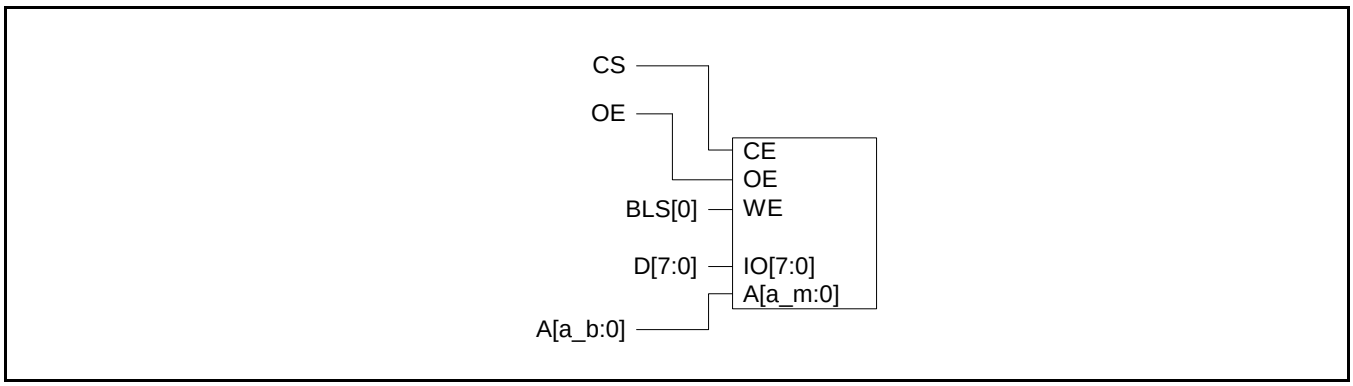


Figure 9: 8 Bit Bank External Memory Interface

TYPICAL BUS SEQUENCES

Following figures show typical external read and write access cycles. XCLK is the clock signal avalable on P3.23. While not necessary used by external memory, In these examples it is used to provide the time reference (XCLK and CCLK were set to have the same frequency).

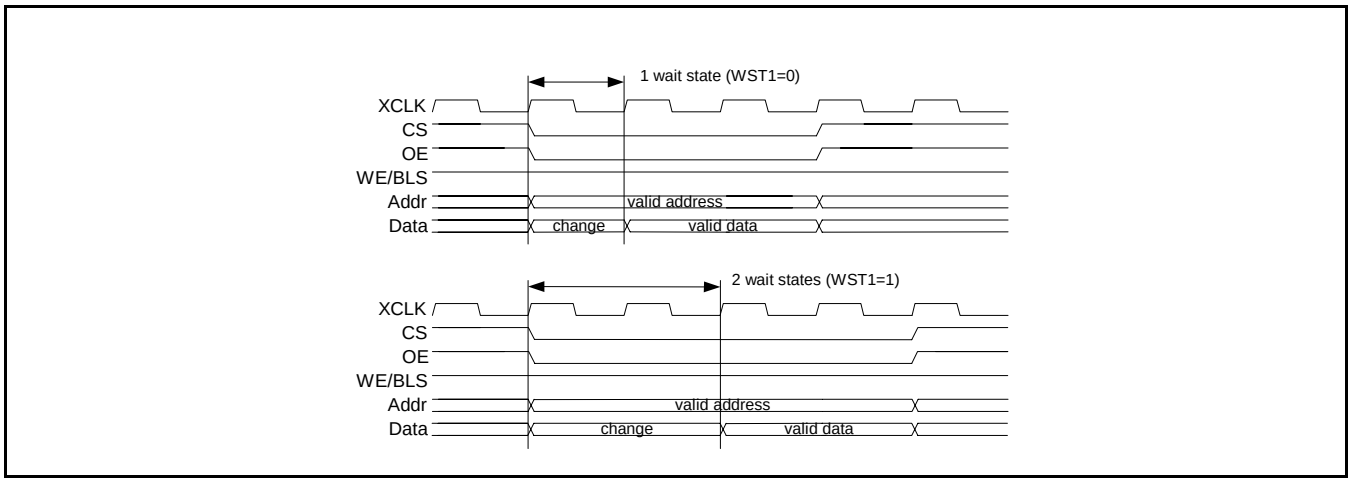


Figure 10: External memory read access (WST1=0 and WST1=1 examples)

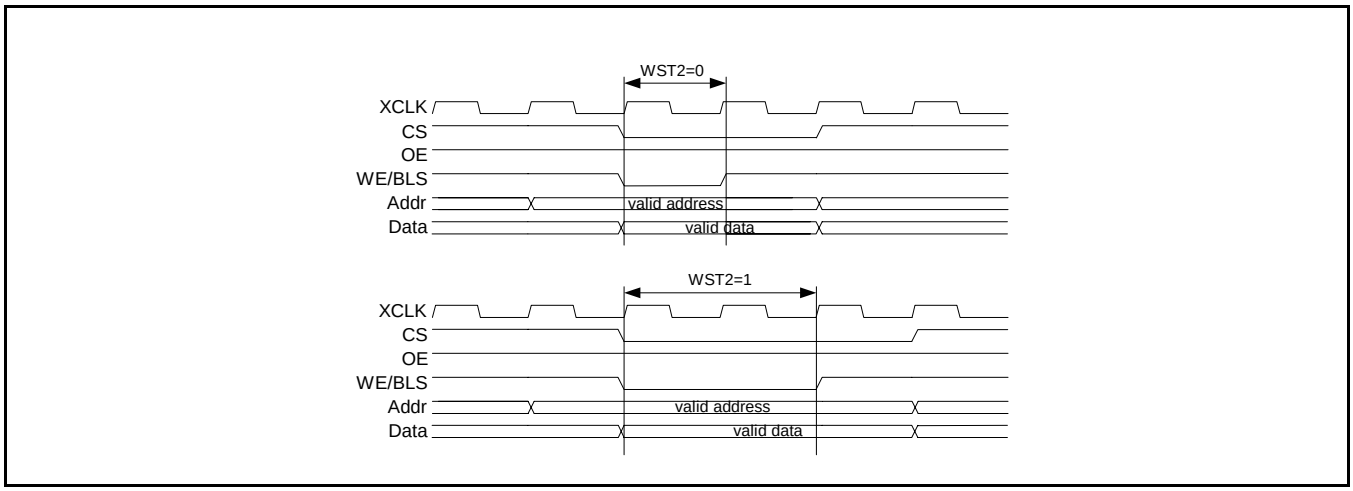


Figure 11: External memory write access (WST2=0 and WST2=1 examples)

Figure 10 and Figure 11 are showing typical read and write accesses to external memory. However, variations can be noticed in some particular cases.

For example, when the first read access to the memory bank that has just been selected is performed, CS and OE lines may become low one XCLK cycle earlier than it is shown in Figure 10.

Likewise, in a sequence of several consecutive write accesses to SRAM, the last write access will look like those shown in Figure 11. On the other hand, leading write cycles in that case will have data valid one cycle longer. Also, isloated write access will be identical to the one in Figure 11.

EXTERNAL MEMORY SELECTION

Based on the description of the EMC operation and external memory in general (appropriate read and write access times t_{AA} and t_{WRITE} respecitely), the following table can be constructed and used for external memory selection. t_{CYC} is the period of a single XCLK cycle (see Figure 10 and Figure 11). f_{max} is the maximum cclk frequency achievable in the system with selected external memory.

Table 10: External memory and system requirements

Access cycle	Max. frequency	WST setting (WST>=0; round up to integer)	Required memory access time
Standard Read	$f_{max} <= \frac{2 + WST1}{t_{RAM} + 20ns}$	$WST1 >= \frac{t_{RAM} + 20ns}{t_{CYC}} - 2$	$t_{RAM} <= t_{CYC}*(2+WST1) - 20ns$
Standard Write	$f_{max} <= \frac{1 + WST2}{t_{RAM} + 5ns}$	$WST2 >= \frac{t_{WRITE} - t_{CYC} + 5ns}{t_{CYC}}$	$t_{WRITE} <= t_{CYC}*(1+WST2) - 5ns$

4. SYSTEM CONTROL BLOCK

SUMMARY OF SYSTEM CONTROL BLOCK FUNCTIONS

The System Control Block includes several system features and control registers for a number of functions that are not related to specific peripheral devices. These include:

- Crystal Oscillator.
- External Interrupt Inputs.
- Memory Mapping Control.
- PLL.
- Power Control.
- Reset.
- VPB Divider.
- Wakeup Timer.

Each type of function has its own register(s) if any are required and unneeded bits are defined as reserved in order to allow future expansion. Unrelated functions never share the same register addresses.

PIN DESCRIPTION

Table 11 shows pins that are associated with System Control block functions.

Table 11: Pin summary

Pin name	Pin direction	Pin Description
X1	Input	**Crystal Oscillator Input-** Input to the oscillator and internal clock generator circuits.
X2	Output	**Crystal Oscillator Output-** Output from the oscillator amplifier.
EINT0	Input	**External Interrupt Input 0-** An active low general purpose interrupt input. This pin may be used to wake up the processor from Idle or Power down modes. Pins P0.1 and P0.16 can be selected to perform EINT0 function. LOW level on this pin immediately after reset is considered as an external hardware request to start the ISP command handler. More details on ISP and Flash memory can be found in "Flash Memory System and Programming" chapter.
EINT1	Input	**External Interrupt Input 1-** See the EINT0 description above. Pins P0.3 and P0.14 can be selected to perform EINT1 function.
EINT2	Input	**External Interrupt Input 2-** See the EINT0 description above. Pins P0.7 and P0.15 can be selected to perform EINT2 function.
EINT3	Input	**External Interrupt Input 3-** See the EINT0 description above. Pins P0.9, P0.20 and P0.30 can be selected to perform EINT3 function.
$\overline{\text{RESET}}$	Input	**External Reset input-** A low on this pin resets the chip, causing I/O ports and peripherals to take on their default states, and the processor to begin execution at address 0.

REGISTER DESCRIPTION

All registers, regardless of size, are on word address boundaries. Details of the registers appear in the description of each function.

Table 12: Summary of System Control Registers

Name	Description	Access	Reset Value*	Address
External Interrupts				
EXTINT	External Interrupt Flag Register.	R/W	0	0xE01FC140
EXTWAKE	External Interrupt Wakeup Register.	R/W	0	0xE01FC144
EXTMODE	External Interrupt Mode Register.	R/W	0	0xE01FC148
EXTPOLAR	External Interrupt Polarity Register.	R/W	0	0xE01FC14C
Memory Mapping Control				
MEMMAP	Memory Mapping Control.	R/W	0	0xE01FC040
Phase Locked Loop				
PLLCON	PLL Control Register.	R/W	0	0xE01FC080
PLLCFG	PLL Configuration Register.	R/W	0	0xE01FC084
PLLSTAT	PLL Status Register.	RO	0	0xE01FC088
PLLFEED	PLL Feed Register.	WO	NA	0xE01FC08C
Power Control				
PCON	Power Control Register.	R/W	0	0xE01FC0C0
PCONP	Power Control for Peripherals.	R/W	0x3BE	0xE01FC0C4
VPB Divider				
VPBDIV	VPB Divider Control.	R/W	0	0xE01FC100

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

CRYSTAL OSCILLATOR

While an input signal of 50-50 duty cycle within a frequency range from 1 MHz to 50 MHz can be used by LPC2119/2129/2194/2292/2294 if supplied to its input XTAL1 pin, this microcontroller's onboard oscillator circuit supports external crystals in the range of 1 MHz to 30 MHz only. If on-chip PLL system or boot-loader is used, input clock frequency is limited to exclusive range of 10 MHz to 25 MHz.

The oscillator output frequency is called F_{osc} and the ARM processor clock frequency is referred to as cclk for purposes of rate equations, etc. elsewhere in this document. F_{osc} and cclk are the same value unless the PLL is running and connected. Refer to the PLL description in this chapter for details and frequency limitations.

Onboard oscillator in LPC2119/2129/2194/2292/2294 can operate in one of two modes: slave mode and oscillation mode.

In slave mode the input clock signal should be coupled by means of a capacitor of 100 pF (Cc in Figure 12, drawing a), with an amplitude of at least 200 mVrms. X2 pin in this configuration can be left not connected. If slave mode is selected, F_{osc} signal of 50-50 duty cycle can range from 1 MHz to 50 MHz.

External components and models used in oscillation mode are shown in Figure 12, drawings b and c, and in Table 13. Since the feedback resistance is integrated on chip, only a crystal and the capacitances C_{X1} and C_{X2} need to be connected externally in case of fundamental mode oscillation (the fundamental frequency is represented by L, C_L and R_S). Capacitance Cp in Figure 12, drawing c, represents the parallel package capacitance and should not be larger than 7 pF. Parameters F_C, C_L, R_S and C_P are supplied by the crystal manufacturer.

Choosing an oscillation mode as an on-board oscillator mode of operation limits F_{osc} clock selection to 1 MHz to 30 MHz.

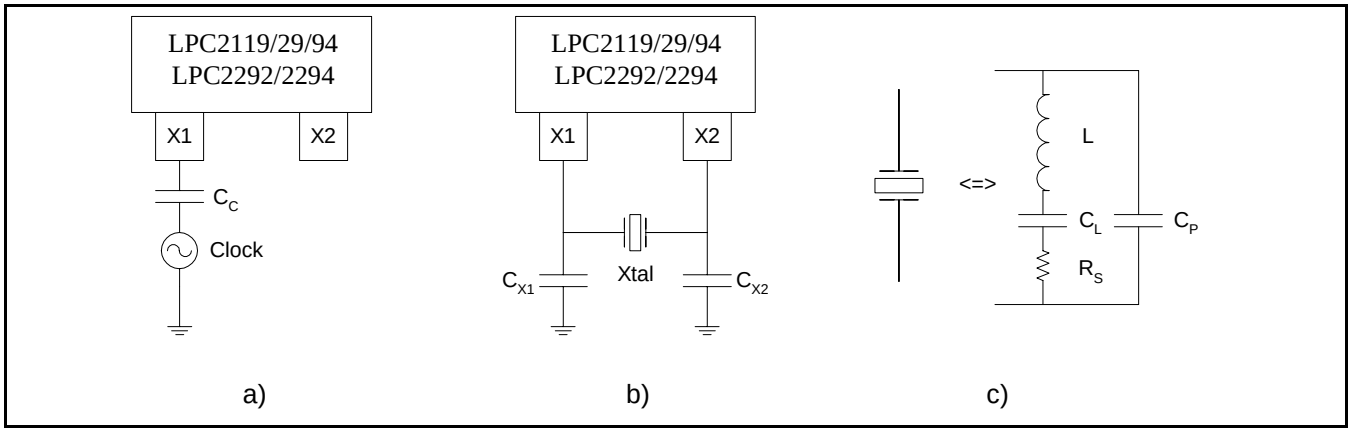


Figure 12: Oscillator modes and models: a) *slave mode* of operation, b) *oscillation mode* of operation, c) external crystal model used for $C_{X1/X2}$ evaluation

Table 13: Recommended values for $C_{X1/X2}$ in oscillation mode (crystal and external components parameters)

Fundamental Oscillation Frequency F_C	Crystal Load Capacitance C_L	Max. Crystal Series Resistence R_S	External Load Capacitors C_{X1}, C_{X2}
1 - 5 MHz	10 pF	n.a.	n.a.
	20 pF	n.a.	n.a.
	30 pF	< 300 Ω	58 pF, 58 pF

Table 13: Recommended values for $C_{X1/X2}$ in oscillation mode (crystal and external components parameters)

Fundamental Oscillation Frequency F_C	Crystal Load Capacitance C_L	Max. Crystal Series Resistence R_S	External Load Capacitors C_{X1}, C_{X2}
5 - 10 MHz	10 pF	< 300 Ω	18 pF, 18 pF
	20 pF	< 300 Ω	38 pF, 38 pF
	30 pF	< 300 Ω	58 pF, 58 pF
10 - 15 MHz	10 pF	< 300 Ω	18 pF, 18 pF
	20 pF	< 220 Ω	38 pF, 38 pF
	30 pF	< 140 Ω	58 pF, 58 pF
15 - 20 MHz	10 pF	< 220 Ω	18 pF, 18 pF
	20 pF	< 140 Ω	38 pF, 38 pF
	30 pF	< 80 Ω	58 pF, 58 pF
20 - 25 MHz	10 pF	< 160 Ω	18 pF, 18 pF
	20 pF	< 90 Ω	38 pF, 38 pF
	30 pF	< 50 Ω	58 pF, 58 pF
25 - 30 MHz	10 pF	<130 Ω	18 pF, 18 pF
	20 pF	<50 Ω	38 pF, 38 pF
	30 pF	n.a.	n.a.

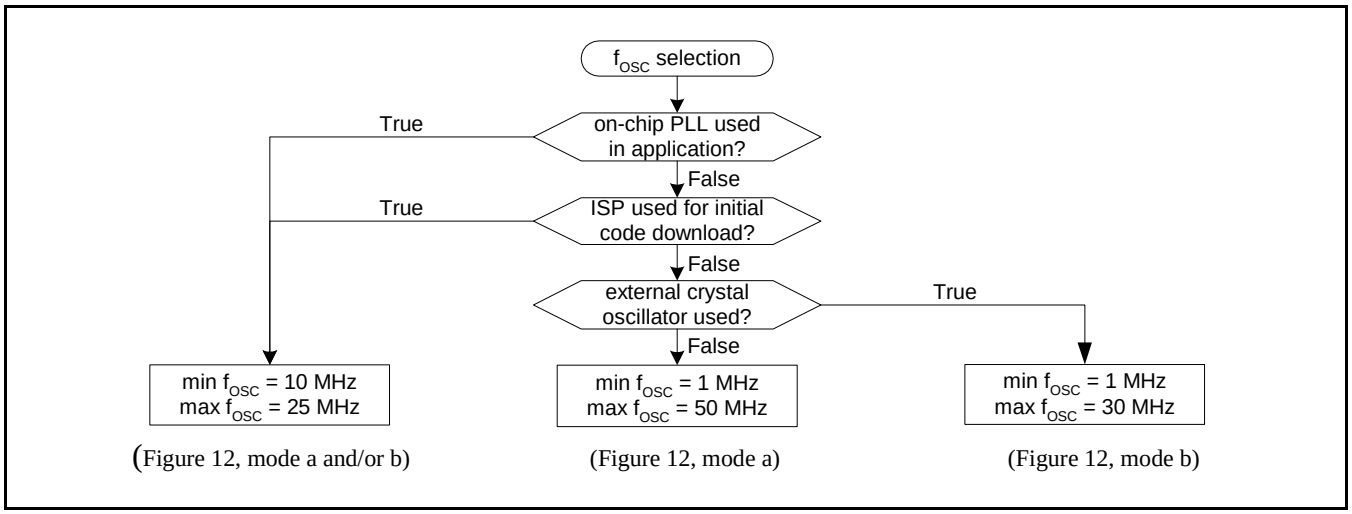


Figure 13: F_{OSC} selection algorithm

EXTERNAL INTERRUPT INPUTS

The LPC2119/2129/2194/2292/2294 includes four External Interrupt Inputs as selectable pin functions. The External Interrupt Inputs can optionally be used to wake up the processor from the Power Down mode.

Register Description

The external interrupt function has four registers associated with it. The EXTINT register contains the interrupt flags, and the EXTWAKEUP register contains bits that enable individual external interrupts to wake up the LPC2119/2129/2292/2294 from Power Down mode. The EXTMODE and EXTPOLAR registers specify the level and edge sensitivity parameters.

Table 14: External Interrupt Registers

Address	Name	Description	Access
0xE01FC140	EXTINT	The External Interrupt Flag Register contains interrupt flags for EINT0, EINT1, and EINT2. See Table 15.	R/W
0xE01FC144	EXTWAKE	The External Interrupt Wakeup Register contains three enable bits that control whether each external interrupt will cause the processor to wake up from Power Down mode. See Table 16.	R/W
0xE01FC148	EXTMODE	The External Interrupt Mode Register controls whether each pin is edge- or level-sensitive.	R/W
0xE01FC14C	EXTPOLAR	The External Interrupt Polarity Register controls which level or edge on each pin will cause an interrupt.	R/W

External Interrupt Flag Register (EXTINT - 0xE01FC140)

When a pin is selected for its external interrupt function, the level or edge on that pin selected by its bits in the EXTPOLAR and EXTMODE registers will set its interrupt flag in this register. This asserts the corresponding interrupt request to the VIC, which will cause an intrerrupt if interrupts from the pin are enabled.

Writing ones to bits EINT0 through EINT3 in EXTINT register clears the corresponding bits. In level-sensitive mode this action is efficacious only when the pin is in its innactive state.

Table 15: External Interrupt Flag Register (EXTINT - 0xE01FC140)

EXTINT	Function	Description	Reset Value
0	EINT0	In level-sensitive mode, this bit is set if the EINT0 function is selected for its pin, and the pin is in its active state. In edge-sensitive mode, this bit is set if the EINT0 function is selected for its pin, and the selected edge occurs on the pin. Up to two pins can be selected to perform EINT0 function (see P0.1 and P0.16 description in "Pin Configuration" chapter.) This bit is cleared by writing a one to it, except in level sensitive mode when the pin is in its active state.	0
1	EINT1	In level-sensitive mode, this bit is set if the EINT1 function is selected for its pin, and the pin is in its active state. In edge-sensitive mode, this bit is set if the EINT1 function is selected for its pin, and the selected edge occurs on the pin. Up to two pins can be selected to perform EINT1 function (see P0.3 and P0.14 description in "Pin Configuration" chapter.) This bit is cleared by writing a one to it, except in level sensitive mode when the pin is in its active state.	0
2	EINT2	In level-sensitive mode, this bit is set if the EINT2 function is selected for its pin, and the pin is in its active state. In edge-sensitive mode, this bit is set if the EINT2 function is selected for its pin, and the selected edge occurs on the pin. Up to two pins can be selected to perform EINT2 function (see P0.7 and P0.15 description in "Pin Configuration" chapter.) This bit is cleared by writing a one to it, except in level sensitive mode when the pin is in its active state.	0
3	EINT3	In level-sensitive mode, this bit is set if the EINT3 function is selected for its pin, and the pin is in its active state. In edge-sensitive mode, this bit is set if the EINT3 function is selected for its pin, and the selected edge occurs on the pin. Up to three pins can be selected to perform EINT3 function (see P0.9, P0.20 and P0.30 description in "Pin Configuration" chapter.) This bit is cleared by writing a one to it, except in level sensitive mode when the pin is in its active state.	0
7:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

External Interrupt Wakeup Register (EXTWAKE - 0xE01FC144)

Enable bits in the EXTWAKE register allow the external interrupts to wake up the processor if it is in Power Down mode. The related EINTn function must be mapped to the pin in order for the wakeup process to take place. It is not necessary for the interrupt to be enabled in the Vectored Interrupt Controller for a wakeup to take place. This arrangement allows additional capabilities, such as having an external interrupt input wake up the processor from Power Down mode without causing an interrupt (simply resuming operation), or allowing an interrupt to be enabled during Power Down without waking the processor up if it is asserted (eliminating the need to disable the interrupt if the wakeup feature is not desirable in the application).

Table 16: External Interrupt Wakeup Register (EXTWAKE - 0xE01FC144)

EXTWAKE	Function	Description	Reset Value
0	EXTWAKE0	When one, assertion of $\overline{\text{EINT0}}$ will wake up the processor from Power Down mode.	0
1	EXTWAKE1	When one, assertion of $\overline{\text{EINT1}}$ will wake up the processor from Power Down mode.	0
2	EXTWAKE2	When one, assertion of $\overline{\text{EINT2}}$ will wake up the processor from Power Down mode.	0
3	EXTWAKE3	When one, assertion of $\overline{\text{EINT3}}$ will wake up the processor from Power Down mode.	0
7:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

External Interrupt Mode Register (EXTMODE - 0xE01FC148)

The bits in this register select whether each EINT pin is level- or edge-sensitive. Only pins that are selected for the EINT function (chapter Pin Connect Block on page 126) and enabled via the VICIntEnable register (chapter Vectored Interrupt Controller (VIC) on page 96) can cause interrupts from the External Interrupt function (though of course pins selected for) other functions may cause interrupts from those functions).

Note: Software should only change a bit in this register when its interrupt is disabled in VICIntEnable, and should write the corresponding 1 to EXTINT before re-enabling the interrupt, to clear the EXTINT bit that could be set by changing the mode.

Table 17: External Interrupt Mode Register (EXTMODE - 0xE01FC148)

EXTMODE	Function	Description	Reset Value
0	EXTMODE0	When 0, level-sensitivity is selected for EINT0. When 1, EINT0 is edge-sensitive.	0
1	EXTMODE1	When 0, level-sensitivity is selected for EINT1. When 1, EINT1 is edge-sensitive.	0
2	EXTMODE2	When 0, level-sensitivity is selected for EINT2. When 1, EINT2 is edge-sensitive.	0
3	EXTMODE3	When 0, level-sensitivity is selected for EINT3. When 1, EINT3 is edge-sensitive.	0
7:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

External Interrupt Polarity Register (EXTPOLAR - 0xE01FC14C)

In level-sensitive mode, the bits in this register select whether the corresponding pin is high- or low-active. In edge-sensitive mode, they select whether the pin is rising- or falling-edge sensitive. Only pins that are selected for the EINT function (chapter Pin Connect Block on page 126) and enabled in the VICIntEnable register (chapter Vectored Interrupt Controller (VIC) on page 96) can cause interrupts from the External Interrupt function (though of course pins selected for other functions may cause interrupts from those functions).

Note: Software should only change a bit in this register when its interrupt is disabled in VICIntEnable, and should write the corresponding 1 to EXTINT before re-enabling the interrupt, to clear the EXTINT bit that could be set by changing the polarity.

Table 18: External Interrupt Polarity Register (EXTPOLAR - 0xE01FC14C)

EXTPOLAR	Function	Description	Reset Value
0	EXTPOLAR0	When 0, EINT0 is low-active or falling-edge sensitive (depending on EXTMODE0). When 1, EINT0 is high-active or rising-edge sensitive (depending on EXTMODE0).	0
1	EXTPOLAR1	When 0, EINT1 is low-active or falling-edge sensitive (depending on EXTMODE1). When 1, EINT1 is high-active or rising-edge sensitive (depending on EXTMODE1).	0
2	EXTPOLAR2	When 0, EINT2 is low-active or falling-edge sensitive (depending on EXTMODE2). When 1, EINT2 is high-active or rising-edge sensitive (depending on EXTMODE2).	0
3	EXTPOLAR3	When 0, EINT3 is low-active or falling-edge sensitive (depending on EXTMODE3). When 1, EINT3 is high-active or rising-edge sensitive (depending on EXTMODE3).	0
7:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Multiple External Interrupt Pins

Software can select multiple pins for each of EINT3:0 in the Pin Select registers, which are described in chapter Pin Connect Block on page 126. The external interrupt logic for each of EINT3:0 receives the state of all of its associated pins from the pins' receivers, along with signals that indicate whether each pin is selected for the EINT function. The external interrupt logic handles the case when more than one pin is so selected, differently according to the state of its Mode and Polarity bits:

- In Low-Active Level Sensitive mode, the states of all pins selected for EINT functionality are digitally combined using a positive logic AND gate.
- In High-Active Level Sensitive mode, the states of all pins selected for EINT functionality are digitally combined using a positive logic OR gate.
- In Edge Sensitive mode, regardless of polarity, the pin with the lowest GPIO port number is used. (Selecting multiple EINT pins in edge-sensitive mode could be considered a programming error.)

The signal derived by this logic is the EINTi signal in the following logic schematic (Figure 14).

When more than one EINT pin is logically ORed, the interrupt service routine can read the states of the pins from GPIO port using IO0PIN0 and IO1PIN registers, to determine which pin(s) caused the interrupt.

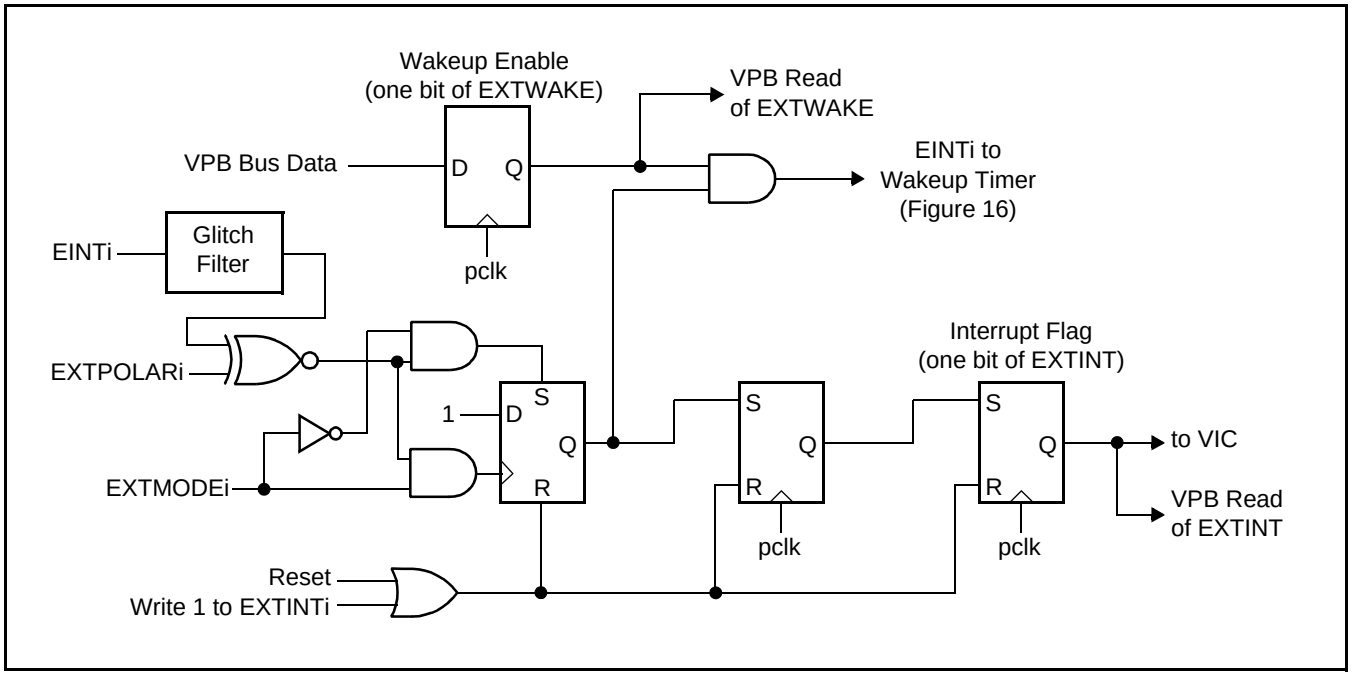


Figure 14: External Interrupt Logic

MEMORY MAPPING CONTROL

The Memory Mapping Control alters the mapping of the interrupt vectors that appear beginning at address 0x00000000. This allows code running in different memory spaces to have control of the interrupts.

Memory Mapping Control Register (MEMMAP - 0xE01FC040)

Table 19: MEMMAP Register

Address	Name	Description	Access
0xE01FC040	MEMMAP	Memory mapping control. Selects whether the ARM interrupt vectors are read from the Flash Boot Block, User Flash or RAM.	R/W

Table 20: Memory Mapping Control Register (MEMMAP - 0xE01FC040)

MEMMAP	Function	Description	Reset Value*
1:0	MAP1:0	00: Boot Loader Mode. Interrupt vectors are re-mapped to Boot Block. 01: User Flash Mode. Interrupt vectors are not re-mapped and reside in Flash. 10: User RAM Mode. Interrupt vectors are re-mapped to Static RAM. 11: User External memory Mode. Interrupt vectors are re-mapped to external memory. This mode is available in L2292/2294 only and must not be specified when LPC2119/2129/2194 are used. **Warning:** Improper setting of this value may result in incorrect operation of the device.	0
7:2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

*: The hardware reset value of the MAP bits is 00 for LPC2119/2129/2194/2292/2294 parts. The apparent reset value that the user will see will be altered by the Boot Loader code, which always runs initially at reset. User documentation will reflect this difference.

Memory Mapping Control Usage Notes

Memory Mapping Control simply selects one out of three available sources of data (sets of 64 bytes each) necessary for handling ARM exceptions (interrupts).

For example, whenever a Software Interrupt request is generated, ARM core will always fetch 32-bit data "residing" on 0x0000 0008 (see Table 3, “ARM Exception Vector Locations,” on page 52). This means that when MEMMAP[1:0]=10 (User RAM Mode), read/fetch from 0x0000 0008 will provide data stored in 0x4000 0008. If MEMMAP[1:0]=01 (User Flash Mode), read/fetch from 0x0000 0008 will provide data stored in on-chip Flash location 0x0000 0008. In case of MEMMAP[1:0]=00 (Boot Loader Mode), read/fetch from 0x0000 0008 will provide data availble also at 0x7FFF E008 (Boot Block remapped from on-chip Flash memory).

PLL (PHASE LOCKED LOOP)

The PLL accepts an input clock frequency in the range of 10 MHz to 25 MHz only. The input frequency is multiplied up into the cclk with the range of 10 MHz to 60 MHz using a Current Controlled Oscillator (CCO). The multiplier can be an integer value from 1 to 32 (in practice, the multiplier value cannot be higher than 6 on the LPC2119/2129/2194/2292/2294 due to the upper frequency limit of the CPU). The CCO operates in the range of 156 MHz to 320 MHz, so there is an additional divider in the loop to keep the CCO within its frequency range while the PLL is providing the desired output frequency. The output divider may be set to divide by 2, 4, 8, or 16 to produce the output clock. Since the minimum output divider value is 2, it is insured that the PLL output has a 50% duty cycle. A block diagram of the PLL is shown in Figure 15.

PLL activation is controlled via the PLLCON register. The PLL multiplier and divider values are controlled by the PLLCFG register. These two registers are protected in order to prevent accidental alteration of PLL parameters or deactivation of the PLL. Since all chip operations, including the Watchdog Timer, are dependent on the PLL when it is providing the chip clock, accidental changes to the PLL setup could result in unexpected behavior of the microcontroller. The protection is accomplished by a feed sequence similar to that of the Watchdog Timer. Details are provided in the description of the PLLFEED register.

The PLL is turned off and bypassed following a chip Reset and when by entering power Down mode. PLL is enabled by software only. The program must configure and activate the PLL, wait for the PLL to Lock, then connect to the PLL as a clock source.

Register Description

The PLL is controlled by the registers shown in Table 21. More detailed descriptions follow.

Warning: Improper setting of PLL values may result in incorrect operation of the device.

Table 21: PLL Registers

Address	Name	Description	Access
0xE01FC080	PLLCON	PLL Control Register. Holding register for updating PLL control bits. Values written to this register do not take effect until a valid PLL feed sequence has taken place.	R/W
0xE01FC084	PLLCFG	PLL Configuration Register. Holding register for updating PLL configuration values. Values written to this register do not take effect until a valid PLL feed sequence has taken place.	R/W
0xE01FC088	PLLSTAT	PLL Status Register. Read-back register for PLL control and configuration information. If PLLCON or PLLCFG have been written to, but a PLL feed sequence has not yet occurred, they will not reflect the current PLL state. Reading this register provides the actual values controlling the PLL, as well as the status of the PLL.	RO
0xE01FC08C	PLLFEED	PLL Feed Register. This register enables loading of the PLL control and configuration information from the PLLCON and PLLCFG registers into the shadow registers that actually affect PLL operation.	WO

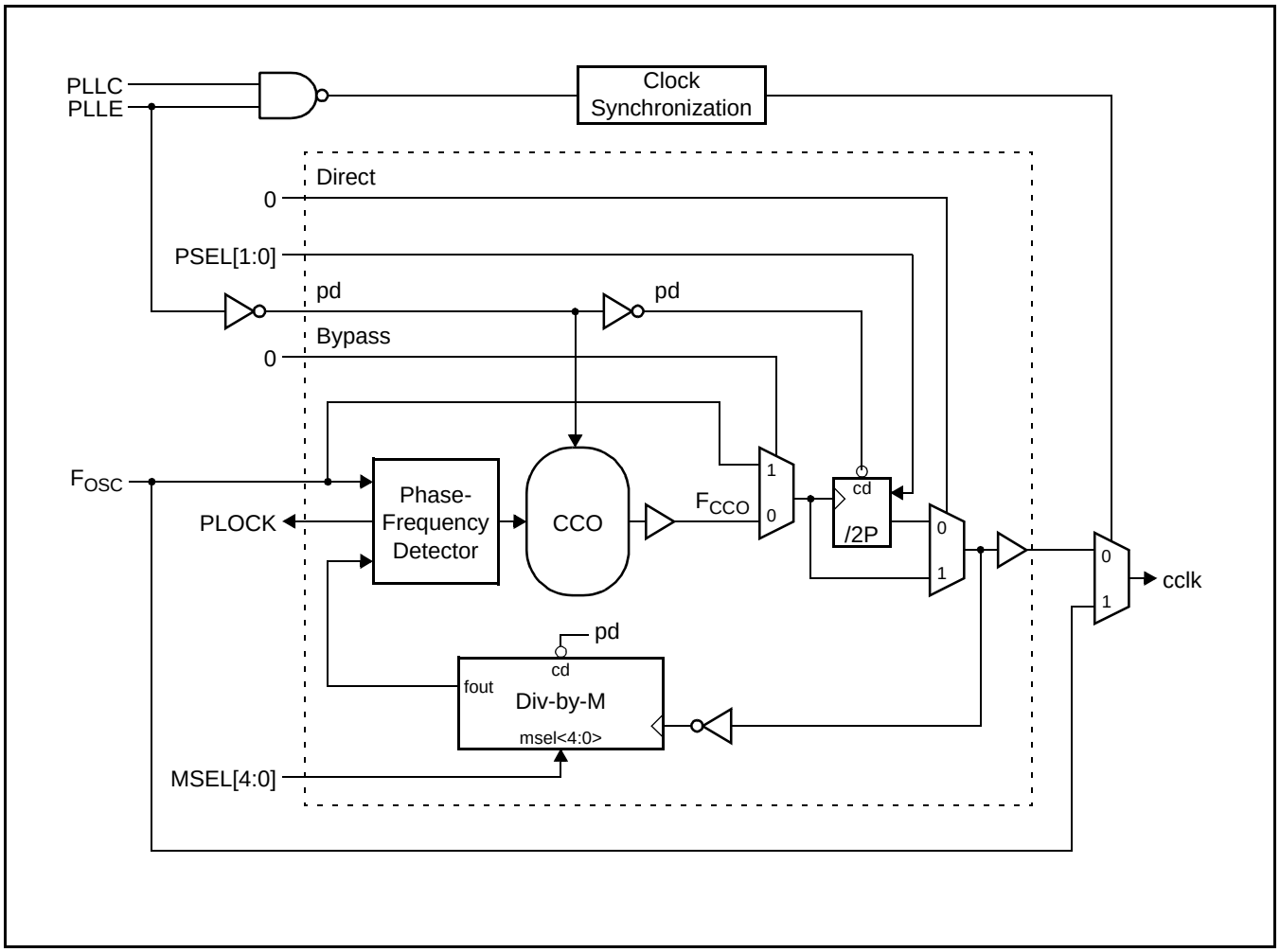


Figure 15: PLL Block Diagram

PLL Control Register (PLLCON - 0xE01FC080)

The PLLCON register contains the bits that enable and connect the PLL. Enabling the PLL allows it to attempt to lock to the current settings of the multiplier and divider values. Connecting the PLL causes the processor and all chip functions to run from the PLL output clock. Changes to the PLLCON register do not take effect until a correct PLL feed sequence has been given (see PLL Feed Register (PLLFEED - 0xE01FC08C) description).

Table 22: PLL Control Register (PLLCON - 0xE01FC080)

PLLCON	Function	Description	Reset Value
0	PLLE	PLL Enable. When one, and after a valid PLL feed, this bit will activate the PLL and allow it to lock to the requested frequency. See PLLSTAT register, Table 24.	0
1	PLLC	PLL Connect. When PLLC and PLLE are both set to one, and after a valid PLL feed, connects the PLL as the clock source for the LPC2119/2129/2194/2292/2294. Otherwise, the oscillator clock is used directly by the LPC2119/2129/2194/2292/2294. See PLLSTAT register, Table 24.	0
7:2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

The PLL must be set up, enabled, and Lock established before it may be used as a clock source. When switching from the oscillator clock to the PLL output or vice versa, internal circuitry synchronizes the operation in order to ensure that glitches are not generated. Hardware does not insure that the PLL is locked before it is connected or automatically disconnect the PLL if lock is lost during operation. In the event of loss of PLL lock, it is likely that the oscillator clock has become unstable and disconnecting the PLL will not remedy the situation.

PLL Configuration Register (PLLCFG - 0xE01FC084)

The PLLCFG register contains the PLL multiplier and divider values. Changes to the PLLCFG register do not take effect until a correct PLL feed sequence has been given (see PLL Feed Register (PLLFEED - 0xE01FC08C) description). Calculations for the PLL frequency, and multiplier and divider values are found in the PLL Frequency Calculation section.

Table 23: PLL Configuration Register (PLLCFG - 0xE01FC084)

PLLCFG	Function	Description	Reset Value
4:0	MSEL4:0	PLL Multiplier value. Supplies the value "M" in the PLL frequency calculations. **Note:** For details on selecting the right value for MSEL4:0 see section "PLL Frequency Calculation" on page 79.	0
6:5	PSEL1:0	PLL Divider value. Supplies the value "P" in the PLL frequency calculations. **Note:** For details on selecting the right value for PSEL1:0 see section "PLL Frequency Calculation" on page 79.	0
7	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

PLL Status Register (PLLSTAT - 0xE01FC088)

The read-only PLLSTAT register provides the actual PLL parameters that are in effect at the time it is read, as well as the PLL status. PLLSTAT may disagree with values found in PLLCON and PLLCFG because changes to those registers do not take effect until a proper PLL feed has occurred (see PLL Feed Register (PLLFEED - 0xE01FC08C) description).

Table 24: PLL Status Register (PLLSTAT - 0xE01FC088)

PLLSTAT	Function	Description	Reset Value
4:0	MSEL4:0	Read-back for the PLL Multiplier value. This is the value currently used by the PLL.	0
6:5	PSEL1:0	Read-back for the PLL Divider value. This is the value currently used by the PLL.	0
7	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
8	PLLE	Read-back for the PLL Enable bit. When one, the PLL is currently activated. When zero, the PLL is turned off. This bit is automatically cleared when Power Down mode is activated.	0
9	PLLC	Read-back for the PLL Connect bit. When PLLC and PLLE are both one, the PLL is connected as the clock source for the LPC2119/2129/2194/2292/2294. When either PLLC or PLLE is zero, the PLL is bypassed and the oscillator clock is used directly by the LPC2119/2129/2194/2292/2294. This bit is automatically cleared when Power Down mode is activated.	0
10	PLOCK	Reflects the PLL Lock status. When zero, the PLL is not locked. When one, the PLL is locked onto the requested frequency.	0
15:11	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

PLL Interrupt

The PLOCK bit in the PLLSTAT register is connected to the interrupt controller. This allows for software to turn on the PLL and continue with other functions without having to wait for the PLL to achieve lock. When the interrupt occurs (PLOCK = 1), the PLL may be connected, and the interrupt disabled.

PLL Modes

The combinations of PLLE and PLLC are shown in Table 25.

Table 25: PLL Control Bit Combinations

PLLC	PLLE	PLL Function
0	0	PLL is turned off and disconnected. The system runs from the unmodified clock input.
0	1	The PLL is active, but not yet connected. The PLL can be connected after PLOCK is asserted.
1	0	Same as 0 0 combination. This prevents the possibility of the PLL being connected without also being enabled.
1	1	The PLL is active and has been connected as the system clock source.

PLL Feed Register (PLLFEED - 0xE01FC08C)

A correct feed sequence must be written to the PLLFEED register in order for changes to the PLLCON and PLLCFG registers to take effect. The feed sequence is:

1. Write the value 0xAA to PLLFEED
2. Write the value 0x55 to PLLFEED.

The two writes must be in the correct sequence, and must be consecutive VPB bus cycles. The latter requirement implies that interrupts must be disabled for the duration of the PLL feed operation. If either of the feed values is incorrect, or one of the previously mentioned conditions is not met, any changes to the PLLCON or PLLCFG register will not become effective.

Table 26: PLL Feed Register (PLLFEED - 0xE01FC08C)

PLLFEED	Function	Description	Reset Value
7:0	PLLFEED	The PLL feed sequence must be written to this register in order for PLL configuration and control register changes to take effect.	undefined

PLL and Power Down Mode

Power Down mode automatically turns off and disconnects the PLL. Wakeup from Power Down mode does not automatically restore the PLL settings, this must be done in software. Typically, a routine to activate the PLL, wait for lock, and then connect the PLL can be called at the beginning of any interrupt service routine that might be called due to the wakeup. It is important not to attempt to restart the PLL by simply feeding it when execution resumes after a wakeup from Power Down mode. This would enable and connect the PLL at the same time, before PLL lock is established.

PLL Frequency Calculation

The PLL equations use the following parameters:

F_{OSC} the frequency from the crystal oscillator
F_{CCO} the frequency of the PLL current controlled oscillator
cclk the PLL output frequency (also the processor clock frequency)
M PLL Multiplier value from the MSEL bits in the PLLCFG register
P PLL Divider value from the PSEL bits in the PLLCFG register

The PLL output frequency (when the PLL is both active and connected) is given by:

$$cclk = M * F_{osc} \quad \text{or} \quad cclk = \frac{F_{cco}}{2 * P}$$

The CCO frequency can be computed as:

$$F_{cco} = cclk * 2 * P \quad \text{or} \quad F_{cco} = F_{osc} * M * 2 * P$$

The PLL inputs and settings must meet the following:

- F_{osc} is in the range of 10 MHz to 25 MHz.
- cclk is in the range of 10 MHz to F_{max} (the maximum allowed frequency for the LPC2119/2129/2194/2292/2294).
- F_{cco} is in the range of 156 MHz to 320 MHz.

Procedure for Determining PLL Settings

If a particular application uses the PLL, its configuration may be determined as follows:

1. Choose the desired processor operating frequency (cclk). This may be based on processor throughput requirements, need to support a specific set of UART baud rates, etc. Bear in mind that peripheral devices may be running from a lower clock than the processor (see the VPB Divider description in this chapter).
2. Choose an oscillator frequency (F_{osc}). cclk must be the whole (non-fractional) multiple of F_{osc}.
3. Calculate the value of M to configure the MSEL bits. M = cclk / F_{osc}. M must be in the range of 1 to 32. The value written to the MSEL bits in PLLCFG is M - 1 (see Table 28).
4. Find a value for P to configure the PSEL bits, such that F_{cco} is within its defined frequency limits. F_{cco} is calculated using the equation given above. P must have one of the values 1, 2, 4, or 8. The value written to the PSEL bits in PLLCFG is 00 for P = 1; 01 for P = 2; 10 for P = 4; 11 for P = 8 (see Table 27).

Table 27: PLL Divider Values

PSEL Bits (PLLCFG bits 6:5)	Value of P
00	1
01	2
10	4
11	8

Table 28: PLL Multiplier Values

MSEL Bits (PLLCFG bits 4:0)	Value of M
00000	1
00001	2
00010	3
00011	4
...	...
11110	31
11111	32

PLL Example

System design asks for F_{osc}= 10 MHz and requires cclk = 60 MHz.

Based on these specifications, M = cclk / F_{osc} = 60 MHz / 10 MHz = 6. Consequenty, M-1 = 5 will be written as PLLCFG 4:0.

Value for P can be derived from P = F_{cco} / (cclk * 2), using condition that F_{cco} must be in range of 156 MHz to 320 MHz. Assuming the lowest allowed frequency for F_{cco} = 156 MHz, P = 156 MHz / (2*60 MHz) = 1.3. The highest F_{cco} frequency criteria produces P = 2.67. The only solution for P that satisfies both of these requirements and is listed in Table 27 is P = 2. Therefore, PLLCFG 6:5 = 1 will be used.

POWER CONTROL

The LPC2119/2129/2194/2292/2294 supports two reduced power modes: Idle mode and Power Down mode. In Idle mode, execution of instructions is suspended until either a Reset or interrupt occurs. Peripheral functions continue operation during Idle mode and may generate interrupts to cause the processor to resume execution. Idle mode eliminates power used by the processor itself, memory systems and related controllers, and internal buses.

In Power Down mode, the oscillator is shut down and the chip receives no internal clocks. The processor state and registers, peripheral registers, and internal SRAM values are preserved throughout Power Down mode and the logic levels of chip pins remain static. The Power Down mode can be terminated and normal operation resumed by either a Reset or certain specific interrupts that are able to function without clocks. Since all dynamic operation of the chip is suspended, Power Down mode reduces chip power consumption to nearly zero.

Entry to Power Down and Idle modes must be coordinated with program execution. Wakeup from Power Down or Idle modes via an interrupt resumes program execution in such a way that no instructions are lost, incomplete, or repeated. Wake up from Power Down mode is discussed further in the description of the Wakeup Timer later in this chapter.

A Power Control for Peripherals feature allows individual peripherals to be turned off if they are not needed in the application, resulting in additional power savings.

Register Description

The Power Control function contains two registers, as shown in Table 29. More detailed descriptions follow.

Table 29: Power Control Registers

Address	Name	Description	Access
0xE01FC0C0	PCON	Power Control Register. This register contains control bits that enable the two reduced power operating modes of the LPC2119/2129/2194/2292/2294. See Table 30.	R/W
0xE01FC0C4	PCONP	Power Control for Peripherals Register. This register contains control bits that enable and disable individual peripheral functions, Allowing elimination of power consumption by peripherals that are not needed.	R/W

Power Control Register (PCON - 0xE01FC0C0)

The PCON register contains two bits. Writing a one to the corresponding bit causes entry to either the Power Down or Idle mode. If both bits are set, Power Down mode is entered.

Table 30: Power Control Register (PCON - 0xE01FC0C0)

PCON	Function	Description	Reset Value
0	IDL	Idle mode - when 1, this bit causes the processor clock to be stopped, while on-chip peripherals remain active. Any enabled interrupt from a peripheral or an external interrupt source will cause the processor to resume execution.	0
1	PD	Power Down mode - when 1, this bit causes the oscillator and all on-chip clocks to be stopped. A wakeup condition from an external interrupt can cause the oscillator to re-start, the PD bit to be cleared, and the processor to resume execution.	0
7:2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Power Control for Peripherals Register (PCONP - 0xE01FC0C4)

The PCONP register allows turning off selected peripheral functions for the purpose of saving power. A few peripheral functions cannot be turned off (i.e. the Watchdog timer, GPIO, the Pin Connect block, and the System Control block). Each bit in PCONP controls one of the peripherals. The bit numbers correspond to the related peripheral number as shown in the VPB peripheral map in the LPC2119/2129/2292/2294 Memory Addressing section.

Table 31: Power Control for Peripherals Register for LPC2119/2129/2292 (PCONP - 0xE01FC0C4)

PCONP	Function	Description	Reset Value
0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
1	PCTIM0	When 1, TIMER0 is enabled. When 0, TIMER0 is disabled to conserve power.	1
2	PCTIM1	When 1, TIMER1 is enabled. When 0, TIMER1 is disabled to conserve power.	1
3	PCURT0	When 1, UART0 is enabled. When 0, UART0 is disabled to conserve power.	1
4	PCURT1	When 1, UART1 is enabled. When 0, UART1 is disabled to conserve power.	1
5	PCPWM0	When 1, PWM0 is enabled. When 0, PWM0 is disabled to conserve power.	1
6	Reserved	User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
7	PCI2C	When 1, the I^2C interface is enabled. When 0, the I^2C interface is disabled to conserve power.	1
8	PCSPI0	When 1, the SPI0 interface is enabled. When 0, the SPI0 is disabled to conserve power.	1
9	PCRTC	When 1, the RTC is enabled. When 0, the RTC is disabled to conserve power.	1
10	PCSPI1	When 1, the SPI1 interface is enabled. When 0, the SPI1 is disabled to conserve power.	1
11	Reserved	User software should write 0 here to reduce power consumption.	1

Table 31: Power Control for Peripherals Register for LPC2119/2129/2292 (PCONP - 0xE01FC0C4)

PCONP	Function	Description	Reset Value
12	PCAD	When 1, the A/D converter is enabled. When 0, the A/D is disabled to conserve power.	1
13	PCCAN1	When 1, CAN Controller 1 is enabled. When 0, it is disabled to save power. **Note:** the Acceptance Filter is enabled if any of CAN Controllers 1-2 is enabled.	1
14	PCCAN2	When 1, CAN Controller 2 is enabled. When 0, it is disabled to save power.	1
31:15	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Table 32: Power Control for Peripherals Register for LPC2194/2294 (PCONP - 0xE01FC0C4)

PCONP	Function	Description	Reset Value
0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
1	PCTIM0	When 1, TIMER0 is enabled. When 0, TIMER0 is disabled to conserve power.	1
2	PCTIM1	When 1, TIMER1 is enabled. When 0, TIMER1 is disabled to conserve power.	1
3	PCURT0	When 1, UART0 is enabled. When 0, UART0 is disabled to conserve power.	1
4	PCURT1	When 1, UART1 is enabled. When 0, UART1 is disabled to conserve power.	1
5	PCPWM0	When 1, PWM0 is enabled. When 0, PWM0 is disabled to conserve power.	1
6	Reserved	User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
7	PCI2C	When 1, the I^2C interface is enabled. When 0, the I^2C interface is disabled to conserve power.	1
8	PCSPI0	When 1, the SPI0 interface is enabled. When 0, the SPI0 is disabled to conserve power.	1
9	PCRTC	When 1, the RTC is enabled. When 0, the RTC is disabled to conserve power.	1
10	PCSPI1	When 1, the SPI1 interface is enabled. When 0, the SPI1 is disabled to conserve power.	1
11	PCEMC	When 1, the External Memory Controller is enabled. When 0, the EMC is disabled to conserve power.	1
12	PCAD	When 1, the A/D converter is enabled. When 0, the A/D is disabled to conserve power.	1
13	PCCAN1	When 1, CAN Controller 1 is enabled. When 0, it is disabled to save power.	1
14	PCCAN2	When 1, CAN Controller 2 is enabled. When 0, it is disabled to save power.	1
15	PCCAN3	When 1, CAN Controller 3 is enabled. When 0, it is disabled to save power.	1
16	PCCAN4	When 1, CAN Controller 4 is enabled. When 0, it is disabled to save power.	1
31:17	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

POWER CONTROL USAGE NOTES

After every reset, PCONP register contains the value that enables all interfaces and peripherals controlled by the PCONP to be enabled. Therefore, apart from proper configuring via peripheral dedicated registers, user's application has no need to access the PCONP in order to start using any of the on-board peripherals.

Power saving oriented systems should have 1s in the PCONP register only in positions that match peripherals really used in the application. All other bits, declared to be "Reserved" or dedicated to the peripherals not used in the current application, must be cleared to 0.

RESET

Reset has two sources on the LPC2119/2129/2194/2292/2294: the $\overline{\text{RESET}}$ pin and Watchdog Reset. The $\overline{\text{RESET}}$ pin is a Schmitt trigger input pin with an additional glitch filter. Assertion of chip Reset by any source starts the Wakeup Timer (see Wakeup Timer description later in this chapter), causing reset to remain asserted until the external Reset is de-asserted, the oscillator is running, a fixed number of clocks have passed, and the Flash controller has completed its initialization. The relationship between Reset, the oscillator, and the Wakeup Timer are shown in Figure 16.

The Reset glitch filter allows the processor to ignore external reset pulses that are very short, and also determines the minimum duration of $\overline{\text{RESET}}$ that must be asserted in order to guarantee a chip reset. Once asserted, $\overline{\text{RESET}}$ pin can be deasserted only when crystal oscillator is fully running and an adequate signal is present on the X1 pin of the LPC2119/2129/2194/2292/2294. Assuming that an external crystal is used in the crystal oscillator subsystem, after power on, the $\overline{\text{RESET}}$ pin should be asserted for 10 ms. For all subsequent resets when crystal osillator is already running and stable signal is on the X1 pin, the $\overline{\text{RESET}}$ pin needs to be asserted for 300 ns only.

Speaking in general, there are no sequence requirements for powering up the supplies (V_{18}, V_3, V_{18A} and V_{3A}). However, for proper reset handling It is absolutely necessary to have valid voltage supply on V_{18} pins, since on-chip Reset circuit and oscillator dedicated hardware are powered by them. V_3 pins enable microcontroller's interface to the environment via its digital pins. Consequently, not providing V_3 power supply will not affect the reset sequence itself, but will prevent microcontroller from communicating with external world.

When the internal Reset is removed, the processor begins executing at address 0, which isinitially the Reset vector mapped from the Boot Block. At that point, all of the processor and peripheral registers have been initialized to predetermined values.

External and internal Resets have some small differences. An external Reset causes the value of certain pins to be latched to configure the part. External circuitry cannot determine when an internal Reset occurs in order to allow setting up those special pins, so those latches are not reloaded during an internal Reset. Pins that are examined during an external Reset for various purposes are: P1.20/TRACESYNC, P1.26/RTCK, BOOT1 and BOOT0 (see chapters Pin Configuration on page 110, Pin Connect Block on page 126 and External Memory Controller (EMC) on page 56). Pin P0.14 (see Flash Memory System and Programming on page 262) is exemined by on-chip bootloader when this code is executed after reset.

It is possible for a chip Reset to occur during a Flash programming or erase operation. The Flash memory will interrupt the ongoing operation and hold off the completion of Reset to the CPU until internal Flash high voltages have settled.

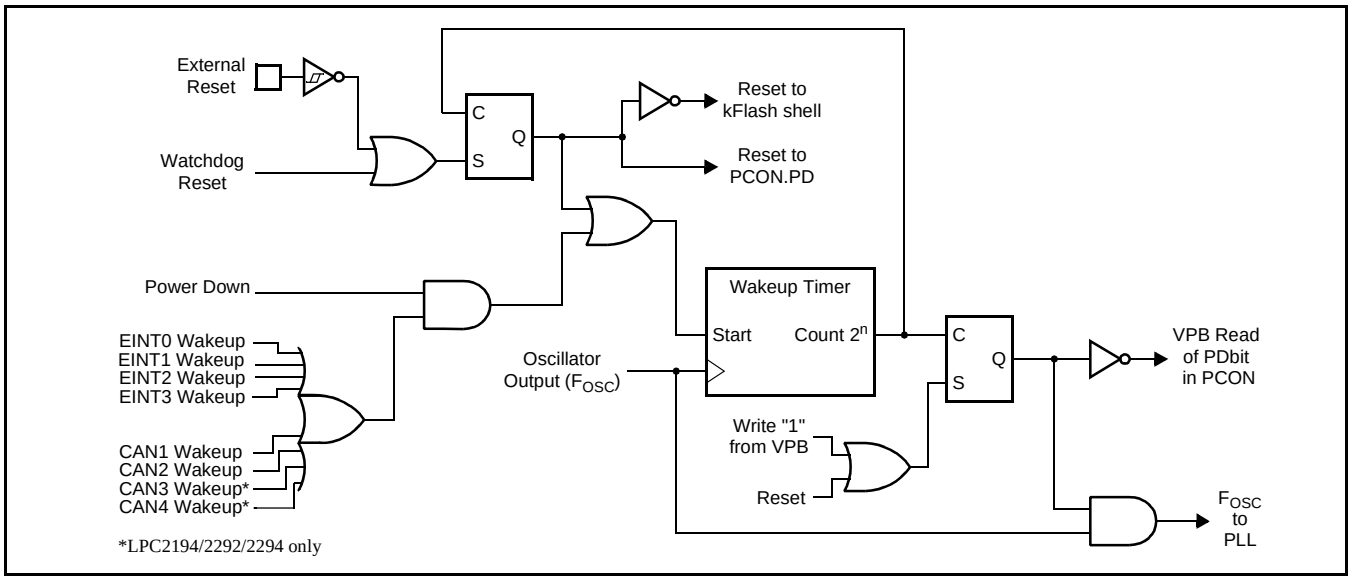


Figure 16: Reset Block Diagram including Wakeup Timer

VPB DIVIDER

The VPB Divider determines the relationship between the processor clock (cclk) and the clock used by peripheral devices (pclk). The VPB Divider serves two purposes. The first is to provides peripherals with desired pclk via VPB bus so that they can operate at the speed chosen for the ARM processor. In order to achieve this, the VPB bus may be slowed down to one half or one fourth of the processor clock rate. Because the VPB bus must work properly at power up (and its timing cannot be altered if it does not work since the VPB divider control registers reside on the VPB bus), the default condition at reset is for the VPB bus to run at one quarter speed. The second purpose of the VPB Divider is to allow power savings when an application does not require any peripherals to run at the full processor rate.

The connection of the VPB Divider relative to the oscillator and the processor clock is shown in Figure 17. Because the VPB Divider is connected to the PLL output, the PLL remains active (if it was running) during Idle mode.

VPBDIV Register (VPBDIV - 0xE01FC100)

The VPB Divider register contains two bits, allowing three divider values, as shown in Table 34.

Table 33: VPBDIV Register Map

Address	Name	Description	Access
0xE01FC100	VPBDIV	Controls the rate of the VPB clock in relation to the processor clock.	R/W

Table 34: VPB Divider Register (VPBDIV - 0xE01FC100)

VPBDIV	Function	Description	Reset Value
1:0	VPBDIV	The rate of the VPB clock is as follows: 0 0: VPB bus clock is one fourth of the processor clock. 0 1: VPB bus clock is the same as the processor clock. 1 0: VPB bus clock is one half of the processor clock. 1 1: Reserved. If this value is written to the VPBDIV register, it has no effect (the previous setting is retained).	0
3:2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
5:4	XCLKDIV	In the LPC2292/2294 (parts in 144 packages) only, these bits control the clock that can be driven onto the A23/XCLK pin. They have the same encoding as the VPBDIV bits above. A bit in the PINSEL2 register (Pin Connect Block on page 126) controls whether the pin carries A23 or the clock selected by this field. **Note: If this field and VPBDIV have the same value, the same clock is used on the VPB and XCLK.** (This might be useful for external logic dealing with the VPB peripherals).	0
7:6	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

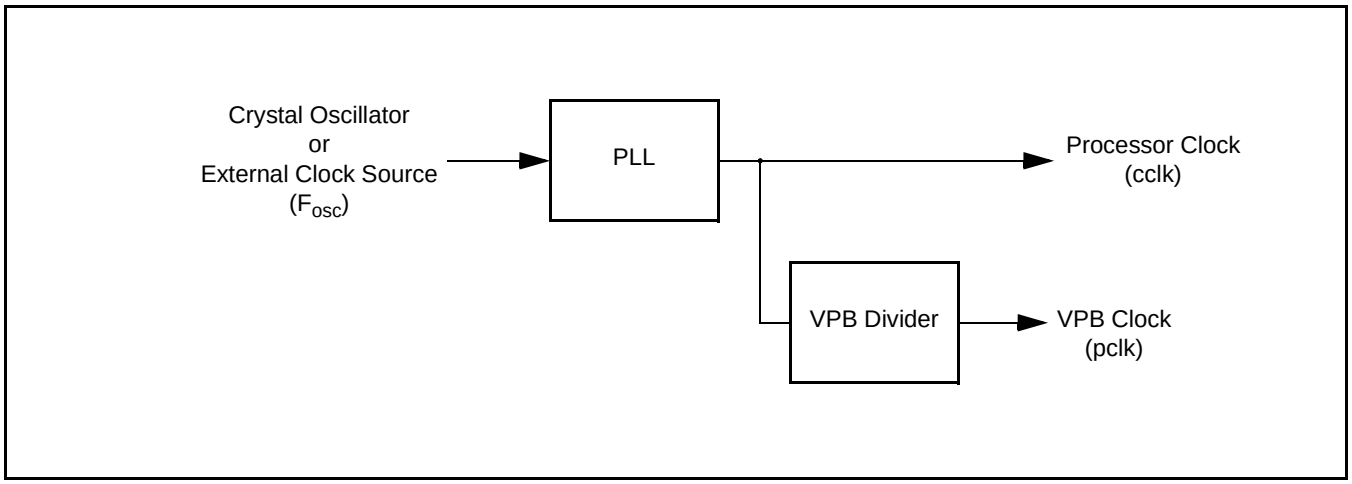


Figure 17: VPB Divider Connections

WAKEUP TIMER

The purpose of the wakeup timer is to ensure that the oscillator and other analog functions required for chip operation are fully functional before the processor is allowed to execute instructions. This is important at power on, all types of Reset, and whenever any of the aforementioned functions are turned off for any reason. Since the oscillator and other functions are turned off during Power Down mode, any wakeup of the processor from Power Down mode makes use of the Wakeup Timer.

The Wakeup Timer monitors the crystal oscillator as the means of checking whether it is safe to begin code execution. When power is applied to the chip, or some event caused the chip to exit Power down mode, some time is required for the oscillator to produce a signal of sufficient amplitude to drive the clock logic. The amount of time depends on many factors, including the rate of Vdd ramp (in the case of power on), the type of crystal and its electrical characteristics (if a quartz crystal is used), as well as any other external circuitry (e.g. capacitors), and the characteristics of the oscillator itself under the existing ambient conditions.

Once a clock is detected, the Wakeup Timer counts 4096 clocks, then enables the Flash memory to initialize. When the Flash memory initialization is complete, the processor is released to execute instructions if the external Reset has been de-asserted. In the case where an external clock source is used in the system (as opposed to a crystal connected to the oscillator pins), the possibility that there could be little or no delay for oscillator start-up must be considered. The Wakeup Timer design then ensures that any other required chip functions will be operational prior to the beginning of program execution.

The LPC2119/2129/2194/2292/2294 does not contain any analog function such as comparators that operate without clocks or any independent clock source such as a dedicated Watchdog oscillator. The only remaining functions that can operate in the absence of a clock source are the external interrupts (EINT0, EINT1, EINT2 and EINT3) and the CAN controllers. When an external interrupt is enabled for wakrup, and its selected event occurs, an oscillator wakeup cycle is started. Similarly,if a CAN block is enabled for wakeup and activity occurs on its CAN bus, an oscillator wakeup cycle is statrted. The actual interrupt (if any) occurs after the wakeup time expires, and is handled by the Vectored Interrupt Controller (VIC).

However, the pin multiplexing on the LPC2119/2129/2194/2292/229 (see Pin Configuration on page 110 and Pin Connect Block on page 126) was designed to allow other peripherals to, in effect, bring the device out of power down mode. The following pin-function pairings allow interrupts from events relating to UART0 or 1, SPI 0 or 1, or the I^2C: RxD0 / EINT0, SDA / EINT1, SSEL0 / EINT2, RxD1 / EINT3, DCD1 / EINT1, RI1 / EINT2, SSEL1 / EINT3.

To put the device in power down mode and allow activity on one or more of these buses or lines to power it back up, software should reprogram the pin function to External Interrupt, select the appropriate mode and polarity for the Interrupt, and then select power down mode. Upon wakeup software should restore the pin mulitplexing to the peripheral function.

All of the bus- or line-activity indications in the list above happen to be low-active. If software wants the device to come out of power -down mode in response to actity on more than one pin that share the same EINTi channel, it should program low-level sensitivity for that channel, because only in level mode will the channel logically OR the signals to wake the device.

The only flaw in this scheme is that the time to restart the oscillator prevents the LPC2119/2129/2194/2292/229 from capturing the bus or line activity that wakes it up. Idle mode is more appropriate than power-down mode for devices that must capture and respond to external activity in a timely manner.

To summarize: on the LPC2119/2129/2194/2292/2294, the Wakeup Timer enforces a minimum reset duration based on the crystal oscillator, and is activated whenever there is a wakeup from Power Down mode or any type of Reset.

5. MEMORY ACCELERATOR MODULE (MAM)

INTRODUCTION

Simply put, the Memory Accelerator Module (MAM) attempts to have the next ARM instruction that will be needed in its latches in time to prevent CPU fetch stalls. The method used is to split the Flash memory into two banks, each capable of independent accesses. Each of the two Flash banks has its own Prefetch Buffer and Branch Trail Buffer. The Branch Trail Buffers for the two banks capture two 128-bit lines of Flash data when an Instruction Fetch is not satisfied by either the Prefetch buffer nor Branch Trail buffer for its bank, and for which a prefetch has not been initiated. Each prefetch buffer captures one 128-bit line of instructions from its Flash bank, at the conclusion of a prefetch cycle initiated speculatively by the MAM.

Each 128 bit value includes four 32-bit ARM instructions or eight 16-bit Thumb instructions. During sequential code execution, typically one Flash bank contains or is fetching the current instruction and the entire Flash line that contains it. The other bank contains or is prefetching the next sequential code line. After a code line delivers its last instruction, the bank that contained it begins to fetch the next line in that bank.

Timing of Flash read operations is programmable and is described later in this section as well as in the System Control Block section.

Branches and other program flow changes cause a break in the sequential flow of instruction fetches described above. When a backward branch occurs, there is a distinct possibility that a loop is being executed. In this case the Branch Trail Buffers may already contain the target instruction. If so, execution continues without the need for a Flash read cycle. For a forward branch, there is also a chance that the new address is already contained in one of the Prefetch Buffers. If it is, the branch is again taken with no delay.

When a branch outside the contents of the Branch Trail and Prefetch buffers is taken, one Flash Access cycle is needed to load the Branch Trail buffers. Subsequently, there will typically be no further fetch delays until another such “Instruction Miss” occurs.

The Flash memory controller detects data accesses to the Flash memory and uses a separate buffer to store the results in a manner similar to that used during code fetches. This allows faster access to data if it is accessed sequentially. A single line buffer is provided for data accesses, as opposed to the two buffers per Flash bank that are provided for code accesses. There is no prefetch function for data accesses.

Memory Accelerator Module Blocks

The Memory Accelerator Module is divided into several functional blocks:

- A Flash Address Latch for each bank. An Incrementer function is associated with the Bank 0 Flash Address latch.
- Two Flash Memory Banks.
- Instruction Latches, Data Latches, Address Comparison latches.
- Wait logic

Figure 18 shows a simplified block diagram of the Memory Accelerator Module data paths.

In the following descriptions, the term “fetch” applies to an explicit Flash read request from the ARM. “prefetch” is used to denote a Flash read of instructions beyond the current processor fetch address.

Flash Memory Banks

There are two banks of Flash memory in order to allow two parallel accesses and eliminate delays for sequential accesses.

Flash programming operations are not controlled by the Memory Accelerator Module, but are handled as a separate function. A “boot block” sector contains Flash programming algorithms that may be called as part of the application program, and a loader that may be run to allow serial programming of the Flash memory.

The Flash memories are wired so that each sector exists in both banks, such that a sector erase operation acts on part of both banks simultaneously. In effect, the existence of two banks is transparent to the programming functions.

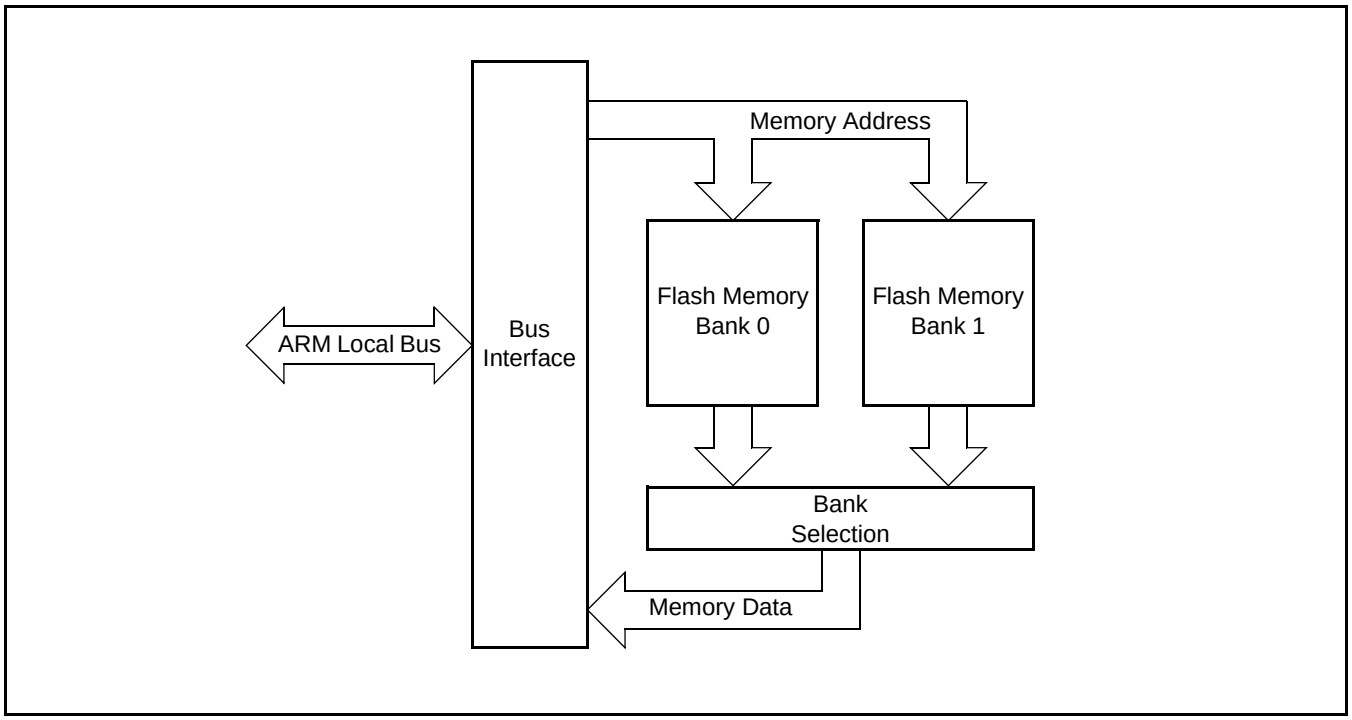


Figure 18: Simplified Block Diagram of the Memory Accelerator Module

Instruction Latches and Data Latches

Code and Data accesses are treated separately by the Memory Accelerator Module. There are two sets of 128-bit Instruction Latches and 12-bit Comparison Address Latches associated with each Flash Bank. One of the two sets, called the Branch Trail Buffer, holds the data and comparison address for that bank from the last Instruction miss. The other set, called the Prefetch Buffer, holds the data and comparison address from prefetches undertaken speculatively by the MAM. Each Instruction Latch holds 4 words of code (4 ARM instructions, or 8 Thumb instructions).

Similarly there is a 128-bit Data Latch and 13-bit Data Address latch, that are used during Data cycles. This single set of latches is shared by both Flash banks. Each Data access that is not in the Data latch causes a Flash fetch of 4 words of data, which are captured in the Data latch. This speeds up sequential Data operations, but has little or no effect on random accesses.

Flash Programming Issues

Since the Flash memory does not allow accesses during programming and erase operations, it is necessary for the MAM to force the CPU to wait if a memory access to a Flash address is requested while the Flash module is busy. (This is accomplished by asserting the ARM7TDMI-S local bus signal CLKEN.) Under some conditions, this delay could result in a Watchdog time-out. The user will need to be aware of this possibility and take steps to insure that an unwanted Watchdog reset does not cause a system failure while programming or erasing the Flash memory.

In order to preclude the possibility of stale data being read from the Flash memory, the MAM holding latches are automatically invalidated at the beginning of any Flash programming or erase operation. Any subsequent read from a Flash address will cause a new fetch to be initiated after the Flash operation has completed.

MEMORY ACCELERATOR MODULE OPERATING MODES

Three modes of operation are defined for the MAM, trading off performance for ease of predictability:

0) MAM off. All memory requests result in a Flash read operation (see note 2 below). There are no instruction prefetches.

1) MAM partially enabled. Sequential instruction accesses are fulfilled from the holding latches if the data is present. Instruction prefetch is enabled. Non-sequential instruction accesses initiate Flash read operations (see note 2 below). This means that all branches cause memory fetches. All data operations cause a Flash read because buffered data access timing is hard to predict and is very situation dependent.

2) MAM fully enabled. Any memory request (code or data) for a value that is contained in one of the corresponding holding latches is fulfilled from the latch. Instruction prefetch is enabled. Flash read operations are initiated for instruction prefetch and code or data values not available in the corresponding holding latches.

Table 35: MAM Responses to Program Accesses of Various Types

Program Memory Request Type	MAM Mode		
	0	1	2
Sequential access, data in MAM latches	Initiate Fetch [2]	Use Latched Data [1]	Use Latched Data [1]
Sequential access, data not in MAM latches	Initiate Fetch	Initiate Fetch [1]	Initiate Fetch [1]
Non-Sequential access, data in MAM latches	Initiate Fetch [2]	Initiate Fetch [1, 2]	Use Latched Data [1]
Non-Sequential access, data not in MAM latches	Initiate Fetch	Initiate Fetch [1]	Initiate Fetch [1]

Table 36: MAM Responses to Data and DMA Accesses of Various Types

Data Memory Request Type	MAM Mode		
	0	1	2
Sequential access, data in MAM latches	Initiate Fetch [2]	Initiate Fetch [2]	Use Latched Data
Sequential access, data not in MAM latches	Initiate Fetch	Initiate Fetch	Initiate Fetch
Non-Sequential access, data in MAM latches	Initiate Fetch [2]	Initiate Fetch [2]	Use Latched Data
Non-Sequential access, data not in MAM latches	Initiate Fetch	Initiate Fetch	Initiate Fetch

1. Instruction prefetch is enabled in modes 1 and 2.
2. The MAM actually uses latched data if it is available, but mimics the timing of a Flash read operation. This saves power while resulting in the same execution timing. The MAM can truly be turned off by setting the fetch timing value in MAMTIM to one clock.

MAM CONFIGURATION

After reset the MAM defaults to the disabled state. Software can turn memory access acceleration on or off at any time. This allows most of an application to be run at the highest possible performance, while certain functions can be run at a somewhat slower but more predictable rate if more precise timing is required.

REGISTER DESCRIPTION

All registers, regardless of size, are on word address boundaries. Details of the registers appear in the description of each function.

Table 37: Summary of System Control Registers

Name	Description	Access	Reset Value*	Address
MAM				
MAMCR	Memory Accelerator Module Control Register. Determines the MAM functional mode, that is, to what extent the MAM performance enhancements are enabled. See Table 38.	R/W	0	0xE01FC000
MAMTIM	Memory Accelerator Module Timing control. Determines the number of clocks used for Flash memory fetches (1 to 7 processor clocks).	R/W	0x07	0xE01FC004

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

MAM Control Register (MAMCR - 0xE01FC000)

Two configuration bits select the three MAM operating modes, as shown in Table 38. Following Reset, MAM functions are disabled. Changing the MAM operating mode causes the MAM to invalidate all of the holding latches, resulting in new reads of Flash information as required.

Table 38: MAM Control Register (MAMCR - 0xE01FC000)

MAMCR	Function	Description	Reset Value
1:0	MAM mode control	These bits determine the operating mode of the MAM as follows: 0 0 - MAM functions disabled. 0 1 - MAM functions partially enabled. 1 0 - MAM functions fully enabled. 1 1 - reserved	0
7:2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

MAM Timing Register (MAMTIM - 0xE01FC004)

The MAM Timing register determines how many cclk cycles are used to access the Flash memory. This allows tuning MAM timing to match the processor operating frequency. Flash access times from 1 clock to 7 clocks are possible. Single clock Flash accesses would essentially remove the MAM from timing calculations. In this case the MAM mode may be selected to optimize power usage.

Table 39: MAM Timing Register (MAMTIM - 0xE01FC004)

MAMTIM	Function	Description	Reset Value
2:0	MAM Fetch Cycle timing	These bits set the duration of MAM Flash fetch operations as follows: 0 0 0 = 0 - Reserved. 0 0 1 = 1 - MAM fetch cycles are 1 processor clock (cclk) in duration. 0 1 0 = 2 - MAM fetch cycles are 2 processor clocks (cclks) in duration. 0 1 1 = 3 - MAM fetch cycles are 3 processor clocks (cclks) in duration. 1 0 0 = 4 - MAM fetch cycles are 4 processor clocks (cclks) in duration. 1 0 1 = 5 - MAM fetch cycles are 5 processor clocks (cclks) in duration. 1 1 0 = 6 - MAM fetch cycles are 6 processor clocks (cclks) in duration. 1 1 1 = 7 - MAM fetch cycles are 7 processor clocks (cclks) in duration. **Warning:** Improper setting of this value may result in incorrect operation of the device.	0x07
7:3	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

MAM USAGE NOTES

When changing MAM timing, the MAM must first be turned off by writing a zero to MAMCR. A new value may then be written to MAMTIM. Finally, the MAM may be turned on again by writing a value (1 or 2) corresponding to the desired operating mode to MAMCR.

For system clock slower than 20 MHz, MAMTIM can be 001. For system clock between 20 MHz and 40 MHz, Flash access time is suggested to be 2 CCLKs, while in systems with system clock faster than 40 MHz, 3 CCLKs are proposed.

6. VECTORED INTERRUPT CONTROLLER (VIC)

FEATURES

- ARM PrimeCell™ Vectored Interrupt Controller
- 32 interrupt request inputs
- 16 vectored IRQ interrupts
- 16 priority levels dynamically assigned to interrupt requests
- Software interrupt generation

DESCRIPTION

The Vectored Interrupt Controller (VIC) takes 32 interrupt request inputs and programmably assigns them into 3 categories, FIQ, vectored IRQ, and non-vectored IRQ. The programmable assignment scheme means that priorities of interrupts from the various peripherals can be dynamically assigned and adjusted.

Fast Interrupt reQuest (FIQ) requests have the highest priority. If more than one request is assigned to FIQ, the VIC ORs the requests to produce the FIQ signal to the ARM processor. The fastest possible FIQ latency is achieved when only one request is classified as FIQ, because then the FIQ service routine can simply start dealing with that device. But if more than one request is assigned to the FIQ class, the FIQ service routine can read a word from the VIC that identifies which FIQ source(s) is (are) requesting an interrupt.

Vectored IRQs have the middle priority, but ony 16 of the 32 requests can be assigned to this category. Any of the 32 requests can be assigned to any of the 16 vectored IRQ slots, among which slot 0 has the highest priority and slot 15 has the lowest.

Non-vectored IRQs have the lowest priority.

The VIC ORs the requests from all the vectored and non-vectored IRQs to produce the IRQ signal to the ARM processor. The IRQ service routine can start by reading a register from the VIC and jumping there. If any of the vectored IRQs are requesting, the VIC provides the address of the highest-priority requesting IRQs service routine, otherwise it provides the address of a default routine that is shared by all the non-vectored IRQs. The default routine can read another VIC register to see what IRQs are active.

All registers in the VIC are word registers. Byte and halfword reads and write are not supported.

Additional information on the Vectored Interrupt Controller is available in the ARM PrimeCell™ Vectored Interrupt Controller (PL190) documentation.

REGISTER DESCRIPTION

The VIC implements the registers shown in Table 40. More detailed descriptions follow.

Table 40: VIC Register Map

Name	Description	Access	Reset Value*	Address
VICIRQStatus	IRQ Status Register. This register reads out the state of those interrupt requests that are enabled and classified as IRQ.	RO	0	0xFFFF F000
VICFIQStatus	FIQ Status Requests. This register reads out the state of those interrupt requests that are enabled and classified as FIQ.	RO	0	0xFFFF F004
VICRawIntr	Raw Interrupt Status Register. This register reads out the state of the 32 interrupt requests / software interrupts, regardless of enabling or classification.	RO	0	0xFFFF F008
VICIntSelect	Interrupt Select Register. This register classifies each of the 32 interrupt requests as contributing to FIQ or IRQ.	R/W	0	0xFFFF F00C
VICIntEnable	Interrupt Enable Register. This register controls which of the 32 interrupt requests and software interrupts are enabled to contribute to FIQ or IRQ.	R/W	0	0xFFFF F010
VICIntEnClr	Interrupt Enable Clear Register. This register allows software to clear one or more bits in the Interrupt Enable register.	W	0	0xFFFF F014
VICSoftInt	Software Interrupt Register. The contents of this register are ORed with the 32 interrupt requests from various peripheral functions.	R/W	0	0xFFFF F018
VICSoftIntClear	Software Interrupt Clear Register. This register allows software to clear one or more bits in the Software Interrupt register.	W	0	0xFFFF F01C
VICProtection	Protection enable register. This register allows limiting access to the VIC registers by software running in privileged mode.	R/W	0	0xFFFF F020
VICVectAddr	Vector Address Register. When an IRQ interrupt occurs, the IRQ service routine can read this register and jump to the value read.	R/W	0	0xFFFF F030
VICDefVectAddr	Default Vector Address Register. This register holds the address of the Interrupt Service routine (ISR) for non-vectored IRQs.	R/W	0	0xFFFF F034
VICVectAddr0	Vector address 0 register. Vector Address Registers 0-15 hold the addresses of the Interrupt Service routines (ISRs) for the 16 vectored IRQ slots.	R/W	0	0xFFFF F100
VICVectAddr1	Vector address 1 register	R/W	0	0xFFFF F104
VICVectAddr2	Vector address 2 register	R/W	0	0xFFFF F108
VICVectAddr3	Vector address 3 register	R/W	0	0xFFFF F10C
VICVectAddr4	Vector address 4 register	R/W	0	0xFFFF F110
VICVectAddr5	Vector address 5 register	R/W	0	0xFFFF F114
VICVectAddr6	Vector address 6 register	R/W	0	0xFFFF F118
VICVectAddr7	Vector address 7 register	R/W	0	0xFFFF F11C
VICVectAddr8	Vector address 8 register	R/W	0	0xFFFF F120
VICVectAddr9	Vector address 9 register	R/W	0	0xFFFF F124

Table 40: VIC Register Map

Name	Description	Access	Reset Value*	Address
VICVectAddr10	Vector address 10 register	R/W	0	0xFFFF F128
VICVectAddr11	Vector address 11 register	R/W	0	0xFFFF F12C
VICVectAddr12	Vector address 12 register	R/W	0	0xFFFF F130
VICVectAddr13	Vector address 13 register	R/W	0	0xFFFF F134
VICVectAddr14	Vector address 14 register	R/W	0	0xFFFF F138
VICVectAddr15	Vector address 15 register	R/W	0	0xFFFF F13C
VICVectCntl0	Vector control 0 register. Vector Control Registers 0-15 each control one of the 16 vectored IRQ slots. Slot 0 has the highest priority and slot 15 the lowest.	R/W	0	0xFFFF F200
VICVectCntl1	Vector control 1 register	R/W	0	0xFFFF F204
VICVectCntl2	Vector control 2 register	R/W	0	0xFFFF F208
VICVectCntl3	Vector control 3 register	R/W	0	0xFFFF F20C
VICVectCntl4	Vector control 4 register	R/W	0	0xFFFF F210
VICVectCntl5	Vector control 5 register	R/W	0	0xFFFF F214
VICVectCntl6	Vector control 6 register	R/W	0	0xFFFF F218
VICVectCntl7	Vector control 7 register	R/W	0	0xFFFF F21C
VICVectCntl8	Vector control 8 register	R/W	0	0xFFFF F220
VICVectCntl9	Vector control 9 register	R/W	0	0xFFFF F224
VICVectCntl10	Vector control 10 register	R/W	0	0xFFFF F228
VICVectCntl11	Vector control 11 register	R/W	0	0xFFFF F22C
VICVectCntl12	Vector control 12 register	R/W	0	0xFFFF F230
VICVectCntl13	Vector control 13 register	R/W	0	0xFFFF F234
VICVectCntl14	Vector control 14 register	R/W	0	0xFFFF F238
VICVectCntl15	Vector control 15 register	R/W	0	0xFFFF F23C

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

VIC REGISTERS

This section describes the VIC registers in the order in which they are used in the VIC logic, from those closest to the interrupt request inputs to those most abstracted for use by software. For most people, this is also the best order to read about the registers when learning the VIC.

Software Interrupt Register (VICSoftInt - 0xFFFFF018, Read/Write)

The contents of this register are ORed with the 32 interrupt requests from the various peripherals, before any other logic is applied.

Table 41: Software Interrupt Register (VICSoftInt - 0xFFFFF018, Read/Write)

VICSoftInt	Function	Reset Value
31:0	1:force the interrupt request with this bit number. 0: do not force the interrupt request with this bit number. Writing zeroes to bits in VICSoftInt has no effect, see VICSoftIntClear.	0

Software Interrupt Clear Register (VICSoftIntClear - 0xFFFFF01C, Write Only)

This register allows software to clear one or more bits in the Software Interrupt register, without having to first read it.

Table 42: Software Interrupt Clear Register (VICSoftIntClear - 0xFFFFF01C, Write Only)

VICSoftIntClear	Function	Reset Value
31:0	1: writing a 1 clears the corresponding bit in the Software Interrupt register, thus releasing the forcing of this request. 0: writing a 0 leaves the corresponding bit in VICSoftInt unchanged.	0

Raw Interrupt Status Register (VICRawIntr - 0xFFFFF008, Read Only)

This register reads out the state of the 32 interrupt requests and software interrupts, regardless of enabling or classification.

Table 43: Raw Interrupt Status Register (VICRawIntr - 0xFFFFF008, Read-Only)

VICRawIntr	Function	Reset Value
31:0	1: the interrupt request or software interrupt with this bit number is asserted. 0: the interrupt request or software interrupt with this bit number is negated.	0

Interrupt Enable Register (VICIntEnable - 0xFFFFF010, Read/Write)

This register controls which of the 32 interrupt requests and software interrupts contribute to FIQ or IRQ.

Table 44: Interrupt Enable Register (VICINtEnable - 0xFFFFF010, Read/Write)

VICIntEnable	Function	Reset Value
31:0	When this register is read, 1s indicate interrupt requests or software interrupts that are enabled to contribute to FIQ or IRQ. When this register is written, ones enable interrupt requests or software interrupts to contribute to FIQ or IRQ, zeroes have no effect. See the VICIntEnClear register (Table 45 below), for how to disable interrupts.	0

Interrupt Enable Clear Register (VICIntEnClear - 0xFFFFF014, Write Only)

This register allows software to clear one or more bits in the Interrupt Enable register, without having to first read it.

Table 45: Software Interrupt Clear Register (VICIntEnClear - 0xFFFFF014, Write Only)

VICIntEnClear	Function	Reset Value
31:0	1: writing a 1 clears the corresponding bit in the Interrupt Enable register, thus disabling interrupts for this request. 0: writing a 0 leaves the corresponding bit in VICIntEnable unchanged.	0

Interrupt Select Register (VICIntSelect - 0xFFFFF00C, Read/Write)

This register classifies each of the 32 interrupt requests as contributing to FIQ or IRQ.

Table 46: Interrupt Select Register (VICIntSelect - 0xFFFFF00C, Read/Write)

VICIntSelect	Function	Reset Value
31:0	1: the interrupt request with this bit number is assigned to the FIQ category. 0: the interrupt request with this bit number is assigned to the IRQ category.	0

IRQ Status Register (VICIRQStatus - 0xFFFFF000, Read Only)

This register reads out the state of those interrupt requests that are enabled and classified as IRQ. It does not differentiate between vectored and non-vectored IRQs.

Table 47: IRQ Status Register (VICIRQStatus - 0xFFFFF000, Read-Only)

VICIRQStatus	Function	Reset Value
31:0	1: the interrupt request with this bit number is enabled, classified as IRQ, and asserted.	0

FIQ Status Register (VICFIQStatus - 0xFFFFF004, Read Only)

This register reads out the state of those interrupt requests that are enabled and classified as FIQ. If more than one request is classified as FIQ, the FIQ service routine can read this register to see which request(s) is (are) active.

Table 48: IRQ Status Register (VICFIQStatus - 0xFFFFF004, Read-Only)

VICFIQStatus	Function	Reset Value
31:0	1: the interrupt request with this bit number is enabled, classified as FIQ, and asserted.	0

Vector Control Registers 0-15 (VICVectCntl0-15 - 0xFFFFF200-23C, Read/Write)

Each of these registers controls one of the 16 vectored IRQ slots. Slot 0 has the highest priority and slot 15 the lowest. Note that disabling a vectored IRQ slot in one of the VICVectCntl registers does not disable the interrupt itself, the interrupt is simply changed to the non-vectored form.

Table 49: Vector Control Registers (VICVectCntl0-15 - 0xFFFFF200-23C, Read/Write)

VICVectCntl0-15	Function	Reset Value
5	1: this vectored IRQ slot is enabled, and can produce a unique ISR address when its assigned interrupt request or software interrupt is enabled, classified as IRQ, and asserted.	0
4:0	The number of the interrupt request or software interrupt assigned to this vectored IRQ slot. As a matter of good programming practice, software should not assign the same interrupt number to more than one enabled vectored IRQ slot. But if this does occur, the lower-numbered slot will be used when the interrupt request or software interrupt is enabled, classified as IRQ, and asserted.	0

Vector Address Registers 0-15 (VICVectAddr0-15 - 0xFFFFF100-13C, Read/Write)

These registers hold the addresses of the Interrupt Service routines (ISRs) for the 16 vectored IRQ slots.

Table 50: Vector Address Registers (VICVectAddr0-15 - 0xFFFFF100-13C, Read/Write)

VICVectAddr0-15	Function	Reset Value
31:0	When one or more interrupt request or software interrupt is (are) enabled, classified as IRQ, asserted, and assigned to an enabled vectored IRQ slot, the value from this register for the highest-priority such slot will be provided when the IRQ service routine reads the Vector Address register (VICVectAddr).	0

Default Vector Address Register (VICDefVectAddr - 0xFFFFF034, Read/Write)

This register holds the address of the Interrupt Service routine (ISR) for non-vectored IRQs.

Table 51: Default Vector Address Register (VICDefVectAddr - 0xFFFFF034, Read/Write)

VICDefVectAddr	Function	Reset Value
31:0	When an IRQ service routine reads the Vector Address register (VICVectAddr), and no IRQ slot responds as described above, this address is returned.	0

Vector Address Register (VICVectAddr - 0xFFFFF030, Read/Write)

When an IRQ interrupt occurs, the IRQ service routine can read this register and jump to the value read.

Table 52: Vector Address Register (VICVectAddr - 0xFFFFF030, Read/Write)

VICVectAddr	Function	Reset Value
31:0	If any of the interrupt requests or software interrupts that are assigned to a vectored IRQ slot is (are) enabled, classified as IRQ, and asserted, reading from this register returns the address in the Vector Address Register for the highest-priority such slot (lowest-numbered) such slot. Otherwise it returns the address in the Default Vector Address Register. Writing to this register does not set the value for future reads from it. Rather, this register should be written near the end of an ISR, to update the priority hardware.	0

Protection Enable Register (VICProtection - 0xFFFFF020, Read/Write)

This one-bit register controls access to the VIC registers by software running in User mode.

Table 53: Protection Enable Register (VICProtection - 0xFFFFF020, Read/Write)

VICProtection	Function	Reset Value
0	1: the VIC registers can only be accessed in privileged mode. 0: VIC registers can be accessed in User or privileged mode.	0

INTERRUPT SOURCES

Table 54 lists the interrupt sources for each peripheral function. Each peripheral device has one interrupt line connected to the Vectored Interrupt Controller, but may have several internal interrupt flags. Individual interrupt flags may also represent more than one interrupt source.

Table 54: Connection of Interrupt Sources to the Vectored Interrupt Controller

Block	Flag(s)	VIC Channel #
WDT	Watchdog Interrupt (WDINT)	0
-	Reserved for software interrupts only	1
ARM Core	Embedded ICE, DbgCommRx	2
ARM Core	Embedded ICE, DbgCommTx	3
TIMER0	Match 0 - 3 (MR0, MR1, MR2, MR3) Capture 0 - 3 (CR0, CR1, CR2, CR3)	4
TIMER1	Match 0 - 3 (MR0, MR1, MR2, MR3) Capture 0 - 3 (CR0, CR1, CR2, CR3)	5
UART0	Rx Line Status (RLS) Transmit Holding Register Empty (THRE) Rx Data Available (RDA) Character Time-out Indicator (CTI)	6
UART1	Rx Line Status (RLS) Transmit Holding Register Empty (THRE) Rx Data Available (RDA) Character Time-out Indicator (CTI) Modem Status Interrupt (MSI)	7
PWM0	Match 0 - 6 (MR0, MR1, MR2, MR3, MR4, MR5, MR6)	8
I^2C	SI (state change)	9
SPI0	SPI Interrupt Flag (SPIF) Mode Fault (MODF)	10
SPI1	SPI Interrupt Flag (SPIF) Mode Fault (MODF)	11
PLL	PLL Lock (PLOCK)	12
RTC	Counter Increment (RTCCIF) Alarm (RTCALF)	13
System Control	External Interrupt 0 (EINT0)	14
System Control	External Interrupt 1 (EINT1)	15
System Control	External Interrupt 2 (EINT2)	16
System Control	External Interrupt 3 (EINT3)	17
A/D	A/D Converter	18

Table 54: Connection of Interrupt Sources to the Vectored Interrupt Controller

Block	Flag(s)	VIC Channel #
CAN	CAN and Acceptance Filter (1 ORed CAN, LUTerr int)	19
	CAN1 Tx	20
	CAN2 Tx	21
	CAN3 Tx (LPC2194/2292/2294 only, otherwise Reserved)	22
	CAN4 Tx (LPC2194/2292/2294 only, otherwise Reserved)	23
	Reserved	24-25
	CAN1 Rx	26
	CAN2 Rx	27
	CAN3 Rx (LPC2194/2292/2294 only, otherwise Reserved)	28
	CAN4 Rx (LPC2194/2292/2294 only, otherwise Reserved)	29
	Reserved	30-31

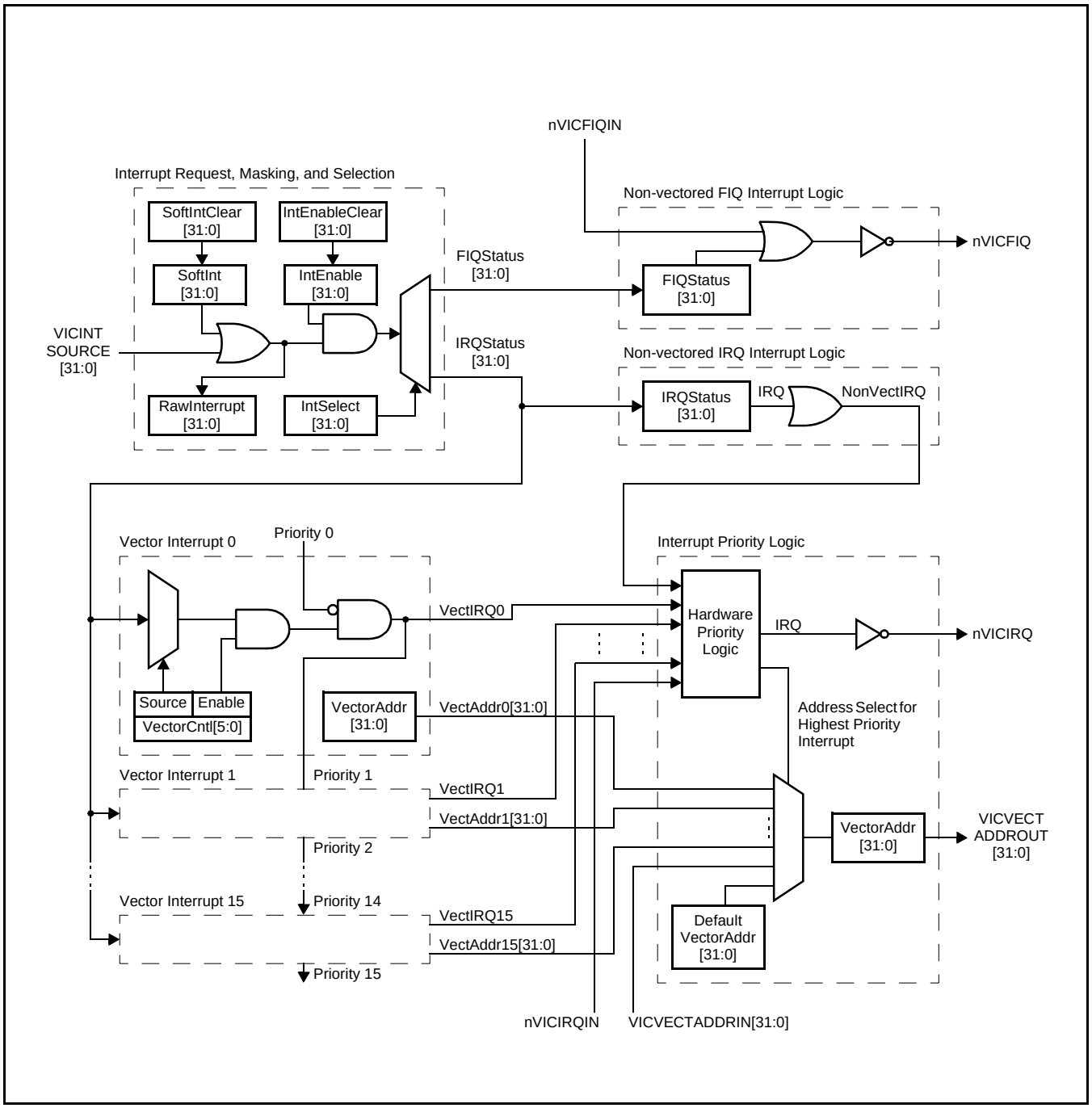


Figure 19: Block Diagram of the Vectored Interrupt Controller

SPURIOUS INTERRUPTS

Spurious interrupts are possible to occur in the ARM7TDMI based microcontroller such as the LPC2119/2129/2194/2292/2294 due to the asynchronous interrupt handling. The asynchronous character of the interrupt processing has its roots in the interaction of the core and the VIC. If the VIC state is changed between the moments when the core detects an interrupt and the core actually processes an interrupt, problems may be generated.

Real-life application may experience following scenario:

1) VIC decides there is an IRQ interrupt and sends the IRQ signal to the core.

2) Core latches the IRQ state.

3) Processing continues for a few cycles due to pipelining.

4) Core loads IRQ address from VIC.

Furthermore, It is possible that the VIC state has changed during the step 3. For example, VIC was modified so that the interrupt that triggered the sequence starting with step 1) is no longer pending -interrupt got disabled in the executed code. In this case, the VIC will not be able to clearly identify the interrupt that generated the interrupt request, and as a result the VIC will return the default interrupt VicDefVectAddr (0xFFFF F034).

This potentialy disastrous chain of events can be prevented in two ways:

1. Application code should be set up in a way to prevent the spurious interrupts to ever happen. Simple guarding of changes to the VIC may not not be enough, since for example glitches on level sensitive interrupts can also cause spurious interrupts.

2. VIC default handler should be set up and tested properly.

Details and Case Studies on Spurious Interrupts

This chapter contains details that can be obtained from the official ARM website (http://www.arm.com), FAQ section under the "Technical Support" link: http://www.arm.com/support/faqip/3677.html.

What happens if an interrupt occurs as it is being disabled?

Applies to: ARM7TDMI

If an interrupt is received by the core during execution of an instruction that disables interrupts, the ARM7 family will still take the interrupt. This occurs for both IRQ and FIQ interrupts.

For example, consider the follow instruction sequence:

```
MRS    r0, cpsr
ORR    r0, r0, #I_Bit:OR:F_Bit    ;disable IRQ and FIQ interrupts
MSR    cpsr_c, r0
```

If an IRQ interrupt is received during execution of the MSR instruction, then the behavior will be as follows:

- The IRQ interrupt is latched
- The MSR cpsr, r0 executes to completion setting both the I bit and the F bit in the CPSR
- The IRQ interrupt is taken because the core was committed to taking the interrupt exception before the I bit was set in the CPSR.
- The CPSR (with the I bit and F bit set) is moved to the SPSR_irq

This means that, on entry to the IRQ interrupt service routine, one can see the unusual effect that an IRQ interrupt has just been taken while the I bit in the SPSR is set. In the example above, the F bit will also be set in both the CPSR and SPSR. This means that FIQs are disabled upon entry to the IRQ service routine, and will remain so until explicitly re-enabled. FIQs will not be re-enabled automatically by the IRQ return sequence.

Although the example shows both IRQ and FIQ interrupts being disabled, similar behavior occurs when only one of the two interrupt types is being disabled. The fact that the core processes the IRQ after completion of the MSR instruction which disables IRQs does not normally cause a problem, since an interrupt arriving just one cycle earlier would be expected to be taken. When the interrupt routine returns with an instruction like:

```
SUBS    pc, lr, #4
```

the SPSR_IRQ is restored to the CPSR. The CPSR will now have the I bit and F bit set, and therefore execution will continue with all interrupts disabled.

However, this can cause problems in the following cases:

Problem 1: A particular routine maybe called as an IRQ handler, or as a regular subroutine. In the latter case, the system guarantees that IRQs would have been disabled prior to the routine being called. The routine exploits this restriction to determine how it was called (by examining the I bit of the SPSR), and returns using the appropriate instruction. If the routine is entered due to an IRQ being received during execution of the MSR instruction which disables IRQs, then the I bit in the SPSR will be set. The routine would therefore assume that it could not have been entered via an IRQ.

Problem 2: FIQs and IRQs are both disabled by the same write to the CPSR. In this case, if an IRQ is received during the CPSR write, FIQs will be disabled for the execution time of the IRQ handler. This may not be acceptable in a system where FIQs must not be disabled for more than a few cycles.

Workaround:

There are 3 suggested workarounds. Which of these is most applicable will depend upon the requirements of the particular system.

Solution 1: Add code similar to the following at the start of the interrupt routine.

```
SUB     lr, lr, #4          ; Adjust LR to point to return
STMFD   sp!, {..., lr}      ; Get some free regs
MRS     lr, SPSR            ; See if we got an interrupt while
TST     lr, #I_Bit          ; interrupts were disabled.
LDMNEFD sp!, {..., pc}^     ; If so, just return immediately.
                            ; The interrupt will remain  pending since we haven't
                            ; acknowledged it and will be reissued when interrupts are
                            ; next enabled.
                            ; Rest of interrupt routine
```

This code will test for the situation where the IRQ was received during a write to disable IRQs. If this is the case, the code returns immediately - resulting in the IRQ not being acknowledged (cleared), and further IRQs being disabled.

Similar code may also be applied to the FIQ handler, in order to resolve the first issue.

This is the recommended workaround, as it overcomes both problems mentioned above. However, in the case of problem two, it does add several cycles to the maximum length of time FIQs will be disabled.

Solution 2: Disable IRQs and FIQs using separate writes to the CPSR, eg:

```
MRS     r0, cpsr
ORR     r0, r0, #I_Bit     ;disable IRQs
MSR     cpsr_c, r0
ORR     r0, r0, #F_Bit     ;disable FIQs
MSR     cpsr_c, r0
```

This is the best workaround where the maximum time for which FIQs are disabled is critical (it does not increase this time at all). However, it does not solve problem one, and requires extra instructions at every point where IRQs and FIQs are disabled together.

Solution 3: Re-enable FIQs at the beginning of the IRQ handler. As the required state of all bits in the c field of the CPSR are known, this can be most efficiently be achieved by writing an immediate value to CPSR_c, for example:

```
MSR     cpsr_c, #I_Bit:OR:irq_MODE      ;IRQ should be disabled
                                        ;FIQ enabled
                                        ;ARM state, IRQ mode
```

This requires only the IRQ handler to be modified, and FIQs may be re-enabled more quickly than by using workaround 1. However, this should only be used if the system can guarantee that FIQs are never disabled while IRQs are enabled. It does not address problem one.

VIC USAGE NOTES

If user's code is runing from the on-chip RAM and an aplication uses interrupts, interrupt vectors must be re-mapped to flash address 0x0. This is necessary because all the exception vectors are located at addresses 0x0 and above. This is easily achieved by configuring MEMMAP register (located in System Control Block) to User RAM mode. Application code should be linked such that at 0x4000 0000 the Interrupt Vector Tabe (IVT) will reside.

Although multiple sources can be selected (VICIntSelect) to generate FIQ request, only one interrupt service routine should be dedicated to service all available/present FIQ request(s). Therefore, if more than one interrupt sources are classified as FIQ the FIQ interrupt service routine must read VICFIQStatus to decide based on this content what to do and how to process the interrupt request. However, it is recommended that only one interrupt source should be classified as FIQ. Classifying more than one interrupt sources as FIQ will increase the interrupt latency.

Following the completion of the desired interrupt service routine, clearing of the interrupt flag on the peripheral level will propagate to corresponding bits in VIC registers (VICRawIntr, VICFIQStatus and VICIRQStatus). Also, before the next interrupt can be serviced, it is necessary that write is performed into the VICVectAddr register before the return from interrupt is executed. This write will clear the respective interrupt flag in the internal interrupt priority hardware.

In order to disable the interrupt at the VIC you need to clear corresponding bit in the VICIntEnClr register, which in turn clears the related bit in the VICIntEnable register. This also applies to the VICSoftInt and VICSoftIntClear in which VICSoftIntClear will clear the respective bits in VICSoftInt. For example, if VICSoftInt=0x0000 0005 and bit 0 has to be cleared, VICSoftIntClear=0x0000 0001 will acomplish this. Before the new clear operation on the same bit in VICSoftInt using writing into VICSoftIntClear is performed in the future, VICSoftIntClear=0x0000 0000 must be assigned. Therefore writing 1 to any bit in Clear register will have one-time-effect in the destination register.

If the watchdog is enabled for interrupt on underflow or invalid feed sequence only then there is no way of clearing the interrupt. The only way you could perform return from interrupt is by disabling the interrupt at the VIC(using VICIntEnClr).

Example:

Assuming that UART0 and SPI0 are generating interrupt requests that are classified as vectored IRQs (UART0 being on the higher level than SPI0), while UART1 and I^2C are generating non-vectored IRQs, the following could be one possibility for VIC setup:

```
VICIntSelect = 0x0000 0000(SPI0, I2C, UART1 and UART0 are IRQ => bit10, bit9, bit7  and bit6=0)
VICIntEnable = 0x0000 06C0(SPI0, I2C, UART1 and UART0 are enabled interrupts => bit10, bit9, bit 7 and bit6=1)
VICDefVectAddr = 0x…       (holds address at what routine for servicing non-vectored IRQs (i.e. UART1 and I2C) starts)
VICVectAddr0 = 0x…         (holds address where UART0 IRQ service routine starts)
VICVectAddr1 = 0x…         (holds address where SPI0 IRQ service routine starts)
VICVectCntl0 = 0x0000 0026(interrupt source with index 6 (UART0) is enabled as the one with priority 0 (the highest))
VICVectCntl1 = 0x0000 002A(interrupt source with index 10 (SPI0) is enabled as the one with priority 1)
```

After any of IRQ requests (SPI0, I2C, UART0 or UART1) is made, microcontroller will redirect code execution to the address specified at location 0x00000018. For vectored and non-vectored IRQ's the following instruction could be placed at 0x18:

```
LDR pc,[pc,#-0xFF0]
```

This instruction loads PC with the address that is present in VICVectAddr register.

In case UART0 request has been made, VICVectAddr will be identical to VICVectAddr0, while in case SPI0 request has been made value from VICVectAddr1 will be found here. If neither UART0 nor SPI0 have generated IRQ request but UART1 and/or I^2C were the reason, content of VICVectAddr will be identical to VICDefVectAddr.

7. PIN CONFIGURATION

LPC2119/2129/2194 PINOUT

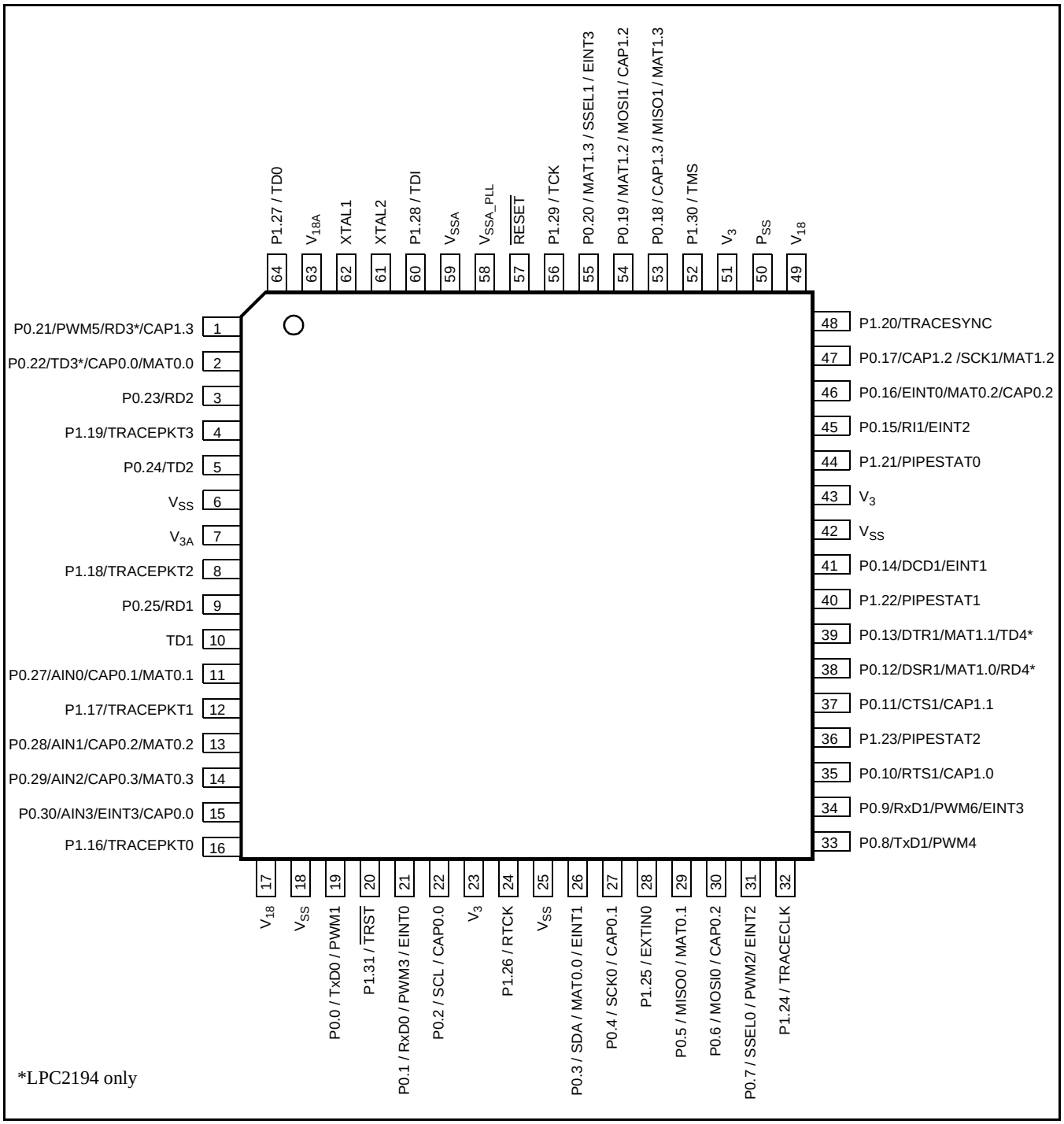


Figure 20: LPC2119/2129/2194 64-pin package

PIN DESCRIPTION FOR LPC2119/2129/2194

Pin description for LPC2119/2129/2194 and a brief of corresponding functions are shown in the following table.

Table 55: Pin description for LPC2119/2129/2194

Pin Name	LQFP64 Pin #	Type	Description		
P0.0 to P0.31		I/O	**Port 0:** Port 0 is a 32-bit bi-directional I/O port with individual direction controls for each bit. The operation of port 0 pins depends upon the pin function selected via the Pin Connect Block. Pins 26 and 31 of port 0 are not available. **Note:** All Port 0 pins excluding those that can be used as A/D inputs (P0.27, P0.28, P0.29 and P0.30) are functionally 5V tolerant. If the A/D converter is not used at all, pins associated with A/D inputs can be used as 5V tolerant digital IO pins. See "A/D Converter" chapter for A/D input pin voltage considerations.		
	19	O O	**P0.0**	**TxD0** **PWM1**	Transmitter output for UART0. Pulse Width Modulator output 1.
	21	I O I	**P0.1**	**RxD0** **PWM3** **EINT0**	Receiver input for UART0. Pulse Width Modulator output 3. External interrupt 0 input.
	22	I/O I	**P0.2**	**SCL** **CAP0.0**	I^2C clock input/output. Open drain output (for I^2C compliance). Capture input for TIMER0, channel 0.
	26	I/O O	**P0.3**	**SDA** **MAT0.0** **EINT1**	I^2C data input/output. Open drain output (for I^2C compliance). Match output for TIMER0, channel 0. External interrupt 1 input.
	27	I/O I	**P0.4**	**SCK0** **CAP0.1**	Serial Clock for SPI0. SPI clock output from master or input to slave. Capture input for TIMER0, channel 1.
	29	I/O O	**P0.5**	**MISO0** **MAT0.1**	Master In Slave Out for SPI0. Data input to SPI master or data output from SPI slave. Match output for TIMER0, channel 1.
	30	I/O I	**P0.6**	**MOSI0** **CAP0.2**	Master Out Slave In for SPI0. Data output from SPI master or data input to SPI slave. Capture input for TIMER0, channel 2.
	31	I O I	**P0.7**	**SSEL0** **PWM2** **EINT2**	Slave Select for SPI0. Selects the SPI interface as a slave. Pulse Width Modulator output 2. External interrupt 2 input.
	33	O O	**P0.8**	**TxD1** **PWM4**	Transmitter output for UART1. Pulse Width Modulator output 4.
	34	I O I	**P0.9**	**RxD1** **PWM6** **EINT3**	Receiver input for UART1. Pulse Width Modulator output 6. External interrupt 3 input.

Table 55: Pin description for LPC2119/2129/2194

Pin Name	LQFP64 Pin #	Type	Description		
	35	O	**P0.10**	**RTS1**	Request to Send output for UART1.
		I		**CAP1.0**	Capture input for TIMER1, channel 0.
	37	I	**P0.11**	**CTS1**	Clear to Send input for UART1.
		I		**CAP1.1**	Capture input for TIMER1, channel 1.
	38	I	**P0.12**	**DSR1**	Data Set Ready input for UART1.
		O		**MAT1.0**	Match output for TIMER1, channel 0.
		I		**RD4**	CAN4 receiver input (available in LPC2194 only).
	39	O	**P0.13**	**DTR1**	Data Terminal Ready output for UART1.
		O		**MAT1.1**	Match output for TIMER1, channel 1.
		O		**TD4**	CAN4 transmitter output (available in LPC2194 only).
	41	I	**P0.14**	**DCD1**	Data Carrier Detect input for UART1.
		I		**EINT1**	External interrupt 1 input. LOW on this pine while $\overline{\text{RESET}}$ is LOW forces on-chip boot-loader to take over control of the part after reset.
				Important:	LOW on pin P0.14 while $\overline{\text{RESET}}$ is LOW forces on-chip boot-loader to take over control of the part after reset.
	45	I	**P0.15**	**RI1**	Ring Indicator input for UART1.
		I		**EINT2**	External interrupt 2 input.
	46	I	**P0.16**	**EINT0**	External interrupt 0 input.
		O		**MAT0.2**	Match output for TIMER0, channel 2.
		I		**CAP0.2**	Capture input for TIMER0, channel 2.
	47	I	**P0.17**	**CAP1.2**	Capture input for TIMER1, channel 2.
		I/O		**SCK1**	Serial Clock for SPI1. SPI clock output from master or input to slave.
		O		**MAT1.2**	Match output for TIMER1, channel 2.
	53	I	**P0.18**	**CAP1.3**	Capture input for TIMER1, channel 3.
		I/O		**MISO1**	Master In Slave Out for SPI1. Data input to SPI master or data output from SPI slave.
		O		**MAT1.3**	Match output for TIMER1, channel 3.
	54	O	**P0.19**	**MAT1.2**	Match output for TIMER1, channel 2.
		I/O		**MOSI1**	Master Out Slave In for SPI1. Data output from SPI master or data input to SPI slave.
		O		**CAP1.2**	Capture input for TIMER1, channel 2.
	55	O	**P0.20**	**MAT1.3**	Match output for TIMER1, channel 3.
		I		**SSEL1**	Slave Select for SPI1. Selects the SPI interface as a slave.
		I		**EINT3**	External interrupt 3 input.
	1	O	**P0.21**	**PWM5**	Pulse Width Modulator output 5.
		I		**RD3**	CAN3 receiver input (available in LPC2194 only).
		I		**CAP1.3**	Capture input for TIMER1, channel 3.

Table 55: Pin description for LPC2119/2129/2194

Pin Name	LQFP64 Pin #	Type	Description		
		O	**P0.22**	**TD3**	CAN3 transmitter output (available in LPC2194 only)
	2	I		**CAP0.0**	Capture input for TIMER0, channel 0.
		O		**MAT0.0**	Match output for TIMER0, channel 0.
	3	I	**P0.23**	**RD2**	CAN2 receiver input.
	5	O	**P0.24**	**TD2**	CAN2 transmitter output.
	9	I	**P0.25**	**RD1**	CAN1 receiver input.
		I	**P0.27**	**AIN0**	A/D converter, input 0. This analog input is always connected to its pin.
	11	I		**CAP0.1**	Capture input for TIMER0, channel 1.
		O		**MAT0.1**	Match output for TIMER0, channel 1.
		I	**P0.28**	**AIN1**	A/D converter, input 1. This analog input is always connected to its pin.
	13	I		**CAP0.2**	Capture input for TIMER0, channel 2.
		O		**MAT0.2**	Match output for TIMER0, channel 2.
		I	**P0.29**	**AIN2**	A/D converter, input 2. This analog input is always connected to its pin.
	14	I		**CAP0.3**	Capture input for TIMER0, channel 3.
		O		**MAT0.3**	Match output for TIMER0, channel 3.
		I	**P0.30**	**AIN3**	A/D converter, input 3. This analog input is always connected to its pin.
	15	I		**EINT3**	External interrupt 3 input.
		I		**CAP0.0**	Capture input for TIMER0, channel 0.
P1.16 to P1.31		I/O	**Port 1:** Port 1 is a 32-bit bi-directional I/O port with individual direction controls for each bit. The operation of port 1 pins depends upon the pin function selected via the Pin Connect Block. Only pins 16 through 31 of port 1 are available. **Note:** All Port 1 pins are 5V tolerant with built-in pull-up resistor that sets input level to high when corresponding pin is used as input.		
	16	O	**P1.16**	**TRACEPKT0**	Trace Packet, bit 0. Standard I/O port with internal pull-up.
	12	O	**P1.17**	**TRACEPKT1**	Trace Packet, bit 1. Standard I/O port with internal pull-up.
	8	O	**P1.18**	**TRACEPKT2**	Trace Packet, bit 2. Standard I/O port with internal pull-up.
	4	O	**P1.19**	**TRACEPKT3**	Trace Packet, bit 3. Standard I/O port with internal pull-up.
	48	O	**P1.20**	**TRACESYNC**	Trace Synchronization. Standard I/O port with internal pull-up. LOW on this pin while $\overline{\text{RESET}}$ is LOW enables pins P1.25:16 to operate as a Trace port after reset. **Important:** LOW on pin P1.20 while $\overline{\text{RESET}}$ is LOW enables pins P1.25:16 to operate as a Trace port after reset.

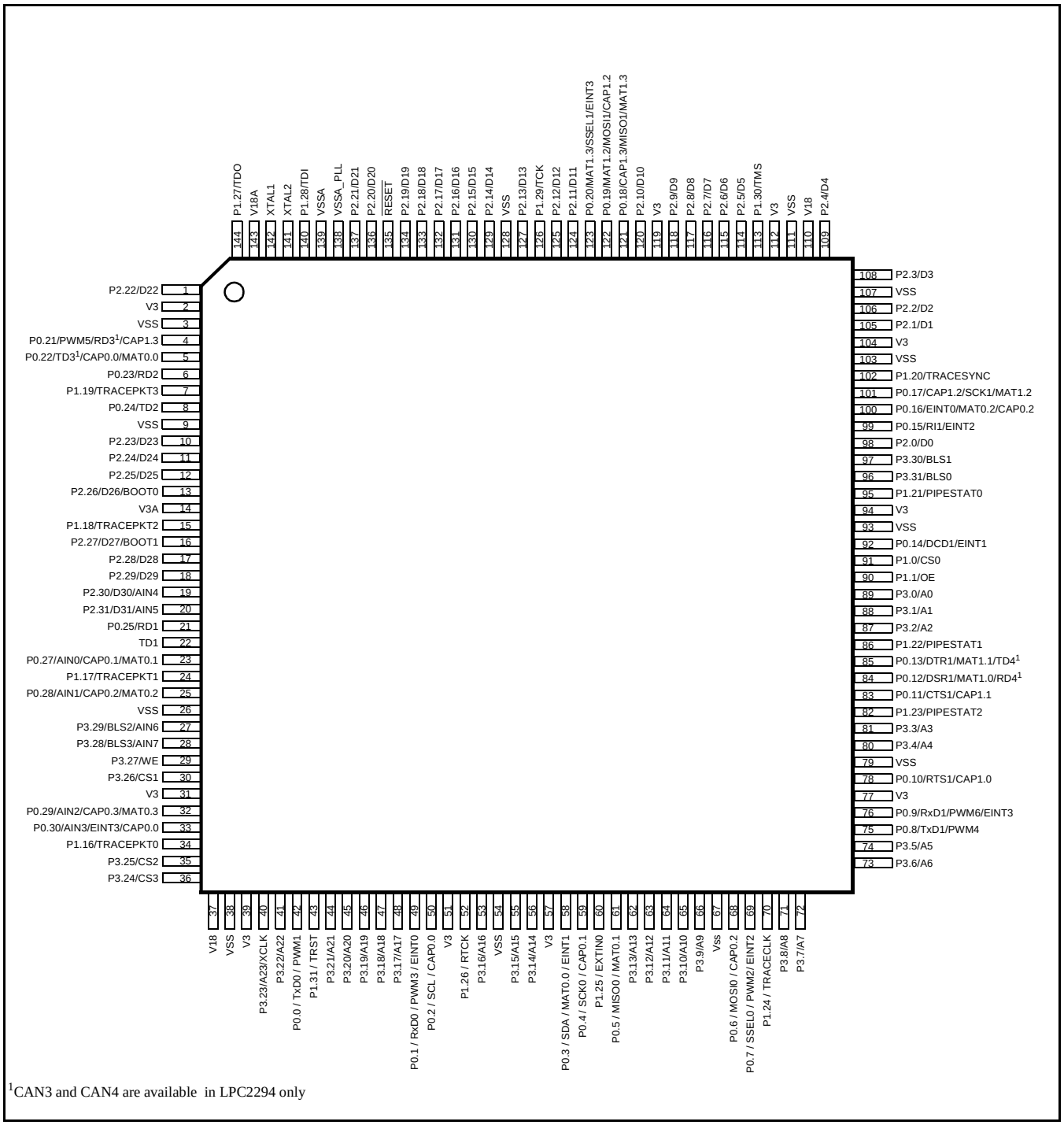
Table 55: Pin description for LPC2119/2129/2194

Pin Name	LQFP64 Pin #	Type	Description
	44	O	**P1.21** **PIPESTAT0** Pipeline Status, bit 0. Standard I/O port with internal pull-up.
	40	O	**P1.22** **PIPESTAT1** Pipeline Status, bit 1. Standard I/O port with internal pull-up.
	36	O	**P1.23** **PIPESTAT2** Pipeline Status, bit 2. Standard I/O port with internal pull-up.
	32	O	**P1.24** **TRACECLK** Trace Clock. Standard I/O port with internal pull-up.
	28	I	**P1.25** **EXTIN0** External Trigger Input. Standard I/O with internal pull-up.
	24	I/O	**P1.26** **RTCK** Returned Test Clock output. Extra signal added to the JTAG port. Assists debugger synchronization when processor frequency varies. Bi-directional pin with internal pullup. LOW on this pin while $\overline{\text{RESET}}$ is LOW enables pins P1.31:26 to operate as a Debug port after reset. **Important:** LOW on pin P1.26 while $\overline{\text{RESET}}$ is LOW enables pins P1.31:26 to operate as a Debug port after reset.
	64	O	**P1.27** **TDO** Test Data out for JTAG interface.
	60	I	**P1.28** **TDI** Test Data in for JTAG interface.
	56	I	**P1.29** **TCK** Test Clock for JTAG interface.
	52	I	**P1.30** **TMS** Test Mode Select for JTAG interface.
	20	I	**P1.31** **$\overline{\text{TRST}}$** Test Reset for JTAG interface.
TD1	10	O	**TD1**:CAN1 transmitter output. Pin is 5 V tolerant with built-in pull-up.
$\overline{\text{RESET}}$	57	I	**External Reset input**: A LOW on this pin resets the device, causing I/O ports and peripherals to take on their default states, and processor execution to begin at address 0. TTL with hysteresis, 5V tolerant.
XTAL1	62	I	Input to the oscillator circuit and internal clock generator circuits.
XTAL2	61	O	Output from the oscillator amplifier.
V_{SS}	6, 18, 25, 42, 50	I	**Ground:** 0V reference.
V_{SSA}	59	I	**Analog Ground:** 0V reference. This should nominally be the same voltage as $V_{SS,}$ but should be isolated to minimize noise and error.
V_{SSA_PLL}	58	I	**PLL Analog Ground:** 0V reference. This should nominally be the same voltage as $V_{SS,}$ but should be isolated to minimize noise and error.

Table 55: Pin description for LPC2119/2129/2194

Pin Name	LQFP64 Pin #	Type	Description
V_{18}	17, 49	I	**1.8V Core Power Supply:** This is the power supply voltage for internal circuitry.
V_{18A}	63	I	**Analog 1.8V Core Power Supply:** This is the power supply voltage for internal circuitry. This should be nominally the same voltage as V18 but should be isolated to minimize noise and error.
V_3	23, 43, 51	I	**3.3V Pad Power Supply:** This is the power supply voltage for the I/O ports.
V_{3A}	7	I	**Analog 3.3V Pad Power Supply:** This should be nominally the same voltage as V3 but should be isolated to minimize noise and error. Level on this pin is used as a reference for AD convertor.

LPC2292/2294 PINOUT



[1]CAN3 and CAN4 are available in LPC2294 only

Figure 21: LPC2292/2294 144-pin package

PIN DESCRIPTION FOR LPC2292/2294

Pin description for LPC2292/2294 and a brief of corresponding functions are shown in the following table.Pin Description

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description
P0.0 to P0.31	42,49,50,58,59,61,68,69,75,76,78,83-85,92,99,100,101,121-123,4-6,8,21,23,25,32,33	I/O	**Port 0:** Port 0 is a 32-bit bi-directional I/O port with individual direction controls for each bit. The operation of port 0 pins depends upon the pin function selected via the Pin Connect Block. Pins 26 and 31 of Port 0 are not available. **Note:** All Port 0 pins excluding those that can be used as A/D inputs (P0.27, P0.28, P0.29 and P0.30) are functionally 5V tolerant. If the A/D converter is not used at all, pins associated with A/D inputs can be used as 5V tolerant digital IO pins. See "A/D Converter" chapter for A/D input pin voltage considerations.
	42	O O	**P0.0** **TxD0** Transmitter output for UART0. **PWM1** Pulse Width Modulator output 1.
	49	I O I	**P0.1** **RxD0** Receiver input for UART0. **PWM3** Pulse Width Modulator output 3. **EINT0** External interrupt 0 input.
	50	I/O I	**P0.2** **SCL** I^2C clock input/output. Open drain output (for I^2C compliance). **CAP0.0** Capture input for TIMER0, channel 0.
	58	I/O O I	**P0.3** **SDA** I^2C data input/output. Open drain output (for I^2C compliance). **MAT0.0** Match output for TIMER0, channel 0. **EINT1** External interrupt 1 input.
	59	I/O I	**P0.4** **SCK0** Serial Clock for SPI0. SPI clock output from master or input to slave. **CAP0.1** Capture input for TIMER0, channel 1.
	61	I/O O	**P0.5** **MISO0** Master In Slave Out for SPI0. Data input to SPI master or data output from SPI slave. **MAT0.1** Match output for TIMER0, channel 1.
	68	I/O I	**P0.6** **MOSI0** Master Out Slave In for SPI0. Data output from SPI master or data input to SPI slave. **CAP0.2** Capture input for TIMER0, channel 2.
	69	I O I	**P0.7** **SSEL0** Slave Select for SPI0. Selects the SPI interface as a slave. **PWM2** Pulse Width Modulator output 2. **EINT2** External interrupt 2 input.
	75	O O	**P0.8** **TxD1** Transmitter output for UART1. **PWM4** Pulse Width Modulator output 4.
	76	I O I	**P0.9** **RxD1** Receiver input for UART1. **PWM6** Pulse Width Modulator output 6. **EINT3** External interrupt 3 input.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description		
	78	O	**P0.10**	**RTS1**	Request to Send output for UART1.
		I		**CAP1.0**	Capture input for TIMER1, channel 0.
	83	I	**P0.11**	**CTS1**	Clear to Send input for UART1.
		I		**CAP1.1**	Capture input for TIMER1, channel 1.
	84	I	**P0.12**	**DSR1**	Data Set Ready input for UART1.
		O		**MAT1.0**	Match output for TIMER1, channel 0.
		I		**RD4**	CAN4 receiver input (available in LPC2294 only).
	85	O	**P0.13**	**DTR1**	Data Terminal Ready output for UART1.
		O		**MAT1.1**	Match output for TIMER1, channel 1.
		O		**TD4**	CAN4 transmitter output (available in LPC2294 only).
	92	I	**P0.14**	**DCD1**	Data Carrier Detect input for UART1.
		I		**EINT1**	External interrupt 1 input. LOW on this pin while $\overline{\text{RESET}}$ is LOW forces on-chip boot-loader to take over control of the part after reset.
				Important:	LOW on pin P0.14 while $\overline{\text{RESET}}$ is LOW forces on-chip boot-loader to take over control of the part after reset.
	99	I	**P0.15**	**RI1**	Ring Indicator input for UART1.
		I		**EINT2**	External interrupt 2 input.
	100	I	**P0.16**	**EINT0**	External interrupt 0 input.
		O		**MAT0.2**	Match output for TIMER0, channel 2.
		I		**CAP0.2**	Capture input for TIMER0, channel 2.
	101	I	**P0.17**	**CAP1.2**	Capture input for TIMER1, channel 2.
		I/O		**SCK1**	Serial Clock for SPI1. SPI clock output from master or input to slave.
		O		**MAT1.2**	Match output for TIMER1, channel 2.
	121	I	**P0.18**	**CAP1.3**	Capture input for TIMER1, channel 3.
		I/O		**MISO1**	Master In Slave Out for SPI1. Data input to SPI master or data output from SPI slave.
		O		**MAT1.3**	Match output for TIMER1, channel 3.
	122	O	**P0.19**	**MAT1.2**	Match output for TIMER1, channel 2.
		I/O		**MOSI1**	Master Out Slave In for SPI1. Data output from SPI master or data input to SPI slave.
		O		**CAP1.2**	Capture input for TIMER1, channel 2.
	123	O	**P0.20**	**MAT1.3**	Match output for TIMER1, channel 3.
		I		**SSEL1**	Slave Select for SPI1. Selects the SPI interface as a slave.
		I		**EINT3**	External interrupt 3 input.
	4	O	**P0.21**	**PWM5**	Pulse Width Modulator output 5.
		I		**RD3**	CAN3 receiver input (available in LPC2294 only).
		I		**CAP1.3**	Capture input for TIMER1, channel 3.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description		
	5	O	**P0.22**	**TD3**	CAN3 transmitter output (available in LPC2294 only).
		I		**CAP0.0**	Capture input for TIMER0, channel 0.
		O		**MAT0.0**	Match output for TIMER0, channel 0.
	6	I	**P0.23**	**RD2**	CAN2 receiver input.
	8	O	**P0.24**	**TD2**	CAN2 transmitter output.
	21	I	**P0.25**	**RD1**	CAN1 receiver input.
	23	I	**P0.27**	**AIN0**	A/D converter, input 0. This analog input is always connected to its pin.
		I		**CAP0.1**	Capture input for TIMER0, channel 1.
		O		**MAT0.1**	Match output for TIMER0, channel 1.
	25	I	**P0.28**	**AIN1**	A/D converter, input 1. This analog input is always connected to its pin.
		I		**CAP0.2**	Capture input for TIMER0, channel 2.
		O		**MAT0.2**	Match output for TIMER0, channel 2.
	32	I	**P0.29**	**AIN2**	A/D converter, input 2. This analog input is always connected to its pin.
		I		**CAP0.3**	Capture input for TIMER0, channel 3.
		O		**MAT0.3**	Match output for TIMER0, channel 3.
	33	I	**P0.30**	**AIN3**	A/D converter, input 3. This analog input is always connected to its pin.
		I		**EINT3**	External interrupt 3 input.
		I		**CAP0.0**	Capture input for TIMER0, channel 0.
P1.0 to P1.31	91,90,34,24,15,7,102,95,86,82,70,60,52,144,140,126,113,43	I/O	**Port 1:** Port 1 is a 32-bit bi-directional I/O port with individual direction controls for each bit. The operation of port 1 pins depends upon the pin function selected via the Pin Connect Block. Pins 2 through 15 of port 1 are not available. **Note:** All Port 1 pins are 5V tolerant with built-in pull-up resistor that sets input level to high when corresponding pin is used as input.		
	91	O	**P1.0**	**CS0**	Low-active Chip Select 0 signal. (Bank 0 addresses range 8000 0000 - 80FF FFFF)
	90	O	**P1.1**	**OE**	Low -active Output Enable signal.
	34	O	**P1.16**	**TRACEPKT0**	Trace Packet, bit 0. Standard I/O port with internal pull-up.
	24	O	**P1.17**	**TRACEPKT1**	Trace Packet, bit 1. Standard I/O port with internal pull-up.
	15	O	**P1.18**	**TRACEPKT2**	Trace Packet, bit 2. Standard I/O port with internal pull-up.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description
	7	O	**P1.19** **TRACEPKT3** Trace Packet, bit 3. Standard I/O port with internal pull-up.
	102	O	**P1.20** **TRACESYNC** Trace Synchronization. Standard I/O port with internal pull-up. LOW on this pin while $\overline{\text{RESET}}$ is LOW enables pins P1.25:16 to operate as a Trace port after reset. **Important:** LOW on pin P1.20 while $\overline{\text{RESET}}$ is LOW enables pins P1.25:16 to operate as a Trace port after reset.
	95	O	**P1.21** **PIPESTAT0** Pipeline Status, bit 0. Standard I/O port with internal pull-up.
	86	O	**P1.22** **PIPESTAT1** Pipeline Status, bit 1. Standard I/O port with internal pull-up.
	82	O	**P1.23** **PIPESTAT2** Pipeline Status, bit 2. Standard I/O port with internal pull-up.
	70	O	**P1.24** **TRACECLK** Trace Clock. Standard I/O port with internal pull-up.
	60	I	**P1.25** **EXTIN0** External Trigger Input. Standard I/O with internal pull-up.
	52	I/O	**P1.26** **RTCK** Returned Test Clock output. Extra signal added to the JTAG port. Assists debugger synchronization when processor frequency varies. Bi-directional pin with internal pullup. LOW on this pin while $\overline{\text{RESET}}$ is LOW enables pins P1.31:26 to operate as a Debug port after reset. **Important:** LOW on pin P1.26 while $\overline{\text{RESET}}$ is LOW enables pins P1.31:26 to operate as a Debug port after reset.
	144	O	**P1.27** **TDO** Test Data out for JTAG interface.
	140	I	**P1.28** **TDI** Test Data in for JTAG interface.
	126	I	**P1.29** **TCK** Test Clock for JTAG interface.
	113	I	**P1.30** **TMS** Test Mode Select for JTAG interface.
	43	I	**P1.31** **$\overline{\text{TRST}}$** Test Reset for JTAG interface.
P2.0 to P2.31	98,105,106,108,109,114-118,120,124,125,127,129-134,136,137,1,10-13,16-20	I/O	**Port 2:** Port 2 is a 32-bit bi-directional I/O port with individual direction controls for each bit. The operation of port 2 pins depends upon the pin function selected via the Pin Connect Block. **Note:** All Port 2 pins excluding those that can be used as A/D inputs (P2.30 and P2.31) are functionally 5V tolerant. Port 2 pin configured to perform an input function will use built-in pull-up resistor to set the default input level to high. If the A/D converter is not used at all, pins associated with A/D inputs can be used as 5V tolerant digital IO pins. See "A/D Converter" chapter for A/D input pin voltage considerations.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description		
	98	I/O	**P2.0**	**D0**	External memory data line 0.
	105	I/O	**P2.1**	**D1**	External memory data line 1.
	106	I/O	**P2.2**	**D2**	External memory data line 2.
	108	I/O	**P2.3**	**D3**	External memory data line 3.
	109	I/O	**P2.4**	**D4**	External memory data line 4.
	114	I/O	**P2.5**	**D5**	External memory data line 5.
	115	I/O	**P2.6**	**D6**	External memory data line 6.
	116	I/O	**P2.7**	**D7**	External memory data line 7.
	117	I/O	**P2.8**	**D8**	External memory data line 8.
	118	I/O	**P2.9**	**D9**	External memory data line 9.
	120	I/O	**P2.10**	**D10**	External memory data line 10.
	124	I/O	**P2.11**	**D11**	External memory data line 11.
	125	I/O	**P2.12**	**D12**	External memory data line 12.
	127	I/O	**P2.13**	**D13**	External memory data line 13.
	129	I/O	**P2.14**	**D14**	External memory data line 14.
	130	I/O	**P2.15**	**D15**	External memory data line 15.
	131	I/O	**P2.16**	**D16**	External memory data line 16.
	132	I/O	**P2.17**	**D17**	External memory data line 17.
	133	I/O	**P2.18**	**D18**	External memory data line 18.
	134	I/O	**P2.19**	**D19**	External memory data line 19.
	136	I/O	**P2.20**	**D20**	External memory data line 20.
	137	I/O	**P2.21**	**D21**	External memory data line 21.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description		
	1	I/O	**P2.22**	**D22**	External memory data line 22.
	10	I/O	**P2.23**	**D23**	External memory data line 23.
	11	I/O	**P2.24**	**D24**	External memory data line 24.
	12	I/O	**P2.25**	**D25**	External memory data line 25.
	13	I/O I	**P2.26**	**D26** **BOOT0**	External memory data line 26. While RESET is low, together with BOOT1 controls booting and internal operation. Internal pullup ensures high state if pin is left unconnected.
	16	I/O I	**P2.27**	**D27** **BOOT1**	External memory data line 27. While RESET is low, together with BOOT0 controls booting and internal operation. Internal pullup ensures high state if pin is left unconnected. BOOT1:0=00 selects 8-bit memory on CS0 for boot. BOOT1:0=01 selects 16-bit memory on CS0 for boot. BOOT1:0=10 selects 32-bit memory on CS0 for boot. BOOT1:0=11 selects Internal Flash memory.
	17	I/O	**P2.28**	**D28**	External memory data line 28.
	18	I/O	**P2.29**	**D29**	External memory data line 29.
	19	I/O I	**P2.30**	**D30** **AIN4**	External memory data line 30. A/D converter, input 4. This analog input is always connected to its pin.
	20	I/O I	**P2.31**	**D31** **AIN5**	External memory data line 31. A/D converter, input 5. This analog input is always connected to its pin.
P3.0 to P3.31	89-87,81,80,74-71,66-62,56,55,53,48-44,41,40,36,35,30-27,97,96	I/O	**Port 3:** Port 3 is a 32-bit bi-directional I/O port with individual direction controls for each bit. The operation of port 3 pins depends upon the pin function selected via the Pin Connect Block. **Note:** All Port 3 pins excluding those that can be used as A/D inputs (P3.28 and P3.29) are functionally 5V tolerant. Port 3 pin configured to perform an input function will use built-in pull-up resistor to set the default input level to high. If the A/D converter is not used at all, pins associated with A/D inputs can be used as 5V tolerant digital IO pins. See "A/D Converter" chapter for A/D input pin voltage considerations.		
	89	O	**P3.0**	**A0**	External memory address line 0.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description		
	88	O	**P3.1**	**A1**	External memory address line 1.
	87	O	**P3.2**	**A2**	External memory address line 2.
	81	O	**P3.3**	**A3**	External memory address line 3.
	80	O	**P3.4**	**A4**	External memory address line 4.
	74	O	**P3.5**	**A5**	External memory address line 5.
	73	O	**P3.6**	**A6**	External memory address line 6.
	72	O	**P3.7**	**A7**	External memory address line 7.
	71	O	**P3.8**	**A8**	External memory address line 8.
	66	O	**P3.9**	**A9**	External memory address line 9.
	65	O	**P3.10**	**A10**	External memory address line 10.
	64	O	**P3.11**	**A11**	External memory address line 11.
	63	O	**P3.12**	**A12**	External memory address line 12.
	62	O	**P3.13**	**A13**	External memory address line 13.
	56	O	**P3.14**	**A14**	External memory address line 14.
	55	O	**P3.15**	**A15**	External memory address line 15.
	53	O	**P3.16**	**A16**	External memory address line 16.
	48	O	**P3.17**	**A17**	External memory address line 17.
	47	O	**P3.18**	**A18**	External memory address line 18.
	46	O	**P3.19**	**A19**	External memory address line 19.
	45	O	**P3.20**	**A20**	External memory address line 20.
	44	O	**P3.21**	**A21**	External memory address line 21.
	41	O	**P3.22**	**A22**	External memory address line 22.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description
	40	I/O O	**P3.23** **A23** **XCLK** External memory address line 23. Clock output.
	36	O	**P3.24** **CS3** Low-active Chip Select 3 signal. (Bank 3 addresses range 8300 0000 - 83FF FFFF)
	35	O	**P3.25** **CS2** Low-active Chip Select 2 signal. (Bank 2 addresses range 8200 0000 - 82FF FFFF)
	30	O	**P3.26** **CS1** Low-active Chip Select 1 signal. (Bank 1 addresses range 8100 0000 - 81FF FFFF)
	29	O	**P3.27** **WE** Low-active Write enable signal.
	28	O I	**P3.28** **BLS3** **AIN7** Low-active Byte Lane Select signal (Bank 3). A/D converter, input 7. This analog input is always connected to its pin.
	27	O I	**P3.29** **BLS2** **AIN6** Low-active Byte Lane Select signal (Bank 2). A/D converter, input 6. This analog input is always connected to its pin.
	97	O	**P3.30** **BLS1** Low-active Byte Lane Select signal (Bank 1).
	96	O	**P3.31** **BLS0** Low-active Byte Lane Select signal (Bank 0).
TD1	22	O	**TD1**:CAN1 transmitter output.Pin is 5 V tolerant with built-in pull-up.
$\overline{\text{RESET}}$	135	I	**External Reset input**: A LOW on this pin resets the device, causing I/O ports and peripherals to take on their default states, and processor execution to begin at address 0. TTL with hysteresis, 5V tolerant.
XTAL1	142	I	Input to the oscillator circuit and internal clock generator circuits.
XTAL2	141	O	Output from the oscillator amplifier.
V_{SS}	3, 9, 26, 38, 54, 67, 79, 93, 103, 107, 111, 128	I	**Ground:** 0V reference.
V_{SSA}	139	I	**Analog Ground:** 0V reference. This should nominally be the same voltage as $V_{SS,}$ but should be isolated to minimize noise and error.
V_{SSA_PLL}	138	I	**PLL Analog Ground:** 0V reference. This should nominally be the same voltage as $V_{SS,}$ but should be isolated to minimize noise and error.

Table 56: Pin description for LPC2292/2294

Pin Name	LQFP144 Pin #	Type	Description
V18	37, 110	I	**1.8V Core Power Supply:** This is the power supply voltage for internal circuitry.
V18A	143	I	**Analog 1.8V Core Power Supply:** This is the power supply voltage for internal circuitry. This should be nominally the same voltage as V18 but should be isolated to minimize noise and error.
V3	2, 31, 39, 51, 57, 77, 94, 104, 112, 119	I	**3.3V Pad Power Supply:** This is the power supply voltage for the I/O ports.
V3A	14	I	**Analog 3.3V Pad Power Supply:** This should be nominally the same voltage as V3 but should be isolated to minimize noise and error.

8. PIN CONNECT BLOCK

FEATURES

- Allows individual pin configuration

APPLICATIONS

The purpose of the Pin Connect Block is to configure the microcontroller pins to the desired functions.

DESCRIPTION

The pin connect block allows selected pins of the microcontroller to have more than one function. Configuration registers control the multiplexers to allow connection between the pin and the on chip peripherals.

Peripherals should be connected to the appropriate pins prior to being activated, and prior to any related interrupt(s) being enabled. Activity of any enabled peripheral function that is not mapped to a related pin should be considered undefined.

Selection of a single function on a port pin completely excludes all other functions otherwise available on the same pin.

The only partial exception from the above rule of exclusion is the case of inputs to the A/D converter. Regardless of the function that is selected for the port pin that also hosts the A/D input, this A/D input can be read at any time and variations of the voltage level on this pin will be reflected in the A/D readings. However, valid analog reading(s) can be obtained if and only if the function of an analog input is selected. Only in this case proper interface circuit is active in between the physical pin and the A/D module. In all other cases, a part of digital logic necessary for the digital function to be performed will be active, and will disrupt proper behavior of the A/D.

REGISTER DESCRIPTION

The Pin Control Module contains 2 registers as shown in Table 57. below.

Table 57: Pin Connect Block Register Map

Name	Description	Access	Reset Value	Address
PINSEL0	Pin function select register 0	Read/Write	0x0000 0000	0xE002C000
PINSEL1	Pin function select register 1	Read/Write	0x1540 0000	0xE002C004
PINSEL2	Pin function select register 2	Read/Write	See Table 63 and Table 64	0xE002C014

Pin Function Select Register 0 (PINSEL0 - 0xE002C000)

The PINSEL0 register controls the functions of the pins as per the settings listed in Table 65. The direction control bit in the IO0DIR register is effective only when the GPIO function is selected for a pin. For other functions, direction is controlled automatically.

Table 58: Pin Function Select Register 0 for LPC2119/2129/2292 (PINSEL0 - 0xE002C000)

PINSEL0	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
1:0	P0.0	GPIO Port 0.0	TxD (UART0)	PWM1	Reserved	00
3:2	P0.1	GPIO Port 0.1	RxD (UART0)	PWM3	EINT0	00
5:4	P0.2	GPIO Port 0.2	SCL (I^2C)	Capture 0.0 (TIMER0)	Reserved	00
7:6	P0.3	GPIO Port 0.3	SDA (I^2C)	Match 0.0 (TIMER0)	EINT1	00
9:8	P0.4	GPIO Port 0.4	SCK (SPI0)	Capture 0.1 (TIMER0)	Reserved	00
11:10	P0.5	GPIO Port 0.5	MISO (SPI0)	Match 0.1 (TIMER0)	Reserved	00
13:12	P0.6	GPIO Port 0.6	MOSI (SPI0)	Capture 0.2 (TIMER0)	Reserved	00
15:14	P0.7	GPIO Port 0.7	SSEL (SPI0)	PWM2	EINT2	00
17:16	P0.8	GPIO Port 0.8	TxD UART1	PWM4	Reserved	00
19:18	P0.9	GPIO Port 0.9	RxD (UART1)	PWM6	EINT3	00
21:20	P0.10	GPIO Port 0.10	RTS (UART1)	Capture 1.0 (TIMER1)	Reserved	00
23:22	P0.11	GPIO Port 0.11	CTS (UART1)	Capture 1.1 (TIMER1)	Reserved	00
25:24	P0.12	GPIO Port 0.12	DSR (UART1)	Match 1.0 (TIMER1)	Reserved	00
27:26	P0.13	GPIO Port 0.13	DTR (UART1)	Match 1.1 (TIMER1)	Reserved	00
29:28	P0.14	GPIO Port 0.14	CD (UART1)	EINT1	Reserved	00
31:30	P0.15	GPIO Port 0.15	RI (UART1)	EINT2	Reserved	00

Table 59: Pin Function Select Register 0 for LPC2194/2294 (PINSEL0 - 0xE002C000)

PINSEL0	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
1:0	P0.0	GPIO Port 0.0	TxD (UART0)	PWM1	Reserved	00
3:2	P0.1	GPIO Port 0.1	RxD (UART0)	PWM3	EINT0	00
5:4	P0.2	GPIO Port 0.2	SCL (I^2C)	Capture 0.0 (TIMER0)	Reserved	00
7:6	P0.3	GPIO Port 0.3	SDA (I^2C)	Match 0.0 (TIMER0)	EINT1	00
9:8	P0.4	GPIO Port 0.4	SCK (SPI0)	Capture 0.1 (TIMER0)	Reserved	00
11:10	P0.5	GPIO Port 0.5	MISO (SPI0)	Match 0.1 (TIMER0)	Reserved	00
13:12	P0.6	GPIO Port 0.6	MOSI (SPI0)	Capture 0.2 (TIMER0)	Reserved	00
15:14	P0.7	GPIO Port 0.7	SSEL (SPI0)	PWM2	EINT2	00
17:16	P0.8	GPIO Port 0.8	TxD UART1	PWM4	Reserved	00
19:18	P0.9	GPIO Port 0.9	RxD (UART1)	PWM6	EINT3	00

Table 59: Pin Function Select Register 0 for LPC2194/2294 (PINSEL0 - 0xE002C000)

PINSEL0	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
21:20	P0.10	GPIO Port 0.10	RTS (UART1)	Capture 1.0 (TIMER1)	Reserved	00
23:22	P0.11	GPIO Port 0.11	CTS (UART1)	Capture 1.1 (TIMER1)	Reserved	00
25:24	P0.12	GPIO Port 0.12	DSR (UART1)	Match 1.0 (TIMER1)	RD4[1](CAN Controller 4)	00
27:26	P0.13	GPIO Port 0.13	DTR (UART1)	Match 1.1 (TIMER1)	TD4[1](CAN Controller 4)	00
29:28	P0.14	GPIO Port 0.14	CD (UART1)	EINT1	Reserved	00
31:30	P0.15	GPIO Port 0.15	RI (UART1)	EINT2	Reserved	00

[1]CAN Controller 4 is available in LPC2294 only. Fields in the table related to CAN4 have Reserved value for all other parts.

Pin Function Select Register 1 (PINSEL1 - 0xE002C004)

The PINSEL1 register controls the functions of the pins as per the settings listed in following tables. The direction control bit in the IO0DIR register is effective only when the GPIO function is selected for a pin. For other functions direction is controlled automatically.

Table 61: Pin Function Select Register 1 for LPC2119/2129/2292 (PINSEL1 - 0xE002C004)

PINSEL1	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
1:0	P0.16	GPIO Port 0.16	EINT0	Match 0.2 (TIMER0)	Capture 0.2 (TIMER0)	00
3:2	P0.17	GPIO Port 0.17	Capture 1.2 (TIMER1)	SCK (SPI1)	Match 1.2 (TIMER1)	00
5:4	P0.18	GPIO Port 0.18	Capture 1.3 (TIMER1)	MISO (SPI1)	Match 1.3 (TIMER1)	00
7:6	P0.19	GPIO Port 0.19	Match 1.2 (TIMER1)	MOSI (SPI1)	Match 1.3 (TIMER1)	00
9:8	P0.20	GPIO Port 0.20	Match 1.3 (TIMER1)	SSEL (SPI1)	EINT3	00
11:10	P0.21	GPIO Port 0.21	PWM5	Reserved	Capture 1.3 (TIMER1)	00
13:12	P0.22	GPIO Port 0.22	Reserved	Capture 0.0 (TIMER0)	Match 0.0 (TIMER0)	00
15:14	P0.23	GPIO Port 0.23	RD2 (CAN Controller 2)	Reserved	Reserved	00
17:16	P0.24	GPIO Port 0.24	TD2 (CAN Controller 2)	Reserved	Reserved	00
19:18	P0.25	GPIO Port 0.25	RD1 (CAN Controller 1)	Reserved	Reserved	00
21:20	P0.26	Reserved				00
23:22	P0.27	GPIO Port 0.27	AIN0 (A/D Converter)	Capture 0.1 (TIMER0)	Match 0.1 (TIMER0)	01
25:24	P0.28	GPIO Port 0.28	AIN1 (A/D Converter)	Capture 0.2 (TIMER0)	Match 0.2 (TIMER0)	01
27:26	P0.29	GPIO Port 0.29	AIN2 (A/D Converter)	Capture 0.3 (TIMER0)	Match 0.3 (TIMER0)	01
29:28	P0.30	GPIO Port 0.30	AIN3 (A/D Converter)	EINT3	Capture 0.0 (TIMER0)	01
31:30	P0.31	Reserved				00

Table 62: Pin Function Select Register 1 for LPC2194/2294 (PINSEL1 - 0xE002C004)

PINSEL1	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
1:0	P0.16	GPIO Port 0.16	EINT0	Match 0.2 (TIMER0)	Capture 0.2 (TIMER0)	00
3:2	P0.17	GPIO Port 0.17	Capture 1.2 (TIMER1)	SCK (SPI1)	Match 1.2 (TIMER1)	00
5:4	P0.18	GPIO Port 0.18	Capture 1.3 (TIMER1)	MISO (SPI1)	Match 1.3 (TIMER1)	00
7:6	P0.19	GPIO Port 0.19	Match 1.2 (TIMER1)	MOSI (SPI1)	Match 1.3 (TIMER1)	00
9:8	P0.20	GPIO Port 0.20	Match 1.3 (TIMER1)	SSEL (SPI1)	EINT3	00
11:10	P0.21	GPIO Port 0.21	PWM5	RD3[1](CAN Controller 3)	Capture 1.3 (TIMER1)	00
13:12	P0.22	GPIO Port 0.22	TD3[1](CAN Controller 3)	Capture 0.0 (TIMER0)	Match 0.0 (TIMER0)	00
15:14	P0.23	GPIO Port 0.23	RD2 (CAN Controller 2)	Reserved	Reserved	00
17:16	P0.24	GPIO Port 0.24	TD2 (CAN Controller 2)	Reserved	Reserved	00
19:18	P0.25	GPIO Port 0.25	RD1 (CAN Controller 1)	Reserved	Reserved	00
21:20	P0.26	Reserved				00
23:22	P0.27	GPIO Port 0.27	AIN0 (A/D Converter)	Capture 0.1 (TIMER0)	Match 0.1 (TIMER0)	01
25:24	P0.28	GPIO Port 0.28	AIN1 (A/D Converter)	Capture 0.2 (TIMER0)	Match 0.2 (TIMER0)	01
27:26	P0.29	GPIO Port 0.29	AIN2 (A/D Converter)	Capture 0.3 (TIMER0)	Match 0.3 (TIMER0)	01
29:28	P0.30	GPIO Port 0.30	AIN3 (A/D Converter)	EINT3	Capture 0.0 (TIMER0)	01
31:30	P0.31	Reserved				00

[1]CAN Controller 3 is available in LPC2294 only. Fields in the table related to CAN3 have Reserved value for all other parts.

Pin Function Select Register 2 (PINSEL2 - 0xE002C014)

The PINSEL2 register controls the functions of the pins as per the settings listed in Table 63. The direction control bit in the IO1DIR register is effective only when the GPIO function is selected for a pin. For other functions direction is controlled automatically.

Warning: use read-modify-write operation when accessing PINSEL2 register. Accidental write of 0 to bit 2 and/or bit 3 results in loss of debug and/or trace functionality! Changing of either bit 4 or bit 5 from 1 to 0 may cause an incorrect code execution!

Table 63: Pin Function Select Register 2 for LPC2119/2129/2194 (PINSEL2 - 0xE002C014)

PINSEL2	Description	Reset Value
1:0	Reserved.	00
2	When 0, pins P1.36:26 are used as GPIO pins. When 1, P1.31:26 are used as a Debug port.	$\overline{\text{P1.26/RTCK}}$
3	When 0, pins P1.25:16 are used as GPIO pins. When 1, P1.25:16 are used as a Trace port.	$\overline{\text{P1.20/}}$ $\overline{\text{TRACESYNC}}$

Table 63: Pin Function Select Register 2 for LPC2119/2129/2194 (PINSEL2 - 0xE002C014)

PINSEL2	Description	Reset Value
4:5	Reserved. **Note:** These bits must not be altered at any time. Changing them may result in an incorrect code execution.	11
6:31	Reserved.	NA

Table 64: Pin Function Select Register 2 for LPC2292/2294 (PINSEL2 - 0xE002C014)

PINSEL2	Description	Reset Value
1:0	Reserved.	00
2	When 0, pins P1.36:26 are used as GPIO pins. When 1, P1.31:26 are used as a Debug port.	$\overline{\text{P1.26}}$/RTCK
3	When 0, pins P1.25:16 are used as GPIO pins. When 1, P1.25:16 are used as a Trace port.	$\overline{\text{P1.20}}$/ TRACESYNC
5:4	Controls the use of the data bus and strobe pins: Pins P2.7:0 — 11 = P2.7:0 — 0x or 10 = D7:0 Pin P1.0 — 11 = P1.0 — 0x or 10 = CS0 Pin P1.1 — 11 = P1.1 — 0x or 10 = OE Pin P3.31 — 11 = P3.31 — 0x or 10 = BLS0 Pins P2.15:8 — 00 or 11 = P2.15:8 — 01 or 10 = D15:8 Pin P3.30 — 00 or 11 = P3.30 — 01 or 10 = BLS1 Pins P2.27:16 — 0x or 11 = P2.27:16 — 10 = D27:16 Pins P2.29:28 — 0x or 11 = P2.29:28 or Reserved — 10 = D29:28 Pins P2.31:30 — 0x or 11 = P2.31:30 or AIN5:4 — 10 = D31:30 Pins P3.29:28 — 0x or 11 = P3.29:28 or AIN6:7 — 10 = BLS2:3	BOOT1:0
6	If bits 5:4 are not 10, controls the use of pin P3.29: 0 enables P3.29, 1 enables AIN6.	1
7	If bits 5:4 are not 10, controls the use of pin P3.28: 0 enables P3.28, 1 enables AIN7.	1
8	Controls the use of pin P3.27: 0 enables P3.27, 1 enables WE.	0
10:9	Reserved.	-
11	Controls the use of pin P3.26: 0 enables P3.26, 1 enables CS1.	0
12	Reserved.	-
13	If bits 25:23 are not 111, controls the use of pin P3.23/A23/XCLK: 0 enables P3.23, 1 enables XCLK.	0
15:14	Controls the use of pin P3.25: 00 enables P3.25, 01 enables CS2, 10 and 11 are reserved values.	00
17:16	Controls the use of pin P3.24: 00 enables P3.24, 01 enables CS3, 10 and 11 are reserved values.	00
19:18	Reserved.	-
20	If bits 5:4 are not 10, controls the use of pin P2.29:28: 0 enables P2.29:28, 1 is reserved.	0
21	If bits 5:4 are not 10, controls the use of pin P2.30: 0 enables P2.30, 1 enables AIN4.	1
22	If bits 5:4 are not 10, controls the use of pin P2.31: 0 enables P2.31, 1 enables AIN5.	1
23	Controls whether P3.0/A0 is a port pin (0) or an address line (1).	1 if BOOT1:0=00 at $\overline{\text{RESET}}$=0, 0 otherwise
24	Controls whether P3.1/A1 is a port pin (0) or an address line (1).	$\overline{\text{BOOT1}}$ during Reset

Table 64: Pin Function Select Register 2 for LPC2292/2294 (PINSEL2 - 0xE002C014)

PINSEL2	Description		Reset Value
27:25	Controls the number of pins among P3.23/A23/XCLK and P3.22:2/A2.22:2 that are address lines:		000 if BOOT1:0=11 at Reset, 111 otherwise
	000 = None	100 = A11:2 are address lines.	
	001 = A3:2 are address lines.	101 = A15:2 are address lines.	
	010 = A5:2 are address lines.	110 = A19:2 are address lines.	
	011 = A7:2 are address lines.	111 = A23:2 are address lines.	
31:28	Reserved.		-

Pin Function Select Register Values

The PINSEL registers control the functions of device pins as shown below. Pairs of bits in these registers correspond to specific device pins.

Table 65: Pin Function Select Register Bits

Pinsel0 and Pinsel1 Values		Function	Value after Reset
0	0	Primary (default) function, typically GPIO Port	00
0	1	First alternate function	
1	0	Second alternate function	
1	1	Reserved	

The direction control bit in the IO0DIR/IO1DIR register is effective only when the GPIO function is selected for a pin. For other functions, direction is controlled automatically. Each derivative typically has a different pinout and therefore a different set of functions possible for each pin. Details for a specific derivative may be found in the appropriate data sheet.

BOOT CONTROL ON 144-PIN PACKAGE

In the 144-pin package only, the state of the BOOT1:0 pins, while $\overline{\text{RESET}}$ is low, controls booting and initial operation. Internal pullups in the receivers ensure high state if a pin is left unconnected. Board designers can connect weak pulldown resistors (~10 kΩ) or transistors that drive low while $\overline{\text{RESET}}$ is low, to these pins to select among the following options:

Table 66: Boot Control on BOOT1:0

BOOT1 (latched from P2.27/D27 on Reset pin rising edge only)	BOOT0 (latched from P2.26/D26 on Reset pin rising edge only)	Boot from
0	0	8-bit memory on CS0
0	1	16-bit memory on CS0
1	0	32-bit memory on CS0
1	1	Internal Flash Memory

Note that if an application enables the Watchdog Timer to Reset the part if it's not serviced, transistors driven by $\overline{\text{RESET}}$ should not be used.

9. GPIO

FEATURES

- Direction control of individual bits
- Separate control of output set and clear
- All I/O default to inputs after reset

APPLICATIONS

- General purpose I/O
- Driving LEDs, or other indicators
- Controlling off-chip devices
- Sensing digital inputs

PIN DESCRIPTION

Table 67: GPIO Pin Description

Pin Name	Type	Description
P0.0 - P0.31 P1.16 - P1.31	Input/ Output	General purpose input/output. The number of GPIOs actually available depends on the use of alternate functions.
P2.0 - P2.31 P3.0 - P3.31	Input/ Output	External bus data/address lines shared with GPIO, digital and analog functions. The number of GPIOs/digital and analog functions actually available depends on the selected bus structure. PORT2 and PORT3 are available in LPC2292/2294 only.

REGISTER DESCRIPTION

LPC2119/2129/2194 has two 32-bit General Purpose I/O ports. Total of 30 out of 32 pins are available on PORT0. PORT1 has up to 16 pins available for GPIO functions. PORT0 and PORT1 are controlled via two groups of 4 registers as shown in Table 68. LPC2292/2294 has two 32-bit additional ports, PORT2 and PORT3, and they are configured to be used either as external memory data address and data bus, or as GPIOs sharing pins with a handful of digital and analog functions. Details on PORT2 and PORT3 usage can be found in Pin Configuration and Pin Connect Block chapters.

Table 68: GPIO Register Map

Generic Name	Description	Access	Reset Value	PORT0 Address & Name	PORT1 Address & Name	PORT2 Address & Name	PORT3 Address & Name
IOPIN	GPIO Port Pin value register. The current state of the GPIO configured port pins can always be read from this register, regardless of pin direction and mode. Activity on non-GPIO configured pins will not be reflected in this register.	Read Only	NA	0xE0028000 IO0PIN	0xE0028010 IO1PIN	0xE0028020 IO2PIN	0xE0028030 IO3PIN
IOSET	GPIO Port Output set register. This register controls the state of output pins in conjunction with the IOCLR register. Writing ones produces highs at the corresponding port pins. Writing zeroes has no effect.	Read/ Write	0x0000 0000	0xE0028004 IO0SET	0xE0028014 IO1SET	0xE0028024 IO2SET	0xE0028034 IO3SET
IODIR	GPIO Port Direction control register. This register individually controls the direction of each port pin.	Read/ Write	0x0000 0000	0xE0028008 IO0DIR	0xE0028018 IO1DIR	0xE0028028 IO2DIR	0xE0028038 IO3DIR
IOCLR	GPIO Port Output clear register. This register controls the state of output pins. Writing ones produces lows at the corresponding port pins and clears the corresponding bits in the IOSET register. Writing zeroes has no effect.	Write Only	0x0000 0000	0xE002800C IO0CLR	0xE002801C IO1CLR	0xE002802C IO2CLR	0xE002803C IO3CLR

GPIO Pin Value Register (IO0PIN - 0xE0028000, IO1PIN - 0xE0028010, IO2PIN - 0xE0028020, IO3PIN - 0xE0028030)

This register provides the value of the GPIO pins. Register's value reflects any outside world influence on the GPIO configured pins only. Monitoring of non-GPIO configured port pins using IOPIN register will not be valid, since activities on non-GPIO configured pins are not indicated in the IOPIN register.

Selection of a single function on a port pin completely excludes all other functions otherwise available on the same pin.

The only partial exception from the above rule of exclusion is in the case of inputs to the A/D converter. Regardless of the function that is selected for the port pin that also hosts the A/D input, this A/D input can be read at any time and variations of the voltage level on this pin will be reflected in the A/D readings. However, valid analog reading(s) can be obtained if and only if the function of an analog input is selected. Only in this case proper interface circuit is active in between the physical pin and the A/D module. In all other cases, a part of digital logic necessary for the digital function to be performed will be active, and will disrupt proper behavior of the A/D.

Table 69: GPIO Pin Value Register (IO0PIN - 0xE0028000, IO1PIN - 0xE0028010, IO2PIN - 0xE0028020, IO3PIN - 0xE0028030)

IOPIN	Description	Value after Reset
31:0	GPIO pin value bits. Bit 0 in IO0PIN corresponds to P0.0 ... Bit 31 in IO0PIN corresponds to P0.31	Undefined

GPIO Output Set Register (IO0SET - 0xE0028004, IO1SET - 0xE0028014, IO2SET - 0xE0028024, IO3SET - 0xE0028034)

This register is used to produce a HIGH level output at the port pins if they are configured as GPIO in an OUTPUT mode. Writing 1 produces a HIGH level at the corresponding port pins. Writing 0 has no effect. If any pin is configured as an input or a secondary function, writing to IOSET has no effect.

Reading the IOSET register returns the value of this register, as determined by previous writes to IOSET and IOCLR (or IOPIN as noted above). This value does not reflect the effect of any outside world influence on the I/O pins.

Table 70: GPIO Output Set Register (IO0SET - 0xE0028004, IO1SET - 0xE0028014, IO2SET - 0xE0028024, IO3SET - 0xE0028034)

IOSET	Description	Value after Reset
31:0	Output value SET bits. Bit 0 in IO0SET corresponds to P0.0 ... Bit 31 in IO0SET corresponds to P0.31	0

GPIO Output Clear Register

(IO0CLR - 0xE002800C, IO1CLR - 0xE002801C, IO2CLR - 0xE002802C, IO3CLR - 0xE002803C)

This register is used to produce a LOW level at port pins if they are configured as GPIO in an OUTPUT mode. Writing 1 produces a LOW level at the corresponding port pins and clears the corresponding bits in the IOSET register. Writing 0 has no effect. If any pin is configured as an input or a secondary function, writing to IOCLR has no effect.

Table 71: GPIO Output Clear Register (IO0CLR - 0xE002800C, IO1CLR - 0xE002801C, IO2CLR - 0xE002802C, IO3CLR - 0xE002803C)

IOCLR	Description	Value after Reset
31:0	Output value CLEAR bits. Bit 0 in IO0CLR corresponds to P0.0 ... Bit 31 in IO0CLR corresponds to P0.31	0

GPIO Direction Register (IO0DIR - 0xE0028008, IO1DIR - 0xE0028018, IO2DIR - 0xE0028028, IO3DIR - 0xE0028038)

This register is used to control the direction of the pins when they are configured as GPIO port pins. Direction bit for any pin must be set according to the pin functionality.

Table 72: GPIO Direction Register (IO0DIR - 0xE0028008, IO1DIR - 0xE0028018, IO2DIR - 0xE0028028, IO3DIR - 0xE0028038)

IODIR	Description	Value after Reset
31:0	Direction control bits (0 = INPUT, 1 = OUTPUT). Bit 0 in IO0DIR controls P0.0 ... Bit 31 in IO0DIR controls P0.31	0

GPIO USAGE NOTES

If for the specified output pin corresponding bit is set both in GPIO Output Set Register (IOnSET) and in GPIO Output Clear Register (IOnCLR), observed pin will output level determined by the later write access of IOnSET nad IOnCLR. This means that in case of sequence:

IO0SET = 0x0000 0080

IO0CLR = 0x0000 0080

pin P0.7 will have low output, since access to Clear register came after access to Set register.

Applications that require instanatneous appearance of zeros and ones on the respected parallel port can use direct access to port's corresponding GPIO Pin Value Register (IOPIN).

Assuming that pins P0.8 to P0.15 are configured as output, write to IO0PIN:

IO0PIN = 0x0000 C700

will produce the same output as following sequence of writes:

IO0SET = 0x0000 C700

IO0CLR = 0x0000 3800

Solution utilizing access to IO0SET and IO0CLR will take more steps compared to a single IO0PIN write access.

10. UART0

FEATURES

- 16 byte Receive and Transmit FIFOs.
- Register locations conform to ‘550 industry standard.
- Receiver FIFO trigger points at 1, 4, 8, and 14 bytes.
- Built-in baud rate generator.

PIN DESCRIPTION

Table 73: UART0 Pin Description

Pin Name	Type	Description
RxD0	Input	**Serial Input**. Serial receive data.
TxD0	Output	**Serial Output**. Serial transmit data.

REGISTER DESCRIPTION

Table 74: UART0 Register Map

Name	Description	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	Access	Reset Value*	Address
U0RBR	Receiver Buffer Register	MSB			READ DATA				LSB	RO	un-defined	0xE000C000 DLAB = 0
U0THR	Transmit Holding Register	MSB			WRITE DATA				LSB	WO	NA	0xE000C000 DLAB = 0
U0IER	Interrupt Enable Register	0	0	0	0	0	Enable Rx Line Status Interrupt	Enable THRE Interrupt	Enable Rx Data Available Interrupt	R/W	0	0xE000C004 DLAB = 0
U0IIR	Interrupt ID Register	FIFOs Enabled		0	0	IIR3	IIR2	IIR1	IIR0	RO	0x01	0xE000C008
U0FCR	FIFO Control Register	Rx Trigger		Reserved		-	Tx FIFO Reset	Rx FIFO Reset	FIFO Enable	WO	0	0xE000C008
U0LCR	Line Control Register	DLAB	Set Break	Stick Parity	Even Parity Select	Parity Enable	Number of Stop Bits	Word Length Select		R/W	0	0xE000C00C
U0LSR	Line Status Register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60	0xE000C014
U0SCR	Scratch Pad Register	MSB							LSB	R/W	0	0xE000C01C
U0DLL	Divisor Latch LSB	MSB							LSB	R/W	0x01	0xE000C000 DLAB = 1
U0DLM	Divisor Latch MSB	MSB							LSB	R/W	0	0xE000C004 DLAB = 1

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

UART0 contains ten 8-bit registers as shown in Table 74. The Divisor Latch Access Bit (DLAB) is contained in U0LCR7 and enables access to the Divisor Latches.

UART0 Receiver Buffer Register (U0RBR - 0xE000C000 when DLAB = 0, Read Only)

The U0RBR is the top byte of the UART0 Rx FIFO. The top byte of the Rx FIFO contains the oldest character received and can be read via the bus interface. The LSB (bit 0) represents the “oldest” received data bit. If the character received is less than 8 bits, the unused MSBs are padded with zeroes.

The Divisor Latch Access Bit (DLAB) in U0LCR must be zero in order to access the U0RBR. The U0RBR is always Read Only.

Table 75: UART0 Receiver Buffer Register (U0RBR - 0xE000C000 when DLAB = 0, Read Only)

U0RBR	Function	Description	Reset Value
7:0	Receiver Buffer Register	The UART0 Receiver Buffer Register contains the oldest received byte in the UART0 Rx FIFO.	un-defined

UART0 Transmitter Holding Register (U0THR - 0xE000C000 when DLAB = 0, Write Only)

The U0THR is the top byte of the UART0 Tx FIFO. The top byte is the newest character in the Tx FIFO and can be written via the bus interface. The LSB represents the first bit to transmit.

The Divisor Latch Access Bit (DLAB) in U0LCR must be zero in order to access the U0THR. The U0THR is always Write Only.

Table 76: UART0 Transmit Holding Register (U0THR - 0xE000C000 when DLAB = 0, Write Only)

U0THR	Function	Description	Reset Value
7:0	Transmit Holding Register	Writing to the UART0 Transmit Holding Register causes the data to be stored in the UART0 transmit FIFO. The byte will be sent when it reaches the bottom of the FIFO and the transmitter is available.	N/A

UART0 Divisor Latch LSB Register (U0DLL - 0xE000C000 when DLAB = 1)

UART0 Divisor Latch MSB Register (U0DLM - 0xE000C004 when DLAB = 1)

The UART0 Divisor Latch is part of the UART0 Baud Rate Generator and holds the value used to divide the VPB clock (pclk) in order to produce the baud rate clock, which must be 16x the desired baud rate. The U0DLL and U0DLM registers together form a 16 bit divisor where U0DLL contains the lower 8 bits of the divisor and U0DLM contains the higher 8 bits of the divisor. A ‘h0000 value is treated like a ‘h0001 value as division by zero is not allowed.The Divisor Latch Access Bit (DLAB) in U0LCR must be one in order to access the UART0 Divisor Latches.

Table 77: UART0 Divisor Latch LSB Register (U0DLL - 0xE000C000 when DLAB = 1)

U0DLL	Function	Description	Reset Value
7:0	Divisor Latch LSB Register	The UART0 Divisor Latch LSB Register, along with the U0DLM register, determines the baud rate of the UART0.	0x01

Table 78: UART0 Divisor Latch MSB Register (U0DLM - 0xE000C004 when DLAB = 1)

U0DLM	Function	Description	Reset Value
7:0	Divisor Latch MSB Register	The UART0 Divisor Latch MSB Register, along with the U0DLL register, determines the baud rate of the UART0.	0

UART0 Interrupt Enable Register (U0IER - 0xE000C004 when DLAB = 0)

The U0IER is used to enable the four UART0 interrupt sources.

Table 79: UART0 Interrupt Enable Register Bit Descriptions (U0IER - 0xE000C004 when DLAB = 0)

U0IER	Function	Description	Reset Value
0	RBR Interrupt Enable	0: Disable the RDA interrupt. 1: Enable the RDA interrupt. U0IER0 enables the Receive Data Available interrupt for UART0. It also controls the Character Receive Time-out interrupt.	0
1	THRE Interrupt Enable	0: Disable the THRE interrupt. 1: Enable the THRE interrupt. U0IER1 enables the THRE interrupt for UART0. The status of this interrupt can be read from U0LSR5.	0
2	Rx Line Status Interrupt Enable	0: Disable the Rx line status interrupts. 1: Enable the Rx line status interrupts. U0IER2 enables the UART0 Rx line status interrupts. The status of this interrupt can be read from U0LSR[4:1].	0
7:3	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

UART0 Interrupt Identification Register (U0IIR - 0xE000C008, Read Only)

The U0IIR provides a status code that denotes the priority and source of a pending interrupt. The interrupts are frozen during an U0IIR access. If an interrupt occurs during an U0IIR access, the interrupt is recorded for the next U0IIR access.

Table 80: UART0 Interrupt Identification Register Bit Descriptions (U0IIR - 0xE000C008, Read Only)

U0IIR	Function	Description	Reset Value
0	Interrupt Pending	0: At least one interrupt is pending. 1: No pending interrupts. Note that U0IIR0 is active low. The pending interrupt can be determined by evaluating U0IER3:1.	1
3:1	Interrupt Identification	011: 1. Receive Line Status (RLS) 010: 2a.Receive Data Available (RDA) 110: 2b.Character Time-out Indicator (CTI) 001: 3. THRE Interrupt. U0IER3 identifies an interrupt corresponding to the UART0 Rx FIFO. All other combinations of U0IER3:1 not listed above are reserved (000,100,101,111).	0
5:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	FIFO Enable	These bits are equivalent to U0FCR0.	0

Interrupts are handled as described in Table 81. Given the status of U0IIR[3:0], an interrupt handler routine can determine the cause of the interrupt and how to clear the active interrupt. Interrupts are handled as described in Table 81. The U0IIR must be read in order to clear the interrupt prior to exitting the Interrupt Service Routine.

The UART0 RLS interrupt (U0IIR3:1=011) is the highest priority interrupt and is set whenever any one of four error conditions occur on the UART0 Rx input: overrun error (OE), parity error (PE), framing error (FE) and break interrupt (BI). The UART0 Rx error condition that set the interrupt can be observed via U0LSR4:1. The interrupt is cleared upon an U0LSR read.

The UART0 RDA interrupt (U0IIR3:1=010) shares the second level priority with the CTI interrupt (U0IIR3:1=110). The RDA is activated when the UART0 Rx FIFO reaches the trigger level defined in U0FCR7:6 and is reset when the UART0 Rx FIFO depth falls below the trigger level. When the RDA interrupt goes active, the CPU can read a block of data defined by the trigger level.

The CTI interrupt (U0IIR3:1=110) is a second level interrupt and is set when the UART0 Rx FIFO contains at least one character and no UART0 Rx FIFO activity has occurred in 3.5 to 4.5 character times. Any UART0 Rx FIFO activity (read or write of UART0 RSR) will clear the interrupt. This interrupt is intended to flush the UART0 RBR after a message has been received that is not a multiple of the trigger level size. For example, if a peripheral wished to send a 105 character message and the trigger level was 10 characters, the CPU would receive 10 RDA interrupts resulting in the transfer of 100 characters and 1 to 5 CTI interrupts (depending on the service routine) resulting in the transfer of the remaining 5 characters.

Table 81: UART0 Interrupt Handling

U0IIR[3:0]	Priority	Interrupt Type	Interrupt Source	Interrupt Reset
0001	-	none	none	-
0110	Highest	Rx Line Status / Error	OE or PE or FE or BI	U0LSR Read
0100	Second	Rx Data Available	Rx data available or trigger level reached in FIFO (U0FCR0=1)	U0RBR Read or UART0 FIFO drops below trigger level
1100	Second	Character Time-out Indication	Minimum of one character in the Rx FIFO and no character input or removed during a time period depending on how many characters are in FIFO and what the trigger level is set at (3.5 to 4.5 character times). The exact time will be: [(word length) X 7 - 2] X 8 + {(trigger level - number of characters) X 8 + 1] RCLKs	U0 RBR Read
0010	Third	THRE	THRE	U0IIR Read (if source of interrupt) or THR write
note: values "0000", “0011”, “0101”, “0111”, “1000”, “1001”, “1010”, “1011”,”1101”,”1110”,”1111” are reserved.				

The UART0 THRE interrupt (U0IIR3:1=001) is a third level interrupt and is activated when the UART0 THR FIFO is empty provided certain initialization conditions have been met. These initialization conditions are intended to give the UART0 THR FIFO a chance to fill up with data to eliminate many THRE interrupts from occurring at system start-up. The initialization conditions implement a one character delay minus the stop bit whenever THRE=1 and there have not been at least two characters in the U0THR at one time since the last THRE=1 event. This delay is provided to give the CPU time to write data to U0THR without a THRE interrupt to decode and service. A THRE interrupt is set immediately if the UART0 THR FIFO has held two or more characters at one time and currently, the U0THR is empty. The THRE interrupt is reset when a U0THR write occurs or a read of the U0IIR occurs and the THRE is the highest interrupt (U0IIR3:1=001).

UART0 FIFO Control Register (U0FCR - 0xE000C008)

The U0FCR controls the operation of the UART0 Rx and Tx FIFOs.

Table 82: UART0 FIFO Control Register Bit Descriptions (U0FCR - 0xE000C008)

U0FCR	Function	Description	Reset Value
0	FIFO Enable	Active high enable for both UART0 Rx and Tx FIFOs and U0FCR7:1 access. This bit must be set for proper UART0 opearation. Any transition on this bit will automatically clear the UART0 FIFOs.	0
1	Rx FIFO Reset	Writing a logic 1 to U0FCR1 will clear all bytes in UART0 Rx FIFO and reset the pointer logic. This bit is self-clearing.	0
2	Tx FIFO Reset	Writing a logic 1 to U0FCR2 will clear all bytes in UART0 Tx FIFO and reset the pointer logic. This bit is self-clearing.	0
5:3	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	Rx Trigger Level Select	00: trigger level 0 (default=1 character or 0x01h) 01: trigger level 1 (default=4 characters or 0x04h) 10: trigger level 2 (default=8 characters or 0x08h) 11: trigger level 3 (default=14 characters or 0x0eh) These two bits determine how many receiver UART0 FIFO characters must be written before an interrupt is activated. The four trigger levels are defined by the user at compilation allowing the user to tune the trigger levels to the FIFO depths chosen.	0

UART0 Line Control Register (U0LCR - 0xE000C00C)

The U0LCR determines the format of the data character that is to be transmitted or received.

Table 83: UART0 Line Control Register Bit Descriptions (U0LCR - 0xE000C00C)

U0LCR	Function	Description	Reset Value
1:0	Word Length Select	00: 5 bit character length 01: 6 bit character length 10: 7 bit character length 11: 8 bit character length	0
2	Stop Bit Select	0: 1 stop bit 1: 2 stop bits (1.5 if U0LCR[1:0]=00)	0
3	Parity Enable	0: Disable parity generation and checking 1: Enable parity generation and checking	0
5:4	Parity Select	00: Odd parity 01: Even parity 10: Forced “1” stick parity 11: Forced “0” stick parity	0
6	Break Control	0: Disable break transmission 1: Enable break transmission. Output pin UART0 TxD is forced to logic 0 when U0LCR6 is active high.	0
7	Divisor Latch Access Bit	0: Disable access to Divisor Latches 1: Enable access to Divisor Latches	0

UART0 Line Status Register (U0LSR - 0xE000C014, Read Only)

The U0LSR is a read-only register that provides status information on the UART0 Tx and Rx blocks.

Table 84: UART0 Line Status Register Bit Descriptions (U0LSR - 0xE000C014, Read Only)

U0LSR	Function	Description	Reset Value
0	Receiver Data Ready (RDR)	0: U0RBR is empty 1: U0RBR contains valid data U0LSR0 is set when the U0RBR holds an unread character and is cleared when the UART0 RBR FIFO is empty.	0
1	Overrun Error (OE)	0: Overrun error status is inactive. 1: Overrun error status is active. The overrun error condition is set as soon as it occurs. An U0LSR read clears U0LSR1. U0LSR1 is set when UART0 RSR has a new character assembled and the UART0 RBR FIFO is full. In this case, the UART0 RBR FIFO will not be overwritten and the character in the UART0 RSR will be lost.	0
2	Parity Error (PE)	0: Parity error status is inactive. 1: Parity error status is active. When the parity bit of a received character is in the wrong state, a parity error occurs. An U0LSR read clears U0LSR2. Time of parity error detection is dependent on U0FCR0. A parity error is associated with the character being read from the UART0 RBR FIFO.	0
3	Framing Error (FE)	0: Framing error status is inactive. 1: Framing error status is active. When the stop bit of a received character is a logic 0, a framing error occurs. An U0LSR read clears U0LSR3. The time of the framing error detection is dependent on U0FCR0. A framing error is associated with the character being read from the UART0 RBR FIFO. Upon detection of a framing error, the Rx will attempt to resynchronize to the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no Framing Error.	0
4	Break Interrupt (BI)	0: Break interrupt status is inactive. 1: Break interrupt status is active. When RxD0 is held in the spacing state (all 0's) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until RxD0 goes to marking state (all 1's). An U0LSR read clears this status bit. The time of break detection is dependent on U0FCR0. The break interrupt is associated with the character being read from the UART0 RBR FIFO.	0
5	Transmitter Holding Register Empty (THRE)	0: U0THR contains valid data. 1: U0THR is empty. THRE is set immediately upon detection of an empty UART0 THR and is cleared on a U0THR write.	1
6	Transmitter Empty (TEMT)	0: U0THR and/or the U0TSR contains valid data. 1: U0THR and the U0TSR are empty. TEMT is set when both U0THR and U0TSR are empty; TEMT is cleared when either the U0TSR or the U0THR contain valid data.	1
7	Error in Rx FIFO (RXFE)	0: U0RBR contains no UART0 Rx errors or U0FCR0=0. 1: UART0 RBR contains at least one UART0 Rx error. U0LSR7 is set when a character with a Rx error such as framing error, parity error or break interrupt, is loaded into the U0RBR. This bit is cleared when the U0LSR register is read and there are no subsequent errors in the UART0 FIFO.	0

UART0 Scratch Pad Register (U0SCR - 0xE000C01C)

The U0SCR has no effect on the UART0 operation. This register can be written and/or read at user's discretion. There is no provision in the interrupt interface that would indicate to the host that a read or write of the U0SCR has occurred.

Table 85: UART0 Scratchpad Register (U0SCR - 0xE000C01C)

U0SCR	Function	Description	Reset Value
7:0	-	A readable, writable byte.	0

ARCHITECTURE

The architecture of the UART0 is shown below in the block diagram.

The VPB interface provides a communications link between the CPU or host and the UART0.

The UART0 receiver block, U0Rx, monitors the serial input line, RxD0, for valid input. The UART0 Rx Shift Register (U0RSR) accepts valid characters via RxD0. After a valid character is assembled in the U0RSR, it is passed to the UART0 Rx Buffer Register FIFO to await access by the CPU or host via the generic host interface.

The UART0 transmitter block, U0Tx, accepts data written by the CPU or host and buffers the data in the UART0 Tx Holding Register FIFO (U0THR). The UART0 Tx Shift Register (U0TSR) reads the data stored in the U0THR and assembles the data to transmit via the serial output pin, TxD0.

The UART0 Baud Rate Generator block, U0BRG, generates the timing enables used by the UART0 Tx block. The U0BRG clock input source is the VPB clock (pclk). The main clock is divided down per the divisor specified in the U0DLL and U0DLM registers. This divided down clock is a 16x oversample clock, NBAUDOUT.

The interrupt interface contains registers U0IER and U0IIR. The interrupt interface receives several one clock wide enables from the U0Tx and U0Rx blocks.

Status information from the U0Tx and U0Rx is stored in the U0LSR. Control information for the U0Tx and U0Rx is stored in the U0LCR.

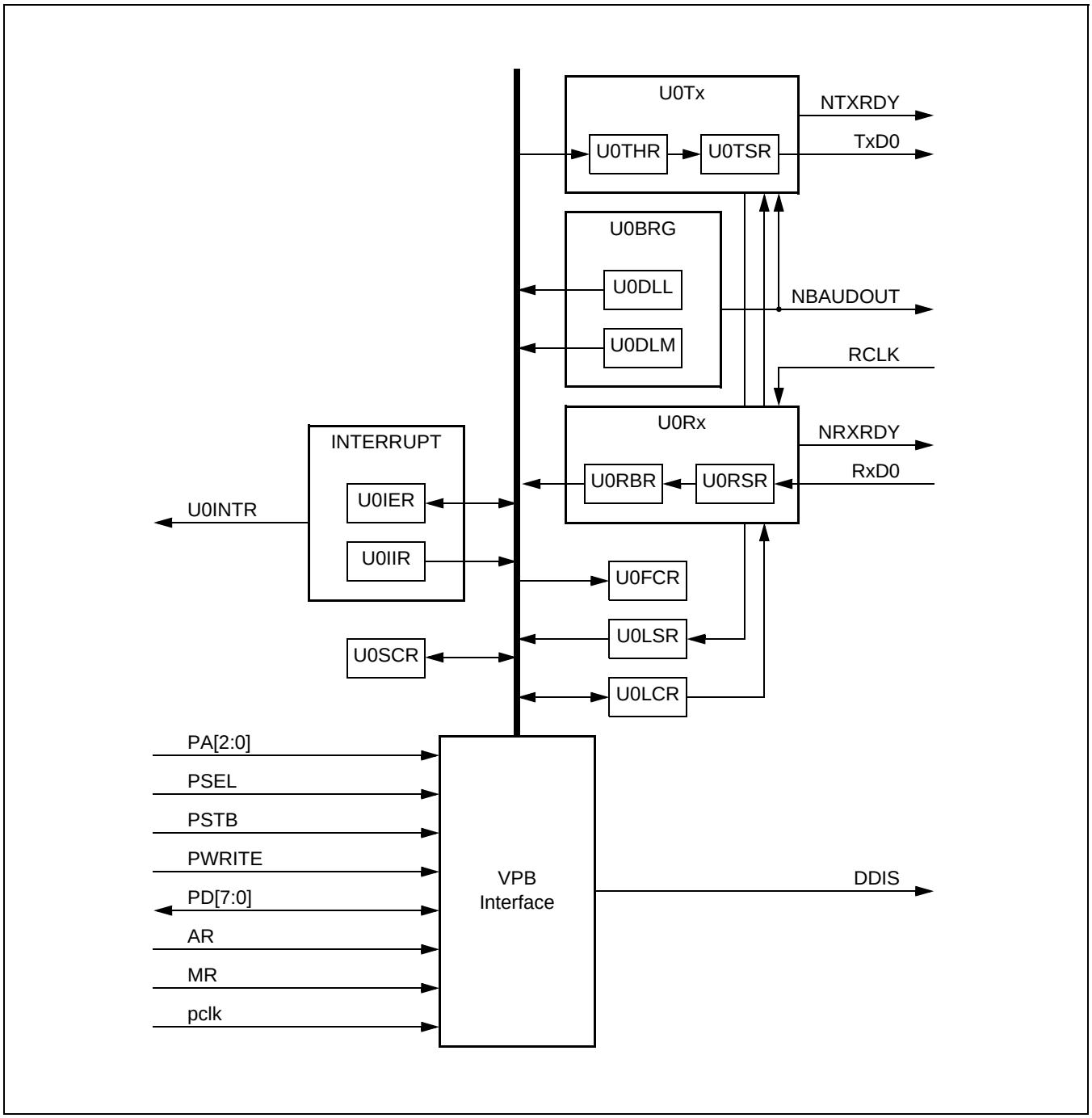


Figure 22: UART0 Block Diagram

11. UART1

FEATURES

- UART1 is identical to UART0, with the addition of a modem interface.
- 16 byte Receive and Transmit FIFOs.
- Register locations conform to ‘550 industry standard.
- Receiver FIFO trigger points at 1, 4, 8, and 14 bytes.
- Built-in baud rate generator.
- Standard modem interface signals included.

PIN DESCRIPTION

Table 86: UART1 Pin Description

Pin Name	Type	Description
RxD1	Input	**Serial Input**. Serial receive data.
TxD1	Output	**Serial Output**. Serial transmit data.
CTS1	Input	**Clear To Send**. Active low signal indicates if the external modem is ready to accept transmitted data via TxD1 from the UART1. In normal operation of the modem interface (U1MCR4=0), the complement value of this signal is stored in U1MSR4. State change information is stored in U1MSR0 and is a source for a priority level 4 interrupt, if enabled (U1IER3=1).
DCD1	Input	**Data Carrier Detect**. Active low signal indicates if the external modem has established a communication link with the UART1 and data may be exchanged. In normal operation of the modem interface (U1MCR4=0), the complement value of this signal is stored in U1MSR7. State change information is stored in U1MSR3 and is a source for a priority level 4 interrupt, if enabled (U1IER3=1).
DSR1	Input	**Data Set Ready**. Active low signal indicates if the external modem is ready to establish a communications link with the UART1. In normal operation of the modem interface (U1MCR4=0), the complement value of this signal is stored in U1MSR5. State change information is stored in U1MSR1 and is a source for a priority level 4 interrupt, if enabled (U1IER3=1).
DTR1	Output	**Data Terminal Ready**. Active low signal indicates that the UART1 is ready to establish connection with external modem. The complement value of this signal is stored in U1MCR0.
RI1	Input	**Ring Indicator**. Active low signal indicates that a telephone ringing signal has been detected by the modem. In normal operation of the modem interface (U1MCR4=0), the complement value of this signal is stored in U1MSR6. State change information is stored in U1MSR2 and is a source for a priority level 4 interrupt, if enabled (U1IER3=1).
RTS1	Output	**Request To Send**. Active low signal indicates that the UART1 would like to transmit data to the external modem. The complement value of this signal is stored in U1MCR1.

REGISTER DESCRIPTION

Table 87: UART1 Register Map

Name	Description	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	Access	Reset Value*	Address
U1RBR	Receiver Buffer Register	MSB			READ DATA				LSB	RO	un-defined	0xE0010000 DLAB = 0
U1THR	Transmit Holding Register	MSB			WRITE DATA				LSB	WO	NA	0xE0010000 DLAB = 0
U1IER	Interrupt Enable Register	0	0	0	0	Enable Modem Status Interrupt	Enable Rx Line Status Interrupt	Enable THRE Interrupt	Enable Rx Data Available Interrupt	R/W	0	0xE0010004 DLAB = 0
U1IIR	Interrupt ID Register	FIFOs Enabled		0	0	IIR3	IIR2	IIR1	IIR0	RO	0x01	0xE0010008
U1FCR	FIFO Control Register	Rx Trigger		Reserved		-	Tx FIFO Reset	Rx FIFO Reset	FIFO Enable	WO	0	0xE0010008
U1LCR	Line Control Register	DLAB	Set Break	Stick Parity	Even Parity Select	Parity Enable	Number of Stop Bits	Word Length Select		R/W	0	0xE001000C
U1MCR	Modem Control Register	0	0	0	Loop Back	0	0	RTS	DTR	R/W	0	0xE0010010
U1LSR	Line Status Register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60	0xE0010014
U1MSR	Modem Status Register	DCD	RI	DSR	CTS	Delta DCD	Trailing Edge RI	Delta DSR	Delta CTS	RO	0	0xE0010018
U1SCR	Scratch Pad Register	MSB							LSB	R/W	0	0xE001001C
U1DLL	Divisor Latch LSB	MSB							LSB	R/W	0	0xE0010000 DLAB = 1
U1DLM	Divisor Latch MSB	MSB							LSB	R/W	0	0xE0010004 DLAB = 1

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

UART1 contains twelve 8-bit registers as shown in Table 87. The Divisor Latch Access Bit (DLAB) is contained in U1LCR7 and enables access to the Divisor Latches.

UART1 Receiver Buffer Register (U1RBR - 0xE0010000 when DLAB = 0, Read Only)

The U1RBR is the top byte of the UART1 Rx FIFO. The top byte of the Rx FIFO contains the oldest character received and can be read via the bus interface. The LSB (bit 0) represents the “oldest” received data bit. If the character received is less than 8 bits, the unused MSBs are padded with zeroes.

The Divisor Latch Access Bit (DLAB) in U1LCR must be zero in order to access the U1RBR. The U1RBR is always Read Only.

Table 88: UART1 Receiver Buffer Register (U1RBR - 0xE0010000 when DLAB = 0, Read Only)

U1RBR	Function	Description	Reset Value
7:0	Receiver Buffer Register	The UART1 Receiver Buffer Register contains the oldest received byte in the UART1 Rx FIFO.	un-defined

UART1 Transmitter Holding Register (U1THR - 0xE0010000 when DLAB = 0, Write Only)

The U1THR is the top byte of the UART1 Tx FIFO. The top byte is the newest character in the Tx FIFO and can be written via the bus interface. The LSB represents the first bit to transmit.

The Divisor Latch Access Bit (DLAB) in U1LCR must be zero in order to access the U1THR. The U1THR is always Write Only.

Table 89: UART1 Transmit Holding Register (U1THR - 0xE0010000 when DLAB = 0, Write Only)

U1THR	Function	Description	Reset Value
7:0	Transmit Holding Register	Writing to the UART1 Transmit Holding Register causes the data to be stored in the UART1 transmit FIFO. The byte will be sent when it reaches the bottom of the FIFO and the transmitter is available.	N/A

UART1 Divisor Latch LSB Register (U1DLL - 0xE0010000 when DLAB = 1)

UART1 Divisor Latch MSB Register (U1DLM - 0xE0010004 when DLAB = 1)

The UART1 Divisor Latch is part of the UART1 Baud Rate Generator and holds the value used to divide the VPB clock (pclk) in order to produce the baud rate clock, which must be 16x the desired baud rate. The U1DLL and U1DLM registers together form a 16 bit divisor where U1DLL contains the lower 8 bits of the divisor and U1DLM contains the higher 8 bits of the divisor. A ‘h0000 value is treated like a ‘h0001 value as division by zero is not allowed.The Divisor Latch Access Bit (DLAB) in U1LCR must be one in order to access the UART1 Divisor Latches.

Table 90: UART1 Divisor Latch LSB Register (U1DLL - 0xE0010000 when DLAB = 1)

U1DLL	Function	Description	Reset Value
7:0	Divisor Latch LSB Register	The UART1 Divisor Latch LSB Register, along with the U1DLM register, determines the baud rate of the UART1.	0x01

Table 91: UART1 Divisor Latch MSB Register (U1DLM - 0xE0010004 when DLAB = 1)

U1DLM	Function	Description	Reset Value
7:0	Divisor Latch MSB Register	The UART1 Divisor Latch MSB Register, along with the U1DLL register, determines the baud rate of the UART1.	0

UART1 Interrupt Enable Register (U1IER - 0xE0010004 when DLAB = 0)

The U1IER is used to enable the four interrupt sources.

Table 92: UART1 Interrupt Enable Register Bit Descriptions (U1IER - 0xE0010004 when DLAB = 0)

U1IER	Function	Description	Reset Value
0	RBR Interrupt Enable	0: Disable the RDA interrupt. 1: Enable the RDA interrupt. U1IER0 enables the Receive Data Available interrupt for UART1. It also controls the Receive Time-out interrupt.	0
1	THRE Interrupt Enable	0: Disable the THRE interrupt. 1: Enable the THRE interrupt. U1IER1 enables the THRE interrupt for UART1. The status of this interrupt can be read from U1LSR5.	0
2	Rx Line Status Interrupt Enable	0: Disable the Rx line status interrupts. 1: Enable the Rx line status interrupts. U1IER2 enables the UART1 Rx line status interrupts. The status of this interrupt can be read from U1LSR[4:1].	0
3	Modem Status Interrupt Enable	0: Disable the modem interrupt. 1: Enable the modem interrupt. U1IER3 enables the modem interrupt. The status of this interrupt can be read from U1MSR[3:0].	0
7:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

UART1 Interrupt Identification Register (U1IIR - 0xE0010008, Read Only)

The U1IIR provides a status code that denotes the priority and source of a pending interrupt. The interrupts are frozen during an U1IIR access. If an interrupt occurs during an U1IIR access, the interrupt is recorded for the next U1IIR access.

Table 93: UART1 Interrupt Identification Register Bit Descriptions (IIR - 0xE0010008, Read Only)

U1IIR	Function	Description	Reset Value
0	Interrupt Pending	0: At least one interrupt is pending. 1: No pending interrupts. Note that U1IIR0 is active low. The pending interrupt can be determined by evaluating U1IIR3:1.	1
3:1	Interrupt Identification	011: 1. Receive Line Status (RLS) 010: 2a.Receive Data Available (RDA) 110: 2b.Character Time-out Indicator (CTI) 001: 3. THRE Interrupt. 000: 4. Modem Interrupt. U1IER3 identifies an interrupt corresponding to the UART1 Rx FIFO and modem signals. All other combinations of U1IER3:1 not listed above are reserved (100,101,111).	0
5:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	FIFO Enable	These bits are equivalent to U1FCR0.	0

Interrupts are handled as described in Table 94. Given the status of U1IIR[3:0], an interrupt handler routine can determine the cause of the interrupt and how to clear the active interrupt. The U1IIR must be read in order to clear the interrupt prior to exiting the Interrupt Service Routine.

The UART1 RLS interrupt (U1IIR3:1=011) is the highest priority interrupt and is set whenever any one of four error conditions occur on the UART1Rx input: overrun error (OE), parity error (PE), framing error (FE) and break interrupt (BI). The UART1 Rx error condition that set the interrupt can be observed via U1LSR4:1. The interrupt is cleared upon an U1LSR read.

The UART1 RDA interrupt (U1IIR3:1=010) shares the second level priority with the CTI interrupt (U1IIR3:1=110). The RDA is activated when the UART1 Rx FIFO reaches the trigger level defined in U1FCR7:6 and is reset when the UART1 Rx FIFO depth falls below the trigger level. When the RDA interrupt goes active, the CPU can read a block of data defined by the trigger level.

The CTI interrupt (U1IIR3:1=110) is a second level interrupt and is set when the UART1 Rx FIFO contains at least one character and no UART1 Rx FIFO activity has occurred in 3.5 to 4.5 character times. Any UART1 Rx FIFO activity (read or write of UART1 RSR) will clear the interrupt. This interrupt is intended to flush the UART1 RBR after a message has been received that is not a multiple of the trigger level size. For example, if a peripheral wished to send a 105 character message and the trigger level was 10 characters, the CPU would receive 10 RDA interrupts resulting in the transfer of 100 characters and 1 to 5 CTI interrupts (depending on the service routine) resulting in the transfer of the remaining 5 characters.

Table 94: UART1 Interrupt Handling

U1IIR[3:0]	Priority	Interrupt Type	Interrupt Source	Interrupt Reset
0001	-	none	none	-
0110	Highest	Rx Line Status / Error	OE or PE or FE or BI	U1LSR Read
0100	Second	Rx Data Available	Rx data available or trigger level reached in FIFO mode (FCR0=1)	U1RBR Read or UART1 FIFO drops below trigger level
1100	Second	Character Time-out Indication	Minimum of one character in the Rx FIFO and no character input or removed during a time period depending on how many characters are in FIFO and what the trigger level is set at (3.5 to 4.5 character times). The exact time will be: [(word length) X 7 - 2] X 8 + {(trigger level - number of characters) X 8 + 1] RCLKs	U1RBR Read
0010	Third	THRE	THRE	U1IIR Read (if source of interrupt) or THR write
0000	Fourth	Modem Status	CTS or DSR or RI or DCD	MSR Read
note: values “0011”. “0101”, “0111”, “1000”, “1001”, “1010”, “1011”,”1101”,”1110”,”1111” are reserved.				

The UART1 THRE interrupt (U1IIR3:1=001) is a third level interrupt and is activated when the UART1 THR FIFO is empty provided certain initialization conditions have been met. These initialization conditions are intended to give the UART1 THR FIFO a chance to fill up with data to eliminate many THRE interrupts from occurring at system start-up. The initialization conditions implement a one character delay minus the stop bit whenever THRE=1 and there have not been at least two characters in the U1THR at one time since the last THRE=1 event. This delay is provided to give the CPU time to write data to U1THR without a THRE interrupt to decode and service. A THRE interrupt is set immediately if the UART1 THR FIFO has held two or more characters at one time and currently, the U1THR is empty. The THRE interrupt is reset when a U1THR write occurs or a read of the U1IIR occurs and the THRE is the highest interrupt (U1IIR3:1=001).

The modem interrupt (U1IIR3:1=000) is the lowest priority interrupt and is activated whenever there is any state change on modem inputs pins, DCD, DSR or CTS. In addition, a low to high transition on modem input RI will generate a modem interrupt. The source of the modem interrupt can be determined by examining U1MSR3:0. A U1MSR read will clear the modem interrupt.

UART1 FIFO Control Register (U1FCR - 0xE0010008)

The U1FCR controls the operation of the UART1 Rx and Tx FIFOs.

Table 95: UART1 FCR Bit Descriptions (U1FCR - 0xE0010008)

U1FCR	Function	Description	Reset Value
0	FIFO Enable	Active high enable for both UART1 Rx and Tx FIFOs and U1FCR7:1 access. This bit must be set for proper UART1 operation. Any transition on this bit will automatically clear the UART1 FIFOs.	0
1	Rx FIFO Reset	Writing a logic 1 to U1FCR1 will clear all bytes in UART1 Rx FIFO and reset the pointer logic. This bit is self-clearing.	0
2	Tx FIFO Reset	Writing a logic 1 to U1FCR2 will clear all bytes in UART1 Tx FIFO and reset the pointer logic. This bit is self-clearing.	0
5:3	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	Rx Trigger Level Select	00: trigger level 0 (default=1 character or 0x01h) 01: trigger level 1 (default=4 characters or 0x04h) 10: trigger level 2 (default=8 characters or 0x08h) 11: trigger level 3 (default=14 characters or 0x0eh) These two bits determine how many receiver UART1 FIFO characters must be written before an interrupt is activated. The four trigger levels are defined by the user at compilation allowing the user to tune the trigger levels to the FIFO depths chosen.	0

UART1 Line Control Register (U1LCR - 0xE001000C)

The U1LCR determines the format of the data character that is to be transmitted or received.

Table 96: UART1 Line Control Register Bit Descriptions (U1LCR - 0xE001000C)

U1LCR	Function	Description	Reset Value
1:0	Word Length Select	00: 5 bit character length 01: 6 bit character length 10: 7 bit character length 11: 8 bit character length	0
2	Stop Bit Select	0: 1 stop bit 1: 2 stop bits (1.5 if U1LCR[1:0]=00)	0
3	Parity Enable	0: Disable parity generation and checking 1: Enable parity generation and checking	0
5:4	Parity Select	00: Odd parity 01: Even parity 10: Forced “1” stick parity 11: Forced “0” stick parity	0
6	Break Control	0: Disable break transmission 1: Enable break transmission. Output pin UART1 TxD is forced to logic 0 when U1LCR6 is active high.	0
7	Divisor Latch Access Bit	0: Disable access to Divisor Latches 1: Enable access to Divisor Latches	0

UART1 Modem Control Register (U1MCR - 0xE0010010)

The U1MCR enables the modem loopback mode and controls the modem output signals.

Table 97: UART1 Modem Control Register Bit Descriptions (U1MCR - 0xE0010010)

U1MCR	Function	Description	Reset Value
0	DTR Control	Source for modem output pin, DTR. This bit reads as 0 when modem loopback mode is active.	0
1	RTS Control	Source for modem output pin RTS. This bit reads as 0 when modem loopback mode is active.	0
2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
4	Loopback Mode Select	0: Disable modem loopback mode 1: Enable modem loopback mode The modem loopback mode provides a mechanism to perform diagnostic loopback testing. Serial data from the transmitter is connected internally to serial input of the receiver. Input pin, RxD1, has no effect on loopback and output pin, TxD1 is held in marking state. The four modem inputs (CTS, DSR, RI and DCD) are disconnected externally. Externally, the modem outputs (RTS, DTR) are set inactive. Internally, the four modem outputs are connected to the four modem inputs. As a result of these connections, the upper four bits of the U1MSR will be driven by the lower four bits of the U1MCR rather than the four modem inputs in normal mode. This permits modem status interrupts to be generated in loopback mode by writing the lower four bits of U1MCR.	0
7:5	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

UART1 Line Status Register (U1LSR - 0xE0010014, Read Only)

The U1LSR is a read-only register that provides status information on the UART1 Tx and Rx blocks.

Table 98: UART1 Line Status Register Bit Descriptions (U1LSR - 0xE0010014, Read Only)

U1LSR	Function	Description	Reset Value
0	Receiver Data Ready (RDR)	0: U1RBR is empty 1: U1RBR contains valid data U1LSR0 is set when the U1RBR holds an unread character and is cleared when the UART1 RBR FIFO is empty.	0
1	Overrun Error (OE)	0: Overrun error status is inactive. 1: Overrun error status is active. The overrun error condition is set as soon as it occurs. An U1LSR read clears U1LSR1. U1LSR1 is set when UART1 RSR has a new character assembled and the UART1 RBR FIFO is full. In this case, the UART1 RBR FIFO will not be overwritten and the character in the UART1 RSR will be lost.	0
2	Parity Error (PE)	0: Parity error status is inactive. 1: Parity error status is active. When the parity bit of a received character is in the wrong state, a parity error occurs. An U1LSR read clears U1LSR2. Time of parity error detection is dependent on U1FCR0. A parity error is associated with the character being read from the UART1 RBR FIFO.	0
3	Framing Error (FE)	0: Framing error status is inactive. 1: Framing error status is active. When the stop bit of a received character is a logic 0, a framing error occurs. An U1LSR read clears this bit. The time of the framing error detection is dependent on U1FCR0. A framing error is associated with the character being read from the UART1 RBR FIFO. Upon detection of a framing error, the Rx will attempt to resynchronize to the data and assume that the bad stop bit is actually an early start bit.	0
4	Break Interrupt (BI)	0: Break interrupt status is inactive. 1: Break interrupt status is active. When RxD1 is held in the spacing state (all 0's) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until RxD1 goes to marking state (all 1's). An U1LSR read clears this status bit. The time of break detection is dependent on U1FCR0. The break interrupt is associated with the character being read from the UART1 RBR FIFO.	0
5	Transmitter Holding Register Empty (THRE)	0: U1THR contains valid data. 1: U1THR is empty. THRE is set immediately upon detection of an empty U1THR and is cleared on a U1THR write.	1
6	Transmitter Empty (TEMT)	0: U1THR and/or the U1TSR contains valid data. 1: U1THR and the U1TSR are empty. TEMT is set when both THR and TSR are empty; TEMT is cleared when either the U1TSR or the U1THR contain valid data.	1
7	Error in Rx FIFO (RXFE)	0: U1RBR contains no UART1 Rx errors or U1FCR0=0. 1: U1RBR contains at least one UART1 Rx error. U1LSR7 is set when a character with a Rx error such as framing error, parity error or break interrupt, is loaded into the U1RBR. This bit is cleared when the U1LSR register is read and there are no subsequent errors in the UART1 FIFO.	0

UART1 Modem Status Register (U1MSR - 0x0xE0010018)

The U1MSR is a read-only register that provides status information on the modem input signals. U1MSR3:0 is cleared on U1MSR read. Note that modem signals have no direct affect on UART1 operation, they facilitate software implementation of modem signal operations.

Table 99: UART1 Modem Status Register Bit Descriptions (U1MSR - 0x0xE0010018)

U1MSR	Function	Description	Reset Value
0	Delta CTS	0: No change detected on modem input, CTS. 1: State change detected on modem input, CTS. Set upon state change of input CTS. Cleared on an U1MSR read.	0
1	Delta DSR	0: No change detected on modem input, DSR. 1: State change detected on modem input, DSR. Set upon state change of input DSR. Cleared on an U1MSR read.	0
2	Trailing Edge RI	0: No change detected on modem input, RI. 1: Low-to-high transition detected on RI. Set upon low to high transition of input RI. Cleared on an U1MSR read.	0
3	Delta DCD	0: No change detected on modem input, DCD. 1: State change detected on modem input, DCD. Set upon state change of input DCD. Cleared on an U1MSR read.	0
4	CTS	Clear To Send State. Complement of input signal CTS. This bit is connected to U1MCR[1] in modem loopback mode.	0
5	DSR	Data Set Ready State. Complement of input signal DSR. This bit is connected to U1MCR[0] in modem loopback mode.	0
6	RI	Ring Indicator State. Complement of input RI. This bit is connected to U1MCR[2] in modem loopback mode.	0
7	DCD	Data Carrier Detect State. Complement of input DCD. This bit is connected to U1MCR[3] in modem loopback mode.	0

UART1 Scratch Pad Register (U1SCR - 0xE001001C)

The U1SCR has no effect on the UART1 operation. This register can be written and/or read at user's discretion. There is no provision in the interrupt interface that would indicate to the host that a read or write of the U1SCR has occurred.

Table 100: UART1 Scratchpad Register (U1SCR - 0xE001001C)

U1SCR	Function	Description	Reset Value
7:0	-	A readable, writable byte.	0

ARCHITECTURE

The architecture of the UART1 is shown below in the block diagram.

The VPB interface provides a communications link between the CPU or host and the UART1.

The UART1 receiver block, U1Rx, monitors the serial input line, RxD1, for valid input. The UART1 Rx Shift Register (U1RSR) accepts valid characters via RxD1. After a valid character is assembled in the U1RSR, it is passed to the UART1 Rx Buffer Register FIFO to await access by the CPU or host via the generic host interface.

The UART1 transmitter block, U1Tx, accepts data written by the CPU or host and buffers the data in the UART1 Tx Holding Register FIFO (U1THR). The UART1 Tx Shift Register (U1TSR) reads the data stored in the U1THR and assembles the data to transmit via the serial output pin, TxD1.

The UART1 Baud Rate Generator block, U1BRG, generates the timing enables used by the UART1 Tx block. The U1BRG clock input source is the VPB clock (pclk). The main clock is divided down per the divisor specified in the U1DLL and u1DLM registers. This divided down clock is a 16x oversample clock, NBAUDOUT.

The modem interface contains registers U1MCR and U1MSR. This interface is responsible for handshaking between a modem peripheral and the UART1.

The interrupt interface contains registers U1IER and U1IIR. The interrupt interface receives several one clock wide enables from the U1Tx, U1Rx and modem blocks.

Status information from the U1Tx and U1Rx is stored in the U1LSR. Control information for the U1Tx and U1Rx is stored in the U1LCR.

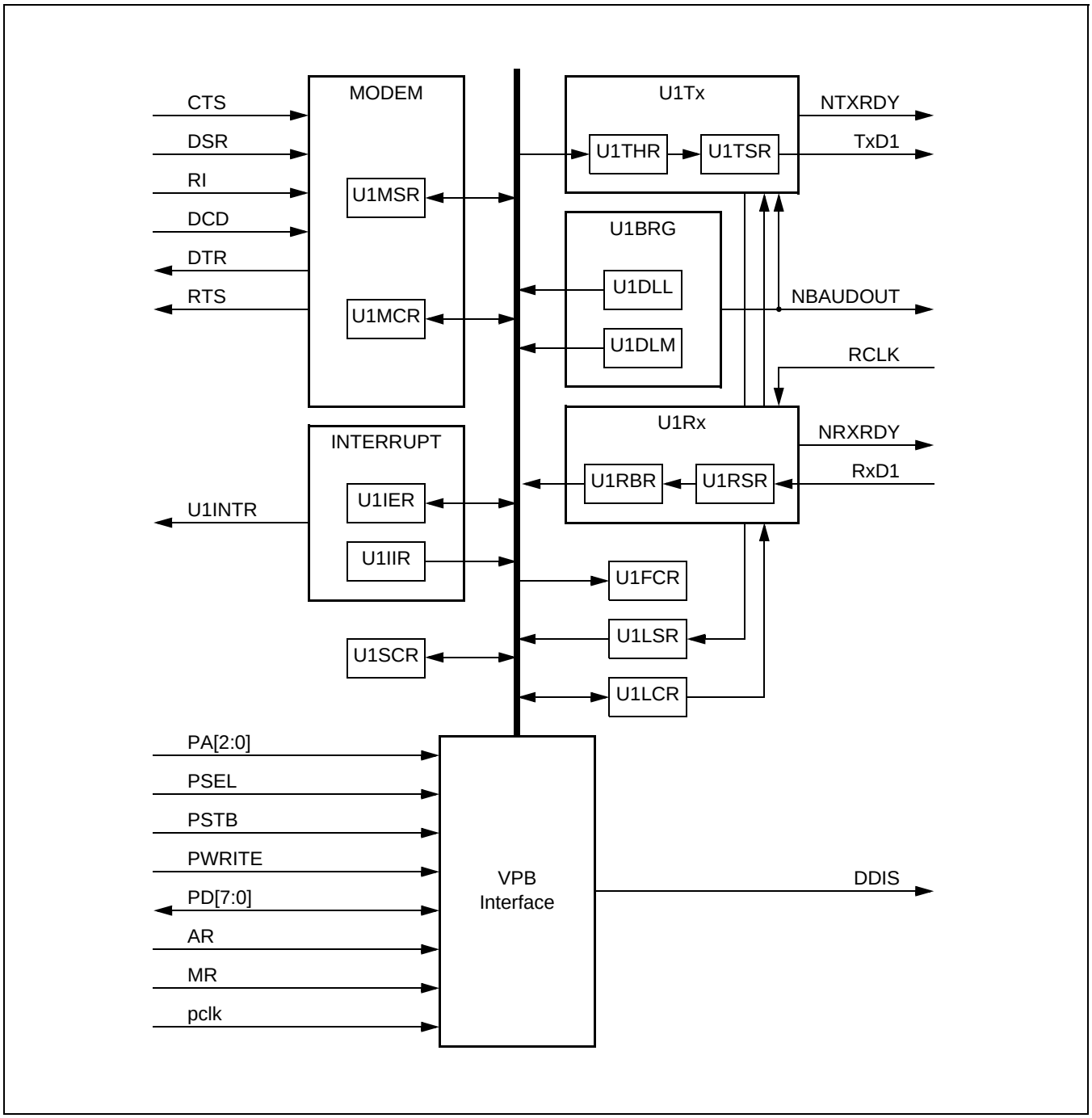


Figure 23: UART1 Block Diagram

12. I^2C INTERFACE

FEATURES

- Standard I^2C compliant bus interface.
- Easy to configure as Master, Slave, or Master/Slave.
- Programmable clocks allow versatile rate control.
- Bidirectional data transfer between masters and slaves.
- Multi-master bus (no central master).
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus.
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus.
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer.
- The I^2C bus may be used for test and diagnostic purposes.

APPLICATIONS

- Interfaces to external I^2C standard parts, such as serial RAMs, LCDs, tone generators, etc.

DESCRIPTION

A typical I^2C bus configuration is shown in Figure 24. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I^2C bus:

- Data transfer from a master transmitter to a slave receiver. The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte.
- Data transfer from a slave transmitter to a master receiver. The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. Next follows the data bytes transmitted by the slave to the master. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a “not acknowledge” is returned. The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the I^2C bus will not be released.

This device provides a byte oriented I^2C interface. It has four operating modes: master transmitter mode, master receiver mode, slave transmitter mode and slave receiver mode.

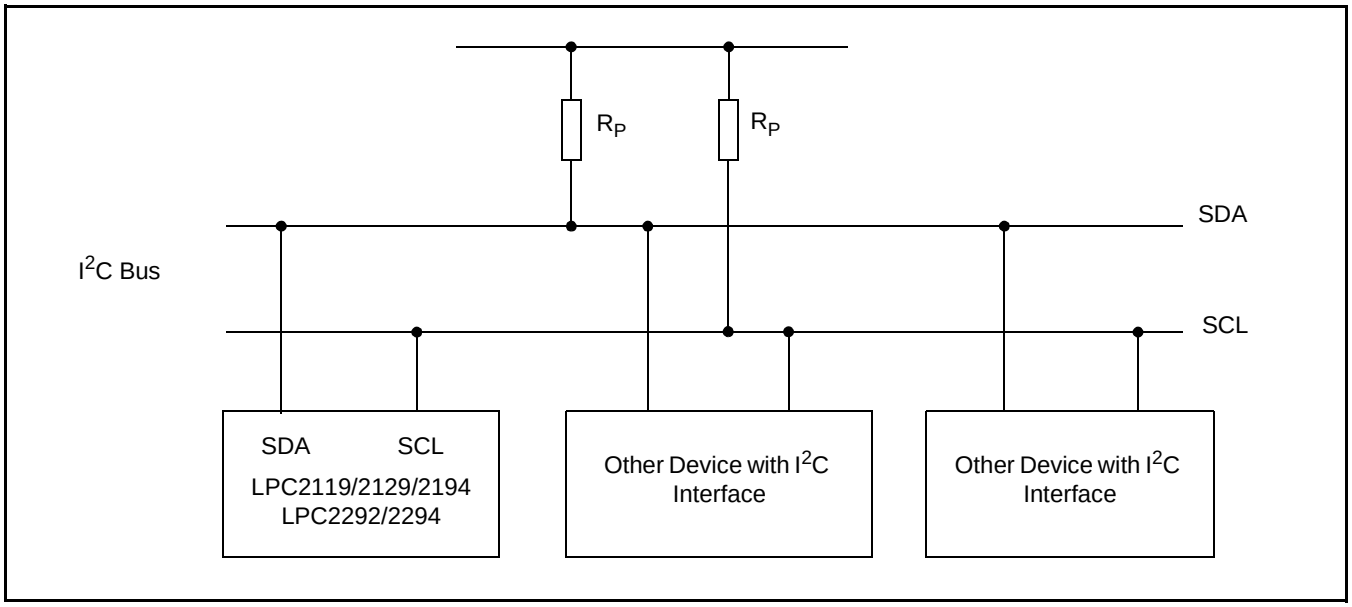


Figure 24: I^2C Bus Configuration

I^2C Operating Modes

Master Transmitter Mode:

In this mode data is transmitted from master to slave. Before the master transmitter mode can be entered, I2CONSET must be initialized as shown in Figure 25. I2EN must be set to 1 to enable the I^2C function. If the AA bit is 0, the I^2C interface will not acknowledge any address when another device is master of the bus, so it can not enter slave mode. The STA, STO and SI bits must be 0. The SI Bit is cleared by writing 1 to the SIC bit in the I2CONCLR register.

	7	6	5	4	3	2	1	0
I2CONSET	-	I2EN	STA	STO	SI	AA	-	-
	-	1	0	0	0	0	-	-

Figure 25: Slave Mode Configuration

The first byte transmitted contains the slave address of the receiving device (7 bits) and the data direction bit. In this mode the data direction bit (R/W) should be 0 which means Write. The first byte transmitted contains the slave address and Write bit. Data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

The I^2C interface will enter master transmitter mode when software sets the STA bit. The I^2C logic will send the START condition as soon as the bus is free. After the START condition is transmitted, the SI bit is set, and the status code in I2STAT should be 08h. This status code must be used to vector to an interrupt service routine which should load the slave address and Write bit to I2DAT (Data Register), and then clear the SI bit. SI is cleared by writing a 1 to the SIC bit in the I2CONCLR register.

When the slave address and R/W bit have been transmitted and an acknowledgment bit has been received, the SI bit is set again, and the possible status codes now are 18h, 20h, or 38h for the master mode, or 68h, 78h, or 0B0h if the slave mode was enabled (by setting AA=1). The appropriate actions to be taken for each of these status codes are shown in Table 3 to Table 6 in "80C51 Family Derivatives 8XC552/562 Overview" datasheet available on-line at

http://www.semiconductors.philips.com/acrobat/various/8XC552_562OVERVIEW_2.pdf.

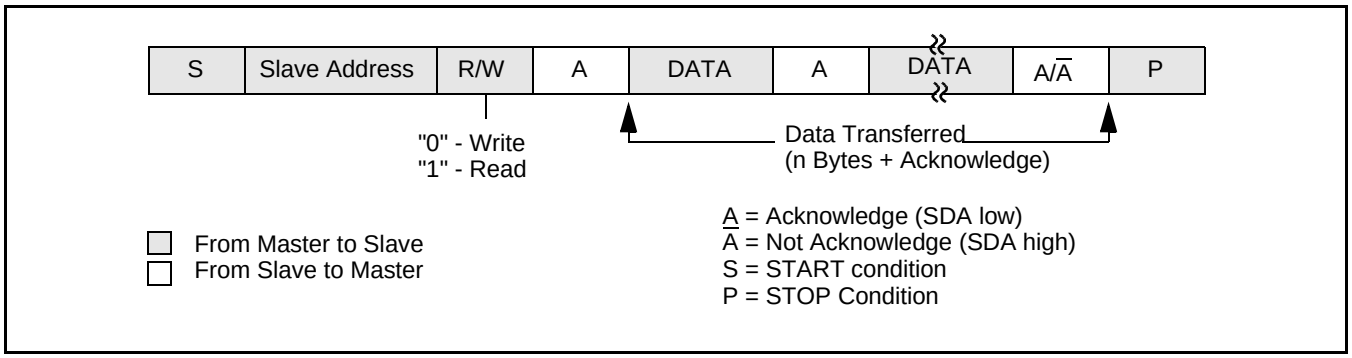


Figure 26: Format in the master transmitter mode

Master Receiver Mode:

In the master receiver mode, data is received from a slave transmitter. The transfer is initiated in the same way as in the master transmitter mode. When the START condition has been transmitted, the interrupt service routine must load the slave address and the data direction bit to I^2C Data Register (I2DAT), and then clear the SI bit.

When the slave address and data direction bit have been transmitted and an acknowledge bit has been received, the SI bit is set, and the Status Register will show the status code. For master mode, the possible status codes are 40H, 48H, or 38H. For slave mode, the possible status codes are 68H, 78H, or B0H. Refer to Table 4 in "80C51 Family Derivatives 8XC552/562 Overview" datasheet available on-line at

http://www.semiconductors.philips.com/acrobat/various/8XC552_562OVERVIEW_2.pdf

for details.

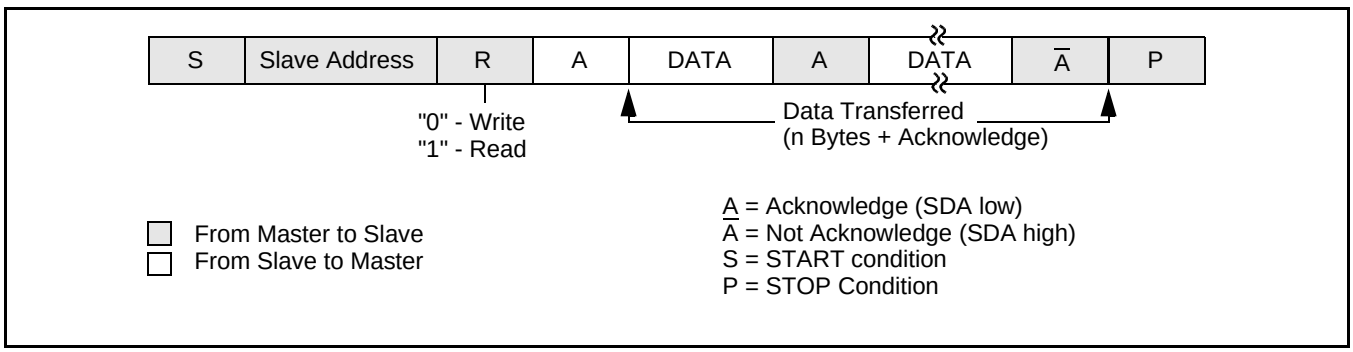


Figure 27: Format of master receiver mode

After a repeated START condition, I^2C may switch to the master transmitter mode.

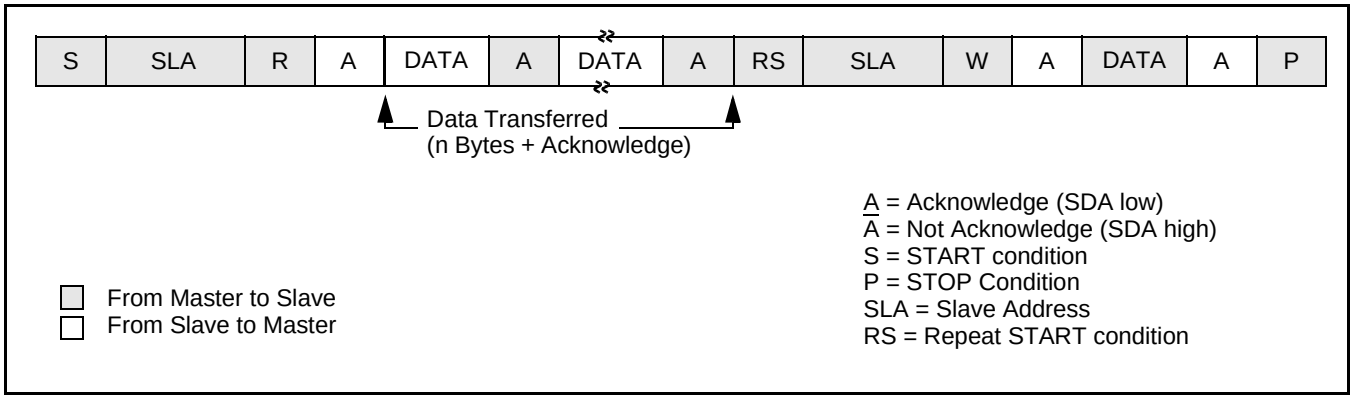


Figure 28: A master receiver switch to master transmitter after sending repeated START

Slave Receiver Mode:

In the slave receiver mode, data bytes are received from a master transmitter. To initialize the slave receiver mode, user should write the Slave Address Register (I2ADR) and write the I^2C Control Set Register (I2CONSET) as shown in Figure 29.

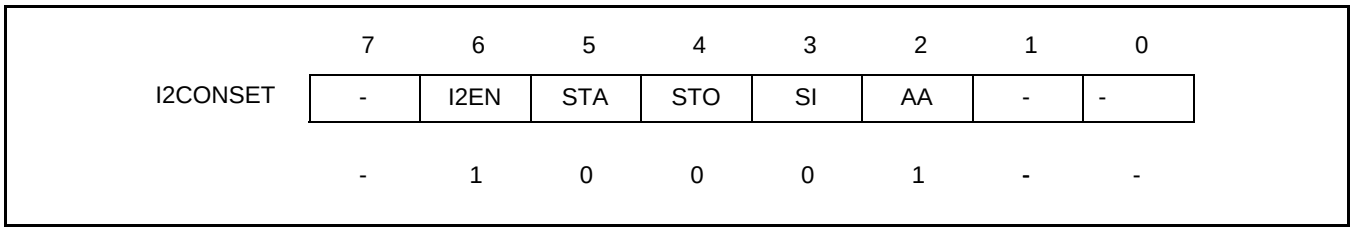


Figure 29: Slave Mode Configuration

I2EN must be set to 1 to enable the I^2C function. AA bit must be set to 1 to acknowledge its own slave address or the general call address. The STA, STO and SI bits are set to 0.

After I2ADR and I2CONSET are initialized, the I^2C interface waits until it is addressed by its own address or general address followed by the data direction bit. If the direction bit is 1(R), it enters slave transmitter mode. After the address and direction bit have been received, the SI bit is set and a valid status code can be read from the Status Register(I2STAT). Refer to Table 5 in "80C51 Family Derivatives 8XC552/562 Overview" datasheet available on-line at

http://www.semiconductors.philips.com/acrobat/various/8XC552_562OVERVIEW_2.pdf

for the status codes and actions.

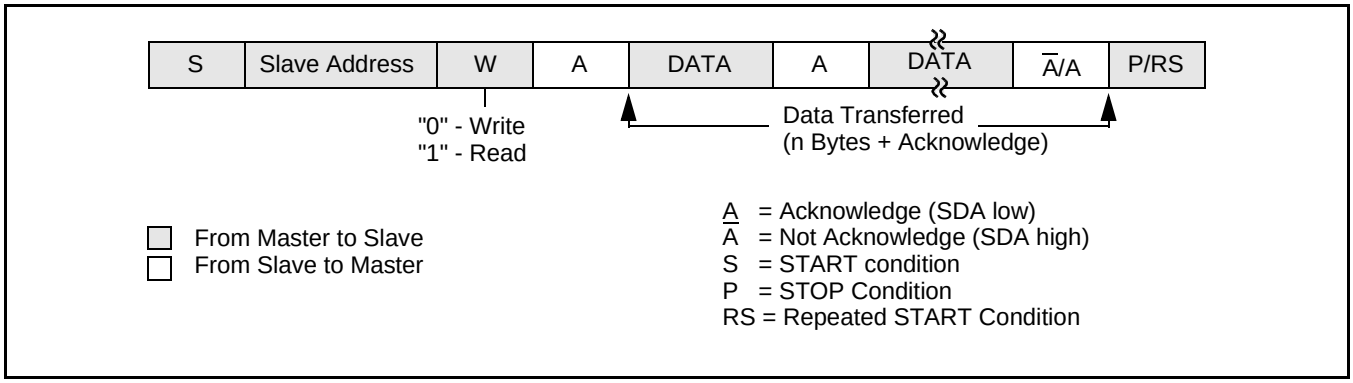


Figure 30: Format of slave receiver mode

Slave Transmitter Mode:

The first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via SDA while the serial clock is input through SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer. In a given application, I^2C may operate as a master and as a slave. In the slave mode, the I^2C hardware looks for its own slave address and the general call address. If one of these addresses is detected, an interrupt is requested. When the microcontroller wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave action is not interrupted. If bus arbitration is lost in the master mode, I^2C switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

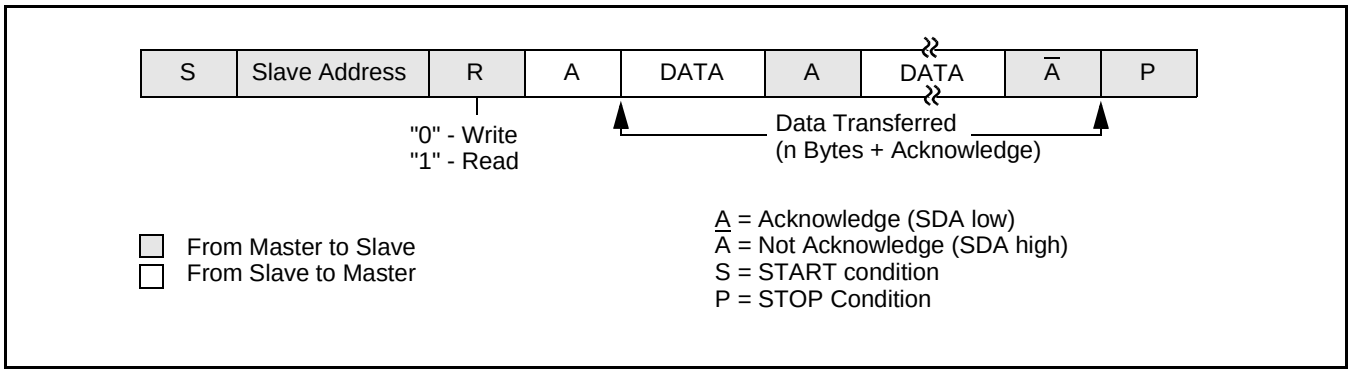


Figure 31: Format of slave transmitter mode

PIN DESCRIPTION

Table 101: I2C Pin Description

Pin Name	Type	Description
SDA	Input/ Output	**Serial Data.** I^2C data input and output. The associated port pin has an open drain output in order to conform to I^2C specifications.
SCL	Input/ Output	**Serial Clock.** I^2C clock input and output. The associated port pin has an open drain output in order to conform to I^2C specifications.

REGISTER DESCRIPTION

The I^2C interface contains 7 registers as shown in Table 102. below.

Table 102: I^2C Register Map

Name	Description	Access	Reset Value*	Address
I2CONSET	I^2C Control Set Register	Read/Set	0	0xE001C000
I2STAT	I^2C Status Register	Read Only	0xF8	0xE001C004
I2DAT	I^2C Data Register	Read/Write	0	0xE001C008
I2ADR	I^2C Slave Address Register	Read/Write	0	0xE001C00C
I2SCLH	SCL Duty Cycle Register High Half Word	Read/Write	0x04	0xE001C010
I2SCLL	SCL Duty Cycle Register Low Half Word	Read/Write	0x04	0xE001C014
I2CONCLR	I^2C Control Clear Register	Clear Only	NA	0xE001C018

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

I²C Control Set Register (I2CONSET - 0xE001C000)

AA is the Assert Acknowledge Flag. When set to 1, an acknowledge (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line on the following situations:

1. The address in the Slave Address Register has been received.
2. The general call address has been received while the general call bit(GC) in I2ADR is set.
3. A data byte has been received while the I²C is in the master receiver mode.
4. A data byte has been received while the I²C is in the addressed slave receiver mode

The AA bit can be cleared by writing 1 to the AAC bit in the I2CONCLR register. When AA is 0, a not acknowledge (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line on the following situations:

1. A data byte has been received while the I²C is in the master receiver mode.
2. A data byte has been received while the I²C is in the addressed slave receiver mode.

SI is the I²C Interrupt Flag. This bit is set when one of the 25 possible I²C states is entered. Typically, the I²C interrupt should only be used to indicate a start condition at an idle slave device, or a stop condition at an idle master device (if it is waiting to use the I²C bus). SI is cleared by writing a 1 to the SIC bit in I2CONCLR register.

STO is the STOP flag. Setting this bit causes the I²C interface to transmit a STOP condition in master mode, or recover from an error condition in slave mode. When STO is 1 in master mode, a STOP condition is transmitted on the I²C bus. When the bus detects the STOP condition, STO is cleared automatically.

In slave mode, setting this bit can recover from an error condition. In this case, no STOP condition is transmitted to the bus. The hardware behaves as if a STOP condition has been received and it switches to "not addressed" slave receiver mode. The STO flag is cleared by hardware automatically.

STA is the START flag. Setting this bit causes the I²C interface to enter master mode and transmit a START condition or transmit a repeated START condition if it is already in master mode.

When STA is 1and the I²C interface is not already in master mode, it enters master mode, checks the bus and generates a START condition if the bus is free. If the bus is not free, it waits for a STOP condition (which will free the bus) and generates a START condition after a delay of a half clock period of the internal clock generator. If the I²C interface is already in master mode and data has been transmitted or received, it transmits a repeated START condition. STA may be set at any time, including when the I²C interface is in an addressed slave mode.

STA can be cleared by writing 1 to the STAC bit in the I2CONCLR register. When STA is 0, no START condition or repeated START condition will be generated.

If STA and STO are both set, then a STOP condition is transmitted on the I²C bus if it the interface is in master mode, and transmits a START condition thereafter. If the I²C interface is in slave mode, an internal STOP condition is generated, but is not transmitted on the bus.

I2EN I^2C Interface Enable. When I2EN is 1, the I^2C function is enabled. I2EN can be cleared by writing 1 to the I2ENC bit in the I2CONCLR register. When I2EN is 0, the I^2C function is disabled.

Table 103: I^2C Control Set Register (I2CONSET - 0xE001C000)

I2CONSET	Function	Description	Reset Value
0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
1	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
2	AA	Assert acknowledge flag	0
3	SI	I^2C interrupt flag	0
4	STO	STOP flag	0
5	STA	START flag	0
6	I2EN	I^2C interface enable	0
7	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

I^2C Control Clear Register (I2CONCLR - 0xE001C018)

Table 104: I^2C Control Clear Register (I2CONCLR - 0xE001C018)

I2CONCLR	Function	Description	Reset Value
0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
1	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
2	AAC	Assert Acknowledge Clear bit. Writing a 1 to this bit clears the AA bit in the I2CONSET register. Writing 0 has no effect.	NA
3	SIC	I^2C Interrupt Clear Bit. Writing a 1 to this bit clears the SI bit in the I2CONSET register. Writing 0 has no effect.	NA
4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
5	STAC	Start flag clear bit. Writing a 1 to this bit clears the STA bit in the I2CONSET register. Writing 0 has no effect.	NA
6	I2ENC	I^2C interface disable. Writing a 1 to this bit clears the I2EN bit in the I2CONSET register.Writing 0 has no effect.	NA
7	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

I²C Status Register (I2STAT - 0xE001C004)

This is a read-only register. It contains the status code of the I²C interface. The least three bits are always 0. There are 26 possible status codes. When the code is F8H, there is no relevant information available and the SI bit is not set. All other 25 status codes correspond to defined I²C states. When any of these states entered, SI bit will be set. Refer to Table 3 to Table 6 in "80C51 Family Derivatives 8XC552/562 Overview" datasheet available on-line at

http://www.semiconductors.philips.com/acrobat/various/8XC552_562OVERVIEW_2.pdf

for a complete list of status codes.

Table 105: I²C Status Register (I2STAT - 0xE001C004)

I2STAT	Function	Description	Reset Value
2:0	Status	These bits are always 0	0
7:3	Status	Status bits	1

I²C Data Register (I2DAT - 0xE001C008)

This register contains the data to be transmitted or the data just received. The CPU can read and write to this register while it is not in the process of shifting a byte. This register can be accessed only when SI bit is set. Data in I2DAT remains stable as long as the SI bit is set. Data in I2DAT is always shifted from right to left: the first bit to be transmitted is the MSB (bit 7), and after a byte has been received, the first bit of received data is located at the MSB of I2DAT.

Table 106: I²C Data Register (I2DAT - 0xE001C008)

I2DAT	Function	Description	Reset Value
7:0	Data	Transmit/Receive data bits	0

I²C Slave Address Register (I2ADR - 0xE001C00C)

This register is readable and writable, and is only used when the I²C is set to slave mode. In master mode, this register has no effect. The LSB of I2ADR is the general call bit. When this bit is set, the general call address (00h) is recognized.

Table 107: I²C Slave Address Register (I2ADR - 0xE001C00C)

I2ADR	Function	Description	Reset Value
0	GC	General Call bit	0
7:1	Address	Slave mode address	0

I^2C SCL Duty Cycle Registers (I2SCLH - 0xE001C010 and I2SCLL - 0xE001C014)

Software must set values for registers I2SCLH and I2SCLL to select the appropriate data rate. I2SCLH defines the number of pclk cycles for SCL high, I2SCLL defines the number of pclk cycles for SCL low. The frequency is determined by the following formula:

Bit Frequency = f_{CLK} / (I2SCLH + I2SCLL)

Where f_{CLK} is the frequency of pclk.

The values for I2SCLL and I2SCLH don't have to be the same. Software can set different duty cycles on SCL by setting these two registers. But the value of the register must ensure that the data rate is in the I^2C data rate range of 0 through 400KHz. So the value of I2SCLL and I2SCLH has some restrictions. Each register value should be greater than or equal to 4.

Table 108: I^2C SCL High Duty Cycle Register (I2SCLH - 0xE001C010)

I2SCLH	Function	Description	Reset Value
15:0	Count	Count for SCL HIGH time period selection	0x 0004

Table 109: I^2C SCL Low Duty Cycle Register (I2SCLL - 0xE001C014)

I2SCLL	Function	Description	Reset Value
15:0	Count	Count for SCL LOW time period selection	0x 0004

Table 110: I2C Clock Rate Selections for VPB Clock Divider = 1

I2SCLL+ I2SCLH	Bit Frequency (kHz) At f_{CCLK} (MHz) & VPB Clock Divider = 1			
	16	20	40	60
8	-	-	-	-
10	-	-	-	-
25	-	-	-	-
50	320.0	400.0	-	-
75	213.333	266.667	-	-
100	160.0	200.0	400.0	-
160	100.0	125.0	250.0	375.0
200	80.0	100.0	200.0	300.0
320	50.0	62.5	125.0	187.5
400	40.0	50.0	100.0	150.0
510	31.373	39.216	78.431	117.647
800	20.0	25.0	50.0	75.0
1280	12.5	15.625	31.25	46.875

Table 111: I2C Clock Rate Selections for VPB Clock Divider = 2

I2SCLL+ I2SCLH	Bit Frequency (kHz) At f_{CCLK} (MHz) & VPB Clock Divider = 2			
	16	20	40	60
8	-	-	-	-
10	-	-	-	-
25	320.0	400.0	-	-
50	160.0	200.0	400.0	-
75	106.667	133.333	266.667	400.0
100	80.0	100.0	200.0	300.0
160	50.0	62.5	125.0	187.5
200	40.0	50.0	100.0	150.0
320	25.0	31.25	62.5	93.75
400	20.0	25.0	50.0	75.0
510	15.686	19.608	39.216	58.824
800	10.0	12.5	25.0	37.5
1280	6.25	7.813	15.625	23.438

Table 112: I2C Clock Rate Selections for VPB Clock Divider = 4

I2SCLL+ I2SCLH	Bit Frequency (kHz) At f_{CCLK} (MHz) & VPB Clock Divider = 4			
	16	20	40	60
8	500.0	-	-	-
10	400.0	-	-	-
25	160.0	200.0	400.0	-
50	80.0	100.0	200.0	300.0
75	53.333	66.667	133.333	200.0
100	40.0	50.0	100.0	150.0
160	25.0	31.25	62.5	93.75
200	20.0	25.0	50.0	75.0
320	12.5	15.625	31.25	46.875
400	10.0	12.5	25.0	37.5
510	7.843	9.804	19.608	29.412
800	5.0	6.25	12.5	18.75
1280	3.125	3.906	7.813	11.719

ARCHITECTURE

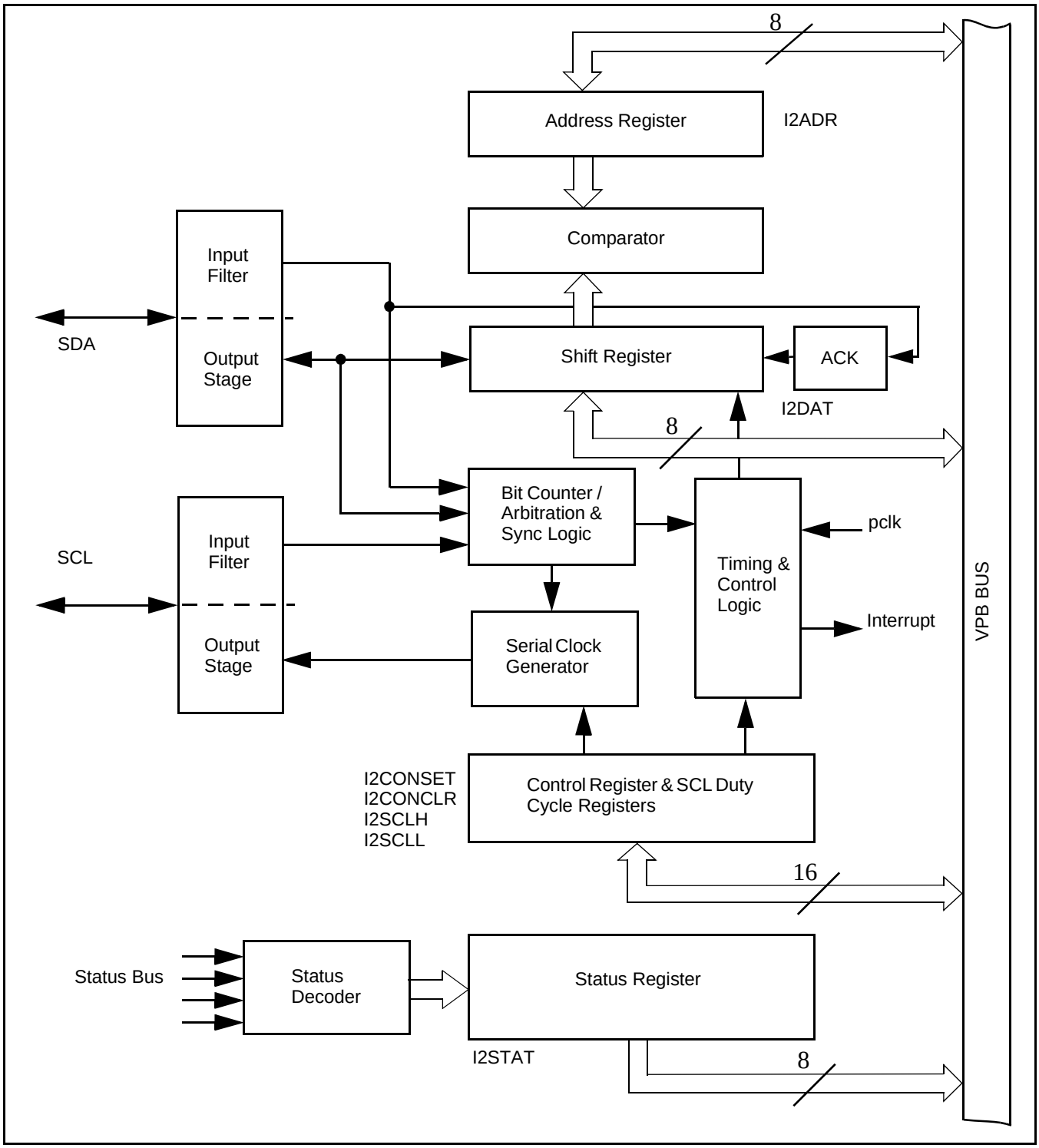


Figure 32: I²C Architecture

13. SPI INTERFACE

FEATURES

- Two complete and independent SPI cintrollers
- Compliant with Serial Peripheral Interface (SPI) specification.
- Synchronous, Serial, Full Duplex Communication.
- Combined SPI master and slave.
- Maximum data bit rate of one eighth of the input clock rate.

DESCRIPTION

SPI Overview

SPI0 and SPI1 are full duplex serial interfaces. They can handle multiple masters and slaves being connected to a given bus. Only a single master and a single slave can communicate on the interface during a given data transfer. During a data transfer the master always sends a byte of data to the slave, and the slave always sends a byte of data to the master.

SPI Data Transfers

Figure 33 is a timing diagram that illustrates the four different data transfer formats that are available with the SPI. This timing diagram illustrates a single 8 bit data transfer. The first thing one should notice in this timing diagram is that it is divided into three horizontal parts. The first part describes the SCK and SSEL signals. The second part describes the MOSI and MISO signals when the CPHA variable is 0. The third part describes the MOSI and MISO signals when the CPHA variable is 1.

In the first part of the timing diagram, note two points. First, the SPI is illustrated wit CPOL set to both 0 and 1. The second point to note is the activation and de-activation of the SSEL signal. When CPHA = 1, the SSEL signal will always go inactive between data transfers. This is not guaranteed when CPHA = 0 (the signal can remain active).

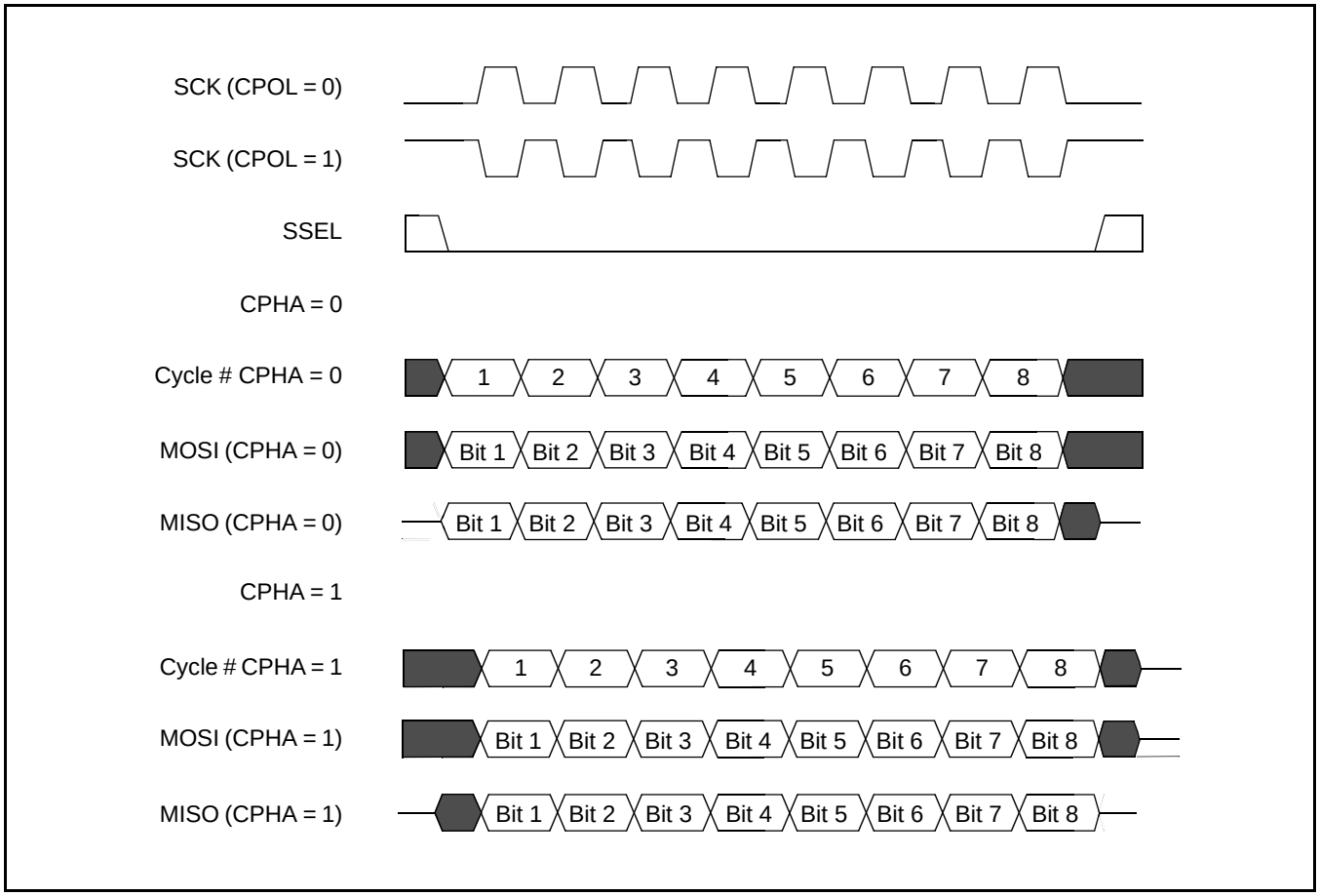


Figure 33: SPI Data Transfer Format (CPHA = 0 and CPHA = 1)

The data and clock phase relationships are summarized in Table 113. This table summarizes the following for each setting of CPOL and CPHA.

- When the first data bit is driven.
- When all other data bits are driven.
- When data is sampled.

Table 113: SPI Data To Clock Phase Relationship

CPOL And CPHA Settings	First Data Driven	Other Data Driven	Data Sampled
CPOL = 0, CPHA = 0	Prior to first SCK rising edge	SCK falling edge	SCK rising edge
CPOL = 0, CPHA = 1	First SCK rising edge	SCK rising edge	SCK falling edge
CPOL = 1, CPHA = 0	Prior to first SCK falling edge	SCK rising edge	SCK falling edge
CPOL = 1, CPHA = 1	First SCK falling edge	SCK falling edge	SCK rising edge

The definition of when an 8 bit transfer starts and stops is dependent on whether a device is a master or a slave, and the setting of the CPHA variable.

When a device is a master, the start of a transfer is indicated by the master having a byte of data that is ready to be transmitted. At this point, the master can activate the clock, and begin the transfer. The transfer ends when the last clock cycle of the transfer is complete.

When a device is a slave, and CPHA is set to 0, the transfer starts when the SSEL signal goes active, and ends when SSEL goes inactive. When a device is a slave, and CPHA is set to 1, the transfer starts on the first clock edge when the slave is selected, and ends on the last clock edge where data is sampled.

SPI Peripheral Details

General Information

There are four registers that control the SPI peripheral. They are described in detail in “Register Description” section.

The SPI control register contains a number of programmable bits used to control the function of the SPI block. The settings for this register must be set up prior to a given data transfer taking place.

The SPI status register contains read only bits that are used to monitor the status of the SPI interface, including normal functions, and exception conditions. The primary purpose of this register is to detect completion of a data transfer. This is indicated by the SPIF bit. The remaining bits in the register are exception condition indicators. These exceptions will be described later in this section.

The SPI data register is used to provide the transmit and receive data bytes. An internal shift register in the SPI block logic is used for the actual transmission and reception of the serial data. Data is written to the SPI data register for the transmit case. There is no buffer between the data register and the internal shift register. A write to the data register goes directly into the internal shift register. Therefore, data should only be written to this register when a transmit is not currently in progress. Read data is buffered. When a transfer is complete, the receive data is transferred to a single byte data buffer, where it is later read. A read of the SPI data register returns the value of the read data buffer.

The SPI clock counter register controls the clock rate when the SPI block is in master mode. This needs to be set prior to a transfer taking place, when the SPI block is a master. This register has no function when the SPI block is a slave.

The I/Os for this implementation of SPI are standard CMOS I/Os. The open drain SPI option is not implemented in this design. When a device is set up to be a slave, its I/Os are only active when it is selected by the SSEL signal being active.

Master Operation

The following sequence describes how one should process a data transfer with the SPI block when it is set up to be the master. This process assumes that any prior data transfer has already completed.

1. Set the SPI clock counter register to the desired clock rate.
2. Set the SPI control register to the desired settings.
3. Write the data to transmitted to the SPI data register. This write starts the SPI data transfer.
4. Wait for the SPIF bit in the SPI status register to be set to 1. The SPIF bit will be set after the last cycle of the SPI data transfer.
5. Read the SPI status register.
6. Read the received data from the SPI data register (optional).
7. Go to step 3 if more data is required to transmit.

Note that a read or write of the SPI data register is required in order to clear the SPIF status bit. Therefore, if the optional read of the SPI data register does not take place, a write to this register is required in order to clear the SPIF status bit.

Slave Operation

The following sequence describes how one should process a data transfer with the SPI block when it is set up to be a slave. This process assumes that any prior data transfer has already completed. It is required that the system clock driving the SPI logic be at least 8X faster than the SPI.

1. Set the SPI control register to the desired settings.
2. Write the data to transmitted to the SPI data register (optional). Note that this can only be done when a slave SPI transfer is not in progress.
3. Wait for the SPIF bit in the SPI status register to be set to 1. The SPIF bit will be set after the last sampling clock edge of the SPI data transfer.
4. Read the SPI status register.
5. Read the received data from the SPI data register (optional).
6. Go to step 2 if more data is required to transmit.

Note that a read or write of the SPI data register is required in order to clear the SPIF status bit. Therefore, at least one of the optional reads or writes of the SPI data register must take place, in order to clear the SPIF status bit.

Exception Conditions

Read Overrun - A read overrun occurs when the SPI block internal read buffer contains data that has not been read by the processor, and a new transfer has completed. The read buffer containing valid data is indicated by the SPIF bit in the status register being active. When a transfer completes, the SPI block needs to move the received data to the read buffer. If the SPIF bit is active (the read buffer is full), the new receive data will be lost, and the read overrun (ROVR) bit in the status register will be activated.

Write Collision - As stated previously, there is no write buffer between the SPI block bus interface, and the internal shift register. As a result, data must not be written to the SPI data register when a SPI data transfer is currently in progress. The time frame where data cannot be written to the SPI data register is from when the transfer starts, until after the status register has been read when the SPIF status is active. If the SPI data register is written in this time frame, the write data will be lost, and the write collision (WCOL) bit in the status register will be activated.

Mode Fault - The SSEL signal must always be inactive when the SPI block is a master. If the SSEL signal goes active, when the SPI block is a master, this indicates another master has selected the device to be a slave. This condition is known as a mode fault. When a mode fault is detected, the mode fault (MODF) bit in the status register will be activated, the SPI signal drivers will be de-activated, and the SPI mode will be changed to be a slave.

Slave Abort - A slave transfer is considered to be aborted, if the SSEL signal goes inactive before the transfer is complete. In the event of a slave abort, the transmit and receive data for the transfer that was in progress are lost, and the slave abort (ABRT) bit in the status register will be activated.

PIN DESCRIPTION

Table 114: SPI Pin Description

Pin Name	Type	Pin Description
SCK1, SCK0	Input/ Output	**Serial Clock.** The SPI is a clock signal used to synchronize the transfer of data across the SPI interface. The SPI is always driven by the master and received by the slave. The clock is programmable to be active high or active low. The SPI is only active during a data transfer. Any other time, it is either in its inactive state, or tri-stated.
SSEL1, SSEL0	Input	**Slave Select.** The SPI slave select signal is an active low signal that indicates which slave is currently selected to participate in a data transfer. Each slave has its own unique slave select signal input. The SSEL must be low before data transactions begin and normally stays low for the duration of the transaction. If the SSEL signal goes high any time during a data transfer, the transfer is considered to be aborted. In this event, the slave returns to idle, and any data that was received is thrown away. There are no other indications of this exception. This signal is not directly driven by the master. It could be driven by a simple general purpose I/O under software control. **Note: LPC2119/2129/2194/2292/2294 configured to operate as SPI master MUST select SSEL functionality on an apropriate pin and have HIGH level on this pin in order to act as a master.**
MISO1, MISO0	Input/ Output	**Master In Slave Out.** The MISO signal is a unidirectional signal used to transfer serial data from the slave to the master. When a device is a slave, serial data is output on this signal. When a device is a master, serial data is input on this signal. When a slave device is not selected, the slave drives the signal high impedance.
MOSI1, MOSI0	Input/ Output	**Master Out Slave In.** The MOSI signal is a unidirectional signal used to transfer serial data from the master to the slave. When a device is a master, serial data is output on this signal. When a device is a slave, serial data is input on this signal.

REGISTER DESCRIPTION

The SPI contains 5 registers as shown in Table 115. All registers are byte, half word and word accessible.

Table 115: SPI Register Map

Generic Name	Description	Access	Reset Value*	SPI0 Address & Name	SPI1 Address & Name
SPCR	SPI Control Register. This register controls the operation of the SPI.	Read/ Write	0	0xE0020000 S0SPCR	0xE0030000 S1SPCR
SPSR	SPI Status Register. This register shows the status of the SPI.	Read Only	0	0xE0020004 S0SPSR	0xE0030004 S1SPSR
SPDR	SPI Data Register. This bi-directional register provides the transmit and receive data for the SPI. Transmit data is provided to the SPI by writing to this register. Data received by the SPI can be read from this register.	Read/ Write	0	0xE0020008 S0SPDR	0xE0030008 S1SPDR
SPCCR	SPI Clock Counter Register. This register controls the frequency of a master's SCK.	Read/ Write	0	0xE002000C S0SPCCR	0xE003000C S1SPCCR
SPINT	SPI Interrupt Flag. This register contains the interrupt flag for the SPI interface.	Read/ Write	0	0xE002001C S0SPINT	0xE003001C S1SPINT

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

SPI Control Register (S0SPCR - 0xE0020000, S1SPCR - 0xE0030000)

The SPCR register controls the operation of the SPI as per the configuration bits setting.

Table 116: SPI Control Register (S0SPCR - 0xE0020000, S1SPCR - 0xE0030000)

SPCR	Function	Description	Reset Value
2:0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	CPHA	Clock phase control determines the relationship between the data and the clock on SPI transfers, and controls when a slave transfer is defined as starting and ending. When 1, data is sampled on the second clock edge of the SCK. A transfer starts with the first clock edge, and ends with the last sampling edge when the SSEL signal is active. When 0, data is sampled on the first clock edge of SCK. A transfer starts and ends with activation and deactivation of the SSEL signal.	0
4	CPOL	Clock polarity control. When 1, SCK is active low. When 0, SCK is active high.	0
5	MSTR	Master mode select. When 1, the SPI operates in Master mode. When 0, the SPI operates in Slave mode.	0
6	LSBF	LSB First controls which direction each byte is shifted when transferred. When 1, SPI data is transferred LSB (bit 0) first. When 0, SPI data is transferred MSB (bit 7) first.	0
7	SPIE	Serial peripheral interrupt enable. When 1, a hardware interrupt is generated each time the SPIF or MODF bits are activated. When 0, SPI interrupts are inhibited.	0

SPI Status Register (S0SPSR - 0xE0020004, S1SPSR - 0xE0030004)

The SPSR register controls the operation of the SPI as per the configuration bits setting.

Table 117: SPI Status Register (S0SPSR - 0xE0020004, S1SPSR - 0xE0030004)

SPSR	Function	Description	Reset Value
2:0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	ABRT	Slave abort. When 1, this bit indicates that a slave abort has occurred. This bit is cleared by reading this register.	0
4	MODF	Mode fault. when 1, this bit indicates that a Mode fault error has occurred. This bit is cleared by reading this register, then writing the SPI control register.	0
5	ROVR	Read overrun. When 1, this bit indicates that a read overrun has occurred. This bit is cleared by reading this register.	0
6	WCOL	Write collision. When 1, this bit indicates that a write collision has occurred. This bit is cleared by reading this register, then accessing the SPI data register.	0
7	SPIF	SPI transfer complete flag. When 1, this bit indicates when a SPI data transfer is complete. When a master, this bit is set at the end of the last cycle of the transfer. When a slave, this bit is set on the last data sampling edge of the SCK. This bit is cleared by first reading this register, then accessing the SPI data register. **Note:** this is not the SPI interrupt flag. This flag is found in the SPINT registrer.	0

SPI Data Register (S0SPDR - 0xE0020008, S1SPDR - 0xE0030008)

This bi-directional data register provides the transmit and receive data for the SPI. Transmit data is provided to the SPI by writing to this register. Data received by the SPI can be read from this register. When a master, a write to this register will start a SPI data transfer. Writes to this register will be blocked from when a data transfer starts to when the SPIF status bit is set, and the status register has not been read.

Table 118: SPI Data Register (S0SPDR - 0xE0020008, S1SPDR - 0xE0030008)

SPDR	Function	Description	Reset Value
7:0	Data	SPI Bi-directional data port	0

SPI Clock Counter Register (S0SPCCR - 0xE002000C, S1SPCCR - 0xE003000C)

This register controls the frequency of a master's SCK. The register indicates the number of pclk cycles that make up an SPI clock. The value of this register must always be an even number. As a result, bit 0 must always be 0. The value of the register must also always be greater than or equal to 8. Violations of this can result in unpredictable behavior.

Table 119: SPI Clock Counter Register (S0SPCCR - 0xE002000C, S1SPCCR - 0xE003000C)

SPCCR	Function	Description	Reset Value
7:0	Counter	SPI Clock counter setting	0

The SPI rate may be calculated as: PCLK rate / SPCCR value. The pclk rate is CCLK / VPB divider rate as determined by the VPBDIV register contents.

SPI Interrupt Register (S0SPINT - 0xE002001C, S1SPINT - 0xE003001C)

This register contains the interrupt flag for the SPI interface.

Table 120: SPI Interrupt Register (S0SPINT - 0xE002001C, S1SPINT - 0xE003001C)

SPINT	Function	Description	Reset Value
0	SPI Interrupt	SPI interrupt flag. Set by the SPI interface to generate an interrupt. Cleared by writing a 1 to this bit. Note: this bit will be set once when SPIE=1 and at least one of SPIF and WCOL bits is 1. However, only when SPI Interrupt bit is set and SPI Interrupt is enabled in the VIC, SPI based interrupt can be processed by interrupt handling software.	0
7:1	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

ARCHITECTURE

The block diagram of the SPI solution implemented in SPI0 and SPI1 interfaces is shown in the Figure 34.

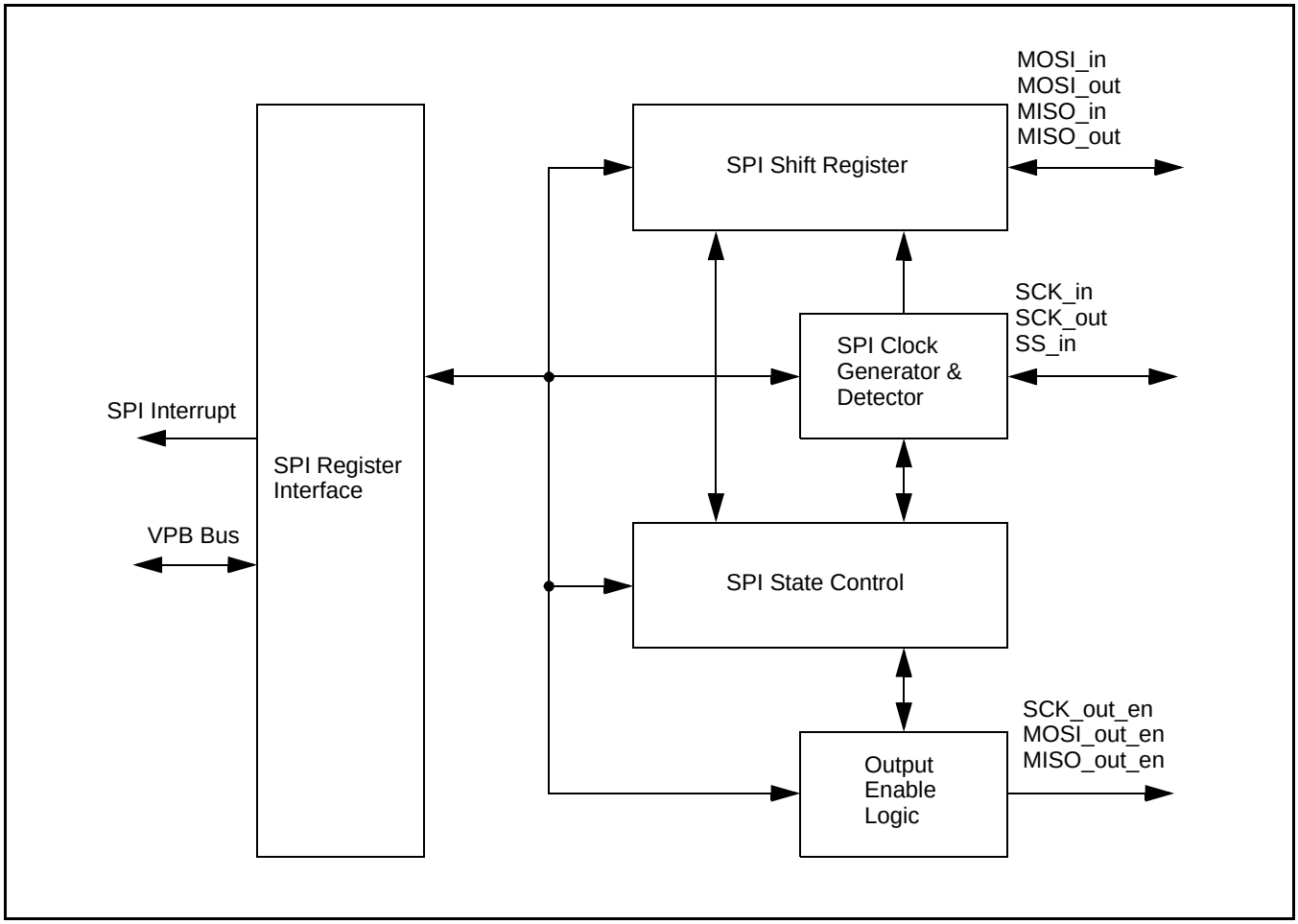


Figure 34: SPI Block Diagram

14. CAN CONTROLLERS AND ACCEPTANCE FILTER

The Controller Area Network (CAN) is a serial communications protocol which efficiently supports distributed real-time control with a very high level of security. Its domain of application ranges from high speed networks to low cost multiplex wiring.

The LPC2119/2129/2194/2292/2294 CAN block is intended to support multiple CAN buses simultaneously, allowing the device to be used as a gateway, switch, or router among a number of CAN buses in industrial or automotive applications.

CAN CONTROLLERS

Each CAN Controller has a register structure similar to the Philips SJA1000 and the PeliCAN Library block, but the 8-bit registers of those devices have been combined in 32 bit words to allow simultaneous access in the ARM environment. The main operational difference is that the recognition of received Identifiers, known in CAN terminology as Acceptance Filtering, has been removed from the CAN controllers and centralized in a global Acceptance Filter. This Acceptance Filter is described after the CAN Controllers.

FEATURES

- 2 or 4 (LPC2119/2129/2292 or LP2194/2294) CAN controllers and buses (2 in 64-pin and 2 or 4 in 144-pin packages)
- Data rates to 1 Mbits/second on each bus
- 32-bit register and RAM access
- Compatible with CAN specification 2.0B, ISO 11898-1
- Global Acceptance Filter recognizes 11- and 29-bit Rx Identifiers for all CAN buses
- Acceptance Filter can provide FullCAN-style automatic reception for selected Standard Identifiers

PIN DESCRIPTION

Table 121: CAN Pin Descriptions

Pin Name	Type	Description
RX4-1	Inputs	**Serial Inputs**. From CAN transceivers. **Note:** RX2 and RX1 are available in all parts having CAN module(s). RX4 and RX3 are available in LPC2194/2294 only.
TX4-1	Outputs	**Serial Outputs**. To CAN transceivers. **Note:** TX2 and TX1 are available in all parts having CAN module(s). TX4 and TX3 are available in LPC2194/2294 only.

MEMORY MAP OF THE CAN BLOCK

The CAN Controllers and Acceptance Filter occupy a number of VPB slots, as follows:

Table 122: Memory Map of the CAN Block

Address Range	Used For
E003 8000 - 87FF	Acceptance Filter RAM
E003 C000 - C017	Acceptance Filter Registers
E004 0000 - 000B	Central CAN Registers
E004 4000 - 405F	CAN Controller 1 Registers
E004 8000 - 805F	CAN Controller 2 Registers
E004 C000 - C05F	CAN Controller 3 Registers (LPC2194/2294 only)
E005 0000 - 005F	CAN Controller 4 Registers (LPC2194/2294 only)

CAN CONTROLLER REGISTERS

CAN block implements the registers shown in Table 123 and 124. More detailed descriptions follow.

Table 123: CAN Acceptance Filter and Central CAN Registers

Name	Description	Access	Reset Value	Address
AFMR	Acceptance Filter Register	R/W	1	0xE003 C000
SFF_sa	Standard Frame Individual Start Address Register	R/W	0	0xE003 C004
SFF_GRP_sa	Standard Frame Group Start Address Register	R/W	0	0xE003 C008
EFF_sa	Extended Frame Start Address Register	R/W	0	0xE003 C00C
EFF_GRP_sa	Extended Frame Group Start Address Register	R/W	0	0xE003 C010
ENDofTable	End of AF Tables register	R/W	0	0xE003 C014
LUTerrAd	LUT Error Address register	RO	0	0xE003 C018

Table 123: CAN Acceptance Filter and Central CAN Registers

Name	Description	Access	Reset Value	Address
LUTerr	LUT Error Register	RO	0	0xE003 C01C
CANTxSR	CAN Central Transmit Status Register	RO	0x003F 3F00	0xE004 0000
CANRxSR	CAN Central Receive Status Register	RO	0	0xE004 0004
CANMSR	CAN Central Miscellaneous Register	RO	0	0xE004 0008

Table 124: CAN1, CAN2, CAN3 and CAN4 Controller Register Map

Generic Register Name	Description	Access	CAN1 Address & Name	CAN2 Address & Name	CAN3 Address & Name	CAN4 Address & Name
CANMOD	Controls the operating mode of the CAN Controller.	R/W	0xE004 4000 C1MOD	0xE004 8000 C2MOD	0xE004 C000 C3MOD	0xE005 0000 C4MOD
CANCMR	Command bits that affect the state of the CAN Controller	WO	0xE004 4004 C1CMR	0xE004 8004 C2CMR	0xE004 C004 C3CMR	0xE005 0004 C4CMR
CANGSR	Global Controller Status and Error Counters	RO[a]	0xE004 4008 C1GSR	0xE004 8008 C2GSR	0xE004 C008 C3GSR	0xE005 0008 C4GSR
CANICR	Interrupt status, Arbitration Lost Capture, Error Code Capture	RO	0xE004 400C C1ICR	0xE004 800C C2ICR	0xE004 C00C C3ICR	0xE005 000C C4ICR
CANIER	Interrupt Enable	R/W	0xE004 4010 C1IER	0xE004 8010 C2IER	0xE004 C010 C3IER	0xE005 0010 C4IER
CANBTR	Bus Timing	R/W[b]	0xE004 4014 C1BTR	0xE004 8014 C2BTR	0xE004 C014 C3BTR	0xE005 0014 C4BTR
CANEWL	Error Warning Limit	R/W[b]	0xE004 4018 C1EWL	0xE004 8018 C2EWL	0xE004 C018 C3EWL	0xE005 0018 C4EWL
CANSR	Status Register	RO	0xE004 401C C1SR	0xE004 801C C2SR	0xE004 C01C C3SR	0xE005 001C C4SR
CANRFS	Receive frame status	R/W[b]	0xE004 4020 C1RFS	0xE004 8020 C2RFS	0xE004 C020 C3RFS	0xE005 0020 C4RFS
CANRID	Received Identifier	R/W[b]	0xE004 4024 C1RID	0xE004 8024 C2RID	0xE004 C024 C3RID	0xE005 0024 C4RID
CANRDA	Received data bytes 1-4	R/W[b]	0xE004 4028 C1RDA	0xE004 8028 C2RDA	0xE004 C028 C3RDA	0xE005 0028 C4RDA
CANRDB	Received data bytes 5-8	R/W[b]	0xE004 402C C1RDB	0xE004 802C C2RDB	0xE004 C02C C3RDB	0xE005 002C C4RDB
CANTFI1	Transmit frame info (1)	R/W	0xE004 4030 C1TFI1	0xE004 8030 C2TFI1	0xE004 C030 C3TFI1	0xE005 0030 C4TFI1
CANTID1	Transmit Identifier (1)	R/W	0xE004 4034 C1TID1	0xE004 8034 C2TID1	0xE004 C034 C3TID1	0xE005 0034 C4TID1
CANTDA1	Transmit data bytes 1-4 (1)	R/W	0xE004 4038 C1TDA1	0xE004 8038 C2TDA1	0xE004 C038 C3TDA1	0xE005 0038 C4TDA1

Table 124: CAN1, CAN2, CAN3 and CAN4 Controller Register Map

Generic Register Name	Description	Access	CAN1 Address & Name	CAN2 Address & Name	CAN3 Address & Name	CAN4 Address & Name
CANTDB1	Transmit data bytes 5-8 (1)	R/W	0xE004 403C C1TDB1	0xE004 803C C2TDB1	0xE004 C03C C3TDB1	0xE005 003C C4TDB1
CANTFI2	Transmit frame info (2)	R/W	0xE004 4040 C1TFI2	0xE004 8040 C2TFI2	0xE004 C040 C3TFI2	0xE005 0040 C4TFI2
CANTID2	Transmit Identifier (2)	R/W	0xE004 4044 C1TID2	0xE004 8044 C2TID2	0xE004 C044 C3TID2	0xE005 0044 C4TID2
CANTDA2	Transmit data bytes 1-4 (2)	R/W	0xE004 4048 C1TDA2	0xE004 8048 C2TDA2	0xE004 C048 C3TDA2	0xE005 0048 C4TDA2
CANTDB2	Transmit data bytes 5-8 (2)	R/W	0xE004 404C C1TDB2	0xE004 804C C2TDB2	0xE004 C04C C3TDB2	0xE005 004C C4TDB2
CANTFI3	Transmit frame info (3)	R/W	0xE004 4050 C1TFI3	0xE004 8050 C2TFI3	0xE004 C050 C3TFI3	0xE005 0050 C4TFI3
CANTID3	Transmit Identifier (3)	R/W	0xE004 4054 C1TID3	0xE004 8054 C2TID3	0xE004 C054 C3TID3	0xE005 0054 C4TID3
CANTDA3	Transmit data bytes 1-4 (3)	R/W	0xE004 4058 C1TDA3	0xE004 8058 C2TDA3	0xE004 C058 C3TDA3	0xE005 0058 C4TDA3
CANTDB3	Transmit data bytes 5-8 (3)	R/W	0xE004 405C C1TDB3	0xE004 805C C2TDB3	0xE004 C05C C3TDB3	0xE005 005C C4TDB3

a. The error counters can only be written when RM in CANMOD is 1.

b. These registers can only be written when RM in CANMOD is 1.

In the following register tables, the column “Reset Value” shows how a hardware reset affects each bit or field, while the column “RM Set” indicates how each bit or field is affected if software sets the RM bit, or RM is set because of a Bus-Off condition. Note that while hardware reset sets RM, in this case the setting noted in the “Reset Value” column prevails over that shown in the “RM Set” column, in the few bits where they differ. In both columns, X indicates the bit or field is unchanged.

Mode Register (CANMOD - 0xE00x x000)

This register controls the basic operating mode of the CAN Controller. Bits not listed read as 0 and should be written as 0. See Table 124 for details on specific CAN channel register address.

Table 125: CAN Mode Register (CANMOD - 0xE00x x000)

CANMOD	Name	Function	Reset Value	RM Set
0	RM	0: the CAN Controller operates, and certain registers can not be written. 1: Reset Mode -- CAN operation is disabled, and writable registers can be written.	1	1
1	LOM	0: the CAN controller acknowledges a successfully-received message on its CAN. 1: Listen Only Mode -- the controller gives no acknowledgment on CAN, even if a message is successfully received. Messages cannot be sent, and the controller operates in “error passive” mode. This mode is intended for software bit rate detection and “hot plugging”.	0	X
2	STM	0: a transmitted message must be acknowledged to be considered successful. 1: Self Test Mode -- the controller will consider a Tx message successful if there is no acknowledgment. Use this state in conjunction with the SRR bit in CANCMR.	0	X
3	TPM	0: the priority of the 3 Transmit Buffers depends on their CAN IDs. 1: the priority of the 3 Transmit Buffers depends on their Tx Priority fields.	0	X
4	SM	0: normal operation 1: Sleep Mode -- the CAN controller sleeps if it is not requesting an interrupt, and there is no bus activity. See the Sleep Mode description on page 203.	0	0
5	RPM	0: RX and TX pins are Low for a dominant bit. 1: Reverse Polarity Mode -- RX pins are High for a dominant bit.	0	0
7	TM	0: normal operation 1: Test Mode. The state of the RX pin is clocked onto the TX pin.	0	0

Note 1: The LOM and STM bits can only be written if the RM bit is 1 prior to the write operation.

Command Register (CANCMR - 0xE00x x004)

Writing to this write-only register initiates an action. Bits not listed should be written as 0. Reading this register yields zeroes. See Table 124 for details on specific CAN channel register address.

Table 126: CAN Command Register (CANCMR - 0xE00x x004)

CANCMR	Name	Function
0	TR	1: Transmission Request -- the message, previously written to the CANTFI, CANTID, and optionally the CANTDA and CANTDB registers, is queued for transmission.
1	AT	1: Abort Transmission -- if not already in progress, a pending Transmission Request is cancelled. If this bit and TR are set in the same write operation, frame transmission is attempted once, and no retransmission is attempted if an error is flagged nor if arbitration is lost.
2	RRB	1: Release Receive Buffer -- the information in the CANRFS, CANRID, and if applicable the CANRDA and CANRDB registers is released, and becomes eligible for replacement by the next received frame. If the next received frame is not available, writing this command clears the RBS bit in CANSR.
3	CDO	1: Clear Data Overrun -- The Data Overrun bit in CANSR is cleared.
4	SRR	1: Self Reception Request -- the message, previously written to the CANTFS, CANTID, and optionally the CANTDA and CANTDB registers, is queued for transmission. This differs from the TR bit above in that the receiver is not disabled during the transmission, so that it receives the message if its Identifier is recognized by the Acceptance Filter.
5	STB1	1: Select Tx Buffer 1 for transmission
6	STB2	1: Select Tx Buffer 2 for transmission
7	STB3	1: Select Tx Buffer 3 for transmission

Global Status Register (CANGSR - 0xE00x x008)

This register is read-only, except that the Error Counters can be written when the RM bit in the CANMOD register is 1. Bits not listed read as 0 and should be written as 0. See Table 124 for details on specific CAN channel register address.

Table 127: CAN Global Status Register (CANGSR - 0xE00x x008)

CANGSR	Name	Function	Reset Value	RM Set
0	RBS	1: Receive Buffer Status -- a received message is available in the CANRFS, CANRID, and if applicable the CANRDA and CANRDB registers. This bit is cleared by the Release Receive Buffer command in CANCMR, if no subsequent received message is available.	0	0
1	DOS	1: Data Overrun Status -- a message was lost because the preceding message to this CAN controller was not read and released quickly enough. 0: No data overrun has occurred since the last Clear Data Overrun command was written to CANCMR (or since Reset).	0	0
2	TBS	1: Transmit Buffer Status -- no transmit message is pending for this CAN controller (in any of the 3 Tx buffers), and software may write to any of the CANTFI, CANTID, CANTDA, and CANTDB registers. 0: as least one previously-queued message for this CAN controller has not yet been sent, and therefore software should not write to the CANTFI, CANTID, CANTDA, nor CANTDB registers of that (those) Tx buffer(s).	1	X
3	TCS	1: Transmit Complete Status -- all requested transmission(s) has (have) been successfully completed. 0: at least one requested transmission has not been successfully completed.	1	0
4	RS	1: Receive Status: the CAN controller is receiving a message.	0	0
5	TS	1: Transmit Status: The CAN controller is sending a message	0	0
6	ES	1: Error Status: one or both of the Transmit and Receive Error Counters has reached the limit set in the Error Warning Limit register.	0	0
7	BS	1: Bus Status: the CAN controller is currently prohibited from bus activity because the Transmit Error Counter reached its limiting value of 255.	0	0
23:16	RXERR	The current value of the Rx Error Counter.	0	X
31:24	TXERR	The current value of the Tx Error Counter.	0	X

Interrupt and Capture Register (CANICR - 0xE00x x00C)

Bits in this register indicate information about events on the CAN bus. This register is read-only. Bits not listed read as 0 and should be written as 0. Bits 1-9 clear when they are read. See Table 124 for details on specific CAN channel register address.

Bits 16-23 are captured when a bus error occurs. At the same time, if the BEIE bit in CANIER is 1, the BEI bit in this register is set, and a CAN interrupt can occur. Bits 24-31 are captured when CAN arbitration is lost. At the same time, if the ALIE bit in CANIER is 1, the ALI bit in this register is set, and a CAN interrupt can occur. Once either of these bytes is captured, its value will remain the same until it is read, at which time it is released to capture a new value.

The clearing of bits 1-9 and the releasing of bits 16-23 and 24-31 all occur on any read from CANICR, regardless of whether part or all of the register is read. This means that software should **always read CANICR as a word**, and process and deal with all bits of the register as appropriate for the application.

Table 128: CAN Interrupt and Capture Register (CANICR - 0xE00x x00C)

CANICR	Name	Function	Reset Value	RM Set
0	RI	1: Receive Interrupt -- this bit is set whenever the RBS bit in CANSR and the RIE bit in CANIER are both 1, indicating that a received message is available.=.	0	0
1	TI1	1: Transmit Interrupt 1 -- this bit is set when the TBS1 bit in CANSR goes from 0 to 1, indicating that Transmit buffer 1 is available, and the TIE1 bit in CANIER is 1.	0	0
2	EI	1: Error Warning Interrupt -- this bit is set on every change (set or clear) of the Error Status or Bus Status bit in CANSR, if the EIE bit in CAN is 1 at the time of the change.	0	X
3	DOI	1: Data Overrun Interrupt -- this bit is set when the DOS bit in CANSR goes from 0 to 1, if the DOIE bit in CANIE is 1.	0	0
4	WUI	1: Wake-Up Interrupt: this bit is set if the CAN controller is sleeping and bus activity is detected, if the WUIE bit in CANIE is 1.	0	0
5	EPI	1: Error Passive Interrupt -- this bit is set if the EPIE bit in CANIE is 1, and the CAN controller switches between Error Passive and Error Active mode in either direction.	0	0
6	ALI	1: Arbitration Lost Interrupt -- this bit is set if the ALIE bit in CANIE is 1, and the CAN controller loses arbitration while attempting to transmit.	0	0
7	BEI	1: Bus Error Interrupt -- this bit is set if the BEIE bit in CANIE is 1, and the CAN controller detects an error on the bus.	0	X
8	IDI	1: ID Ready Interrupt -- this bit is set if the IDIE bit in CANIE is 1, and a CAN Identifier has been received.	0	0
9	TI2	1: Transmit Interrupt 2 -- this bit is set when the TBS2 bit in CANSR goes from 0 to 1, indicating that Transmit buffer 2 is available, and the TIE2 bit in CANIER is 1.	0	0
10	TI3	1: Transmit Interrupt 1 -- this bit is set when the TBS3 bit in CANSR goes from 0 to 1, indicating that Transmit buffer 3 is available, and the TIE3 bit in CANIER is 1.	0	0

Table 128: CAN Interrupt and Capture Register (CANICR - 0xE00x x00C)

CANICR	Name	Function	Reset Value	RM Set
20:16	ERRBIT	Error Code Capture: when the CAN controller detects a bus error, the location of the error within the frame is captured in this field. The value reflects an internal state variable. 00010: ID28:21 00011: Start of Frame 00100: SRTR Bit 00101: IDE Bit 00110: ID20:18 00111: ID17:13 01000: CRC 01001: Reserved Bit 0 01010: Data field 01011: DLC 01100: RTR bit 01101: Reserved Bit 1 01110: ID4:0 01111: ID12:5 10001: Active Error flag 10010: Intermission 10011: dominant OK bits 10110: Passive error flag 10111: Error delimiter 11000: CRC delimiter 11001: Ack slot 11010: End of Frame 11011: Ack delimiter 11100: Overload flag Reading this byte enables another Bus Error Interrupt.	0	X
21	ERRDIR	When the CAN controller detects a bus error, the direction of the current bit is captured in this bit. 1=receiving, 0=transmitting.	0	X
23:22	ERRC	When the CAN controller detects a bus error, the type of error is captured in this field: 00=bit error, 01=Form error, 10=Stuff error, 11=other error.	0	X
28:24	ALCBIT	Each time arbitration is lost while trying to send on the CAN, the bit number within the frame is captured into this field. 0 indicates arbitration loss in the first (MS) bit of the Identifier … 31 indicates loss in the RTR bit of an extended frame. After this byte is read, the ALI bit is cleared and a new Arbitration Lost interrupt can occur.	0	X

Interrupt Enable Register (CANIER - 0xE00x x010)

This read/write register controls whether various events on the CAN controller will result in an interrupt. Bits 7:0 in this register correspond 1-to-1 with bits 7:0 in the CANICR register. See Table 124 for details on specific CAN channel register address.

Table 129: CAN Interrupt Enable Register (CANIER - 0xE00x x010)

CANIER	Name	Function	Reset Value	RM Set
0	RIE	Receiver Interrupt Enable.	0	X
1	TIE1	Transmit Interrupt Enable (1)	0	X
2	EIE	Error Warning Interrupt Enable	0	X
3	DOIE	Data Overrun Interrupt Enable	0	X
4	WUIE	Wake-Up Interrupt Enable	0	X
5	EPIE	Error Passive Interrupt Enable	0	X
6	ALIE	Arbitration Lost Interrupt Enable	0	X
7	BEIE	Bus Error Interrupt Enable	0	X
8	IDIE	ID Ready Interrupt Enable	0	X
9	TIE2	Transmit Interrupt Enable (2)	0	X
10	TIE3	Transmit Interrupt Enable (3)	0	X

Bus Timing Register (CANBTR - 0xE00x x014)

This register controls how various CAN timings are derived from the VPB clock. It can be read at any time, but can only be written if the RM bit in CANmod is 1. See Table 124 for details on specific CAN channel register address.

Table 130: CAN Bus Timing Register (CANBTR - 0xE00x x014)

CANBTR	Name	Function	Reset Value	RM Set
0:9	BRP	Baud Rate Prescaler. The VPB clock is divided by (this value plus one) to produce the CAN clock.	0	X
15:14	SJW	The Synchronization Jump Width is (this value plus one) CAN clocks.	0	X
19:16	TSEG1	The delay from the nominal Sync point to the sample point is (this value plus one) CAN clocks.	1100	X
22:20	TSEG2	The delay from the sample point to the next nominal sync point is (this value plus one) CAN clocks. The nominal CAN bit time is (this value plus the value in TSEG1 plus 3) CAN clocks.	001	X
23	SAM	1: the bus is sampled 3 times (recommended for low to medium speed buses) 0: the bus is sampled once (recommended for high speed buses)	0	X

Error Warning Limit Register (CANEWL - 0xE00x x018)

This register sets a limit on Tx or Rx errors at which an interrupt can occur. It can be read at any time, but can only be written if the RM bit in CANmod is 1. See Table 124 for details on specific CAN channel register address.

Table 131: CAN Error Warning Limit Register (CANEWL - 0xE00x x018)

CANEWL	Name	Function	Reset Value	RM Set
7:0	EWL	During CAN operation, this value is compared to both the Tx and Rx Error Counters. If either of these counter matches this value, the Error Status (ES) bit in CANSR is set.	96_{10}=0x60	X

Status Register (CANSR - 0xE00x x01C)

This register contains three status bytes, in which the bits not related to tranmission are identical to the corresponding bits in the Global Status Register, while those relating to transmission reflect the status of each of the 3 Tx Buffers. See Table 124 for details on specific CAN channel register address.

Table 132: CAN Status Register (CANSR - 0xE00x x01C)

CANSR	Name	Function	Reset Value	RM Set
0, 8, 16	RBS	These bits are identical to the RSB bit in the GSR.	0	0
1, 9, 17	DOS	These bits are identical to the DOS bit in the GSR.	0	0
2, 10, 18	TBS1, TBS2, TBS3	1: software may write a message into the CANTFI, CANTID, CANTDA, and CANTDB registers for this Tx Buffer. 0: software should not write to any of the CANTFI, CANTID, CANTDA, and CANTDB registers for this Tx Buffer.	1	X
3, 11, 19	TCS1, TCS2, TCS3	1: The previously requested transmission for this Tx Buffer has been successfully completed. 0: The previously requested transmission for this Tx Buffer is not complete.	1	0

Table 132: CAN Status Register (CANSR - 0xE00x x01C)

CANSR	Name	Function	Reset Value	RM Set
4, 12, 20	RS	These bits are identical to the RS bit in the GSR.	0	0
5, 13, 21	TS1, TS2, TS3	1: The CAN Controller is transmitting a message from this Tx Buffer.	0	0
6, 14, 22	ES	These bits are identical to the ES bit in the GSR.	0	0
7, 15, 23	BS	These bits are identical to the BS bit in the GSR.	0	0

Rx Frame Status Register (CANRFS - 0xE00x x020)

This register defines the characteristics of the current received message. It is read-only in normal operation, but can be written for testing purposes if the RM bit in CANMOD is 1. See Table 124 for details on specific CAN channel register address.

Table 133: CAN Rx Frame Status Register (CANRFS - 0xE00x x020)

CANRFS	Name	Function	Reset Value	RM Set
9:0	ID Index	If the BP bit (below) is 0, this value is the zero-based number of the Lookup Table RAM entry at which the Acceptance Filter matched the received Identifier. Disabled entries in the Standard tables are included in this numbering, but will not be matched. See the section “Examples of Acceptance Filter Tables and ID Index Values“ on page 209 for examples of ID Index values.	0	X
10	BP	If this bit is 1, the current message was received in AF Bypass mode, and the ID Index field (above) is meaningless.	0	X
19:16	DLC	The field contains the Data Length Code (DLC) field of the current received message. When RTR=0, this is related to the number of data bytes available in the CANRDA and CANRDB registers as follows: 0000-0111 = 0 to 7 bytes 1000-1111 = 8 bytes With RTR=1, this value indicates the number of data bytes requested to be sent back, with the same encoding.	0	X
30	RTR	This bit contains the Remote Transmission Request bit of the current received message. 0 indicates a Data Frame, in which (if DLC is non-zero) data can be read from the CANRDA and possibly the CANRDB registers. 1 indicates a Remote frame, in which case the DLC value identifies the number of data bytes requested to be sent using the same Identifier.	0	X
31	FF	A 0 in this bit indicates that the current received message included an 11-bit Identifier, while a 1 indicates a 29-bit Identifier. This affects the contents of the CANid register described below.	0	X

Rx Identifier Register (CANRID - 0xE00x x024)

This register contains the Identifier field of the current received message. It is read-only in normal operation, but can be written for testing purposes if the RM bit in CANmod is 1. It has two different formats depending on the FF bit in CANRFS. See Table 124 for details on specific CAN channel register address.

Table 134: CAN Rx Identifier Register when FF=0 (CANRID - 0xE00x x024)

CANRID	Name	Function	Reset Value	RM Set
10:0	ID	The 11-bit Identifier field of the current received message. In CAN 2.0A, these bits are called ID10-0, while in CAN 2.0B they’re called ID29-18.	0	X

Table 135: CAN Rx Identifier Register when FF=1 (CANRID - 0xE00x x024)

CANRID	Name	Function	Reset Value	RM Set
28:0	ID	The 29-bit Identifier field of the current received message. In CAN 2.0B these bits are called ID29-0.	0	X

Rx Data Register A (CANRDA - 0xE00x x028)

This register contains the first 1-4 Data bytes of the current received message. It is read-only in normal operation, but can be written for testing purposes if the RM bit in CANMOD is 1. See Table 124 for details on specific CAN channel register address.

Table 136: CAN Rx Data Register 1 (CANRDA - 0xE00x x028)

CANRDA	Name	Function	Reset Value	RM Set
7:0	Data 1	If the DLC field in CANRFS >= 0001, this contains the first Data byte of the current received message.	0	X
15:8	Data 2	If the DLC field in CANRFS >= 0010, this contains the second Data byte of the current received message.	0	X
23:16	Data 3	If the DLC field in CANRFS >= 0011, this contains the third Data byte of the current received message.	0	X
31:24	Data 4	If the DLC field in CANRFS >= 0100, this contains the fourth Data byte of the current received message.	0	X

Rx Data Register B (CANRDB - 0xE00x x02C)

This register contains the 5th through 8th Data bytes of the current received message. It is read-only in normal operation, but can be written for testing purposes if the RM bit in CANMOD is 1. See Table 124 for details on specific CAN channel register address.

Table 137: CAN Rx Data Register B (CANRDB - 0xE00x x02C)

CANRDB	Name	Function	Reset Value	RM Set
7:0	Data 5	If the DLC field in CANRFS >= 0101, this contains the 5th Data byte of the current received message.	0	X
15:8	Data 6	If the DLC field in CANRFS >= 0110, this contains the 6th Data byte of the current received message.	0	X
23:16	Data 7	If the DLC field in CANRFS >= 0111, this contains the 7th Data byte of the current received message.	0	X
31:24	Data 8	If the DLC field in CANRFS >= 1000, this contains the 8th Data byte of the current received message.	0	X

Tx Frame Information Register (CANTFI1, 2, 3 - 0xE00x x030, 40, 50)

When the corresponding TBS bit in CANSR is 1, software can write to one of these registers to define the format of the next transmit message for that Tx buffer. Bits not listed read as 0 and should be written as 0. See Table 124 for details on specific CAN channel register address.

Table 138: CAN Tx Frame Information Register (CANTFI1, 2, 3 - 0xE00x x030, 40, 50)

CANTFI	Name	Function	Reset Value	RM Set
7:0	PRIO	If the TPM bit in the CANMOD register is 1, enabled Tx Buffers contend for the right to send their messages based on this field. The lowest binary value has priority.		
19:16	DLC	This value is sent in the DLC field of the next transmit message. In addition, if RTR=0, this value controls the number of Data bytes sent in the next transmit message, from the CANTDA and CANTDB registers: 0000-0111 = 0-7 bytes 1xxx = 8 bytes	0	X
30	RTR	This value is sent in the RTR bit of the next transmit message. If this bit is 0, the number of data bytes called out by the DLC field are sent from the CANTDA and CANTDB registers. If it's 1, a Remote Frame is sent, containing a request for that number of bytes.	0	X
31	FF	If this bit is 0, the next transmit message will be sent with an 11-bit Identifier, while if it's 1, the message will be sent with a 29-bit Identifier.	0	X

Tx Identifier Register (CANTID1, 2, 3 - 0xE00x x034, 44, 54)

When the corresponding TBS bit in CANSR is 1, software can write to one of these registers to define the Identifier field of the next transmit message. Bits not listed read as 0 and should be written as 0. The register assumes two different formats depending on the FF bit in CANTFI. See Table 124 for details on specific CAN channel register address.

Table 139: CAN Tx Identifier Register when FF=0 (CANTID1, 2, 3 - 0xE00x x034, 44, 54)

CANTID	Name	Function	Reset Value	RM Set
10:0	ID	The 11-bit Identifier to be sent in the next transmit message.	0	X

Table 140: CAN Tx Identifier Register when FF=1 (CANTID1, 2, 3 - 0xE00x x034, 44, 54)

CANTID	Name	Function	Reset Value	RM Set
28:0	ID	The 29-bit Identifier to be sent in the next transmit message.	0	X

Tx Data Register A (CANTDA1, 2, 3 - 0xE00x x038, 48, 58)

When the corresponding TBS bit in CANSR is 1, software can write to one of these registers to define the first 1-4 Data bytes of the next transmit message. See Table 124 for details on specific CAN channel register address.

Table 141: CAN Tx Data Register A (CANTDA1, 2, 3 - 0xE00x x038, 48, 58)

CANTDA	Name	Function	Reset Value	RM Set
7:0	Data 1	If RTR=0 and DLC >= 0001 in the corresponding CANTFI, this byte is sent as the first Data byte of the next transmit message.	0	X
15:8	Data 2	If RTR=0 and DLC >= 0010 in the corresponding CANTFI, this byte is sent as the 2nd Data byte of the next transmit message.	0	X
23:16	Data 3	If RTR=0 and DLC >= 0011 in the corresponding CANTFI, this byte is sent as the 3rd Data byte of the next transmit message.	0	X
31:24	Data 4	If RTR=0 and DLC >= 0100 in the corresponding CANTFI, this byte is sent as the 4th Data byte of the next transmit message.	0	X

Tx Data Register B (CANTDB1, 2, 3 - 0xE00x x03C, 4C, 5C)

When the corresponding TBS bit in CANSR is 1, software can write to one of these registers to define the 5th through 8th Data bytes of the next transmit message. See Table 124 for details on specific CAN channel register address.

Table 142: CAN Tx Data Register B (CANTDB1, 2, 3 - 0xE00x x03C, 4C, 5C)

CANTDB	Name	Function	Reset Value	RM Set
7:0	Data 5	If RTR=0 and DLC >= 0101 in the corresponding CANTFI, this byte is sent as the 5th Data byte of the next transmit message.	0	X
15:8	Data 6	If RTR=0 and DLC >= 0110 in the corresponding CANTFI, this byte is sent as the 6th Data byte of the next transmit message.	0	X
23:16	Data 7	If RTR=0 and DLC >= 0111 in the corresponding CANTFI, this byte is sent as the 7th Data byte of the next transmit message.	0	X
31:24	Data 8	If RTR=0 and DLC >= 1000 in the corresponding CANTFI, this byte is sent as the 8th Data byte of the next transmit message.	0	X

CAN CONTROLLER OPERATION

Error Handling

The CAN Controllers count and handle transmit and receive errors as specified in CAN Spec 2.0B. The Transmit and Receive Error Counters are incremented for each detected error and are decremented when operation is error-free. If the Transmit Error counter contains 255 and another error occurs, the CAN Controller is forced into a state called Bus-Off. In this state, the following register bits are set: BS in CANSR, BEI and EI in CANIR if these are enabled, and RM in CANMOD. RM resets and disables much of the CAN Controller. Also at this time the Transmit Error Counter is set to 127 and the Receive Error Counter is cleared. Software must next clear the RM bit. Thereafter the Transmit Error Counter will count down 128 occurrences of the Bus Free condition (11 consecutive recessive bits). Software can monitor this countdown by reading the Tx Error Counter. When this countdown is complete, the CAN Controller clears BS and ES in CANSR, and sets EI in CANSR if EIE in IER is 1.

The Tx and Rx error counters can be written if RM in CANMOD is 1. Writing 255 to the Tx Error Counter forces the CAN Controller to Bus-Off state. If Bus-Off (BS in CANSR) is 1, writing any value 0 through 254 to the Tx Error Counter clears Bus-Off. When software clears RM in CANMOD thereafter, only one Bus Free condition (11 consecutive recessive bits) is needed before operation resumes.

Sleep Mode

The CAN Controller will enter sleep mode if the SM bit in the CAN Mode register is 1, no CAN interrupt is pending, and there is no activity on the CAN bus. Software can only set SM when RM in the CAN Mode register is 0; it can also set the WUIE bit in the CAN Interrupt Enable register to enable an interrupt on any wake-up condition.

The CAN Controller wakes up (and sets WUI in the CAN Interrupt register if WUIE in the CAN Interrupt Enable register is 1) in response to a) a dominant bit on the CAN bus, or b) software clearing SM in the CAN Mode register. A sleeping CAN Controller, that wakes up in response to bus activity, is not able to receive an initial message, until after it detects Bus_Free (11 consecutive recessive bits). If an interrupt is pending or the CAN bus is active when software sets SM, the wakeup is immediate.

Interrupts

Each CAN Controller produces 3 interrupt requests, Receive, Transmit, and “other status”. The Transmit interrupt is the OR of the Transmit interrupts from the three Tx Buffers. Each Receive and Transmit interrupt request from each controller is assigned its own channel in the Vectored Interrupt Controller (VIC), and can have its own interrupt service routine. The “other status” interrupts from all of the CAN controllers, and the Acceptance Filter LUTerr condition, are ORed into one VIC channel.

Transmit Priority

If the TPM bit in the CANMOD register is 0, multiple enabled Tx Buffers contend for the right to send their messages based on the value of their CAN Identifier (TID). If TPM is 1, they contend based on the PRIO fields in bits 7:0 of their CANTFS registers. In both cases the smallest binary value has priority. If two (or three) transmit-enabled buffers have the same smallest value, the lowest-numbered buffer sends first.

The CAN controller selects among multiple enabled Tx Buffers dynamically, just before it sends each message.

CENTRALIZED CAN REGISTERS

Three read-only registers group the bits in the Status registers of the CAN controllers for common accessibility. If devices with more or fewer CAN controllers are defined, the number of bits used in the active bytes will change correspondingly. Each defined byte of the following registers contains one particular status bit from each of the CAN controllers, in its LS bits.

Central Transmit Status Register (CANTxSR - 0xE004 0000)

Table 143: CAN Central Transmit Status Register (CANTxSR - 0xE004 0000)

CANTxSR	Name	Function	Reset Value
3:0	TS4:1	1: the CAN controller is sending a message (same as TS in the CANGSR) TS4:3 are available in LPC2294 only. In other parts these bits are reserved.	0
7:4	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
11:8	TBS4:1	1: all 3 Tx Buffers are available to the CPU (same as TBS in CANGSR) TBS4:3 are available in LPC2294 only. In other parts these bits are reserved.	all 1
15:12	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
19:16	TCS4:1	1: all requested transmissions have been completed successfully (same as TCS in CANGSR) TCS4:3 are available in LPC2294 only. In other parts these bits are reserved.	all 1
31:20	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Central Receive Status Register (CANRxSR - 0xE004 0004)

Table 144: CAN Central Receive Status Register (CANRxSR - 0xE004 0004)

CANRxSR	Name	Function	Reset Value
3:0	RS4:1	1: the CAN controller is receiving a message (same as RS in CANGSR) RS4:3 are available in LPC2294 only. In other parts these bits are reserved.	0
7:4	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
11:8	RBS4:1	1: a received message is available in the CAN controller (same as RBS in CANGSR) RBS4:3 are available in LPC2294 only. In other parts these bits are reserved.	0
15:12	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
19:16	DOS4:1	1: a message was lost because the preceding message to this CAN controller was not read out quickly enough (same as DOS in CANGSR) DOS4:3 are available in LPC2294 only. In other parts these bits are reserved.	0
31:20	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Central Miscellaneous Status Register (CANMSR - 0xE004 0008)

Table 145: CAN Central Miscellaneous Status Register (CANMSR - 0xE004 0008)

CANMSR	Name	Function	Reset Value
3:0	ES4:1	1: one or both of the Tx and Rx Error Counters has reached the limit set in the EWL register (same as ES in CANGSR) ES4:3 are available in LPC2294 only. In other parts these bits are reserved.	0
7:4	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
11:8	BS4:1	1: the CAN controller is currently involved in bus activities (same as BS in CANGSR) BS4:3 are available in LPC2294 only. In other parts these bits are reserved.	0
31:12	Reserved	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

GLOBAL ACCEPTANCE FILTER

This block provides lookup for received Identifiers (called Acceptance Filtering in CAN terminology) for all the CAN Controllers. It includes a 512 x 32 (2K byte) RAM in which software maintains one to five tables of Identifiers. This RAM can contain up to 1024 Standard Identifiers or 512 Extended Identifiers, or a mixture of both types.

If Standard (11-bit) Identifiers are used in the application, at least one of 3 tables in Acceptance Filter RAM must not be empty. If the optional “fullCAN mode” is enabled, the first table contains Standard identifiers for which reception is to be handled in this mode. The next table contains individual Standard Identifiers and the third contains ranges of Standard Identifiers, for which messages are to be received via the CAN Controllers. The tables of fullCAN and individual Standard Identifiers must be arranged in ascending numerical order, one per halfword, two per word. Since each CAN bus has its own address map, each entry also contains the number of the CAN Controller (001-110) to which it applies.

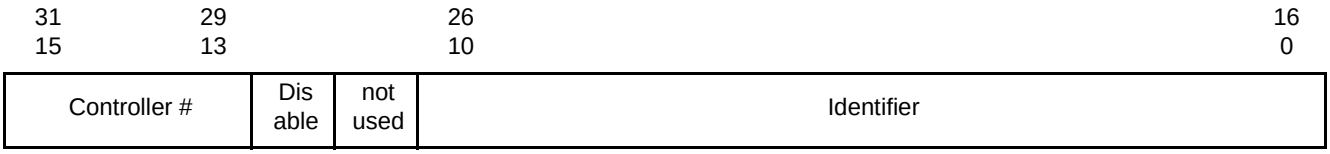


Figure 35: Entry in fullCAN and Individual Standard Identifier Tables

The table of Standard Identifier Ranges contains paired upper and lower (inclusive) bounds, one pair per word. These must also be arranged in ascending numerical order.

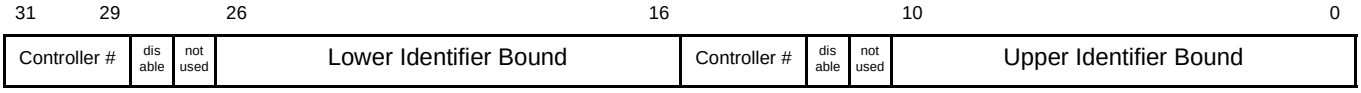


Figure 36: Entry in Standard Identifier Range Table

The disable bits in Standard entries provide a means to turn response, to particular CAN Identifiers or ranges of Identifiers, on and off dynamically. When the Acceptance Filter function is enabled, only the disable bits in Acceptance Filter RAM can be changed by software. Response to a range of Standard addresses can be enabled by writing 32 zero bits to its word in RAM, and turned off by writing 32 one bits (0xFFFF FFFF) to its word in RAM. Only the disable bits are actually changed. Disabled entries must maintain the ascending sequence of Identifiers.

If Extended (29-bit) Identifiers are used in the application, at least one of the other two tables in Acceptance Filter RAM must not be empty, one for individual Extended Identifiers and one for ranges of Extended Identifiers. The table of individual Extended Identifiers must be arranged in ascending numerical order.

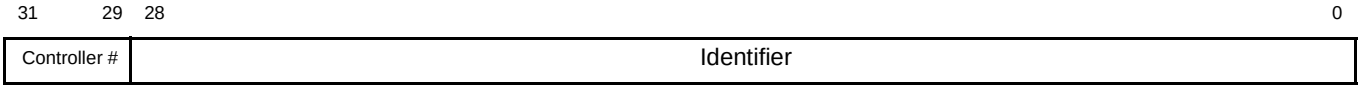


Figure 37: Entry in either Extended Identifier Table

The table of ranges of Extended Identifiers must contain an even number of entries, of the same form as in the individual Extended Identifier table. Like the Individual Extended table, the Extended Range must be arranged in ascending numerical order. The first and second (3rd and 4th …) entries in the table are implicitly paired as an inclusive range of Extended addresses, such that any received address that falls in the inclusive range is received (accepted). Software must maintain the table to consist of such word pairs.

There is no facility to receive messages to Extended identifiers using the fullCAN method.

Five address registers point to the boundaries between the tables in Acceptance Filter RAM: fullCAN Standard addresses, Standard Individual addresses, Standard address ranges, Extended Individual addresses, and Extended address ranges. These tables must be consecutive in memory. The start of each of the latter four tables is implicitly the end of the preceding table. The end of the Extended range table is given in an End of Tables register. If the start address of a table equals the start of the next table or the End Of Tables register, that table is empty.

When the Receive side of a CAN controller has received a complete Identifier, it signals the Acceptance Filter of this fact. The Acceptance Filter responds to this signal, and reads the Controller number, the size of the Identifier, and the Identifier itself from the Controller. It then proceeds to search its RAM to determine whether the message should be received or ignored.

If fullCAN mode is enabled and the CAN controller signals that the current message contains a Standard identifier, the Acceptance Filter first searches the table of identifiers for which reception is to be done in fullCAN mode. Otherwise, or if the AF doesn't find a match in the fullCAN table, it searches its individual Identifier table for the size of Identifier signalled by the CAN controller. If it finds an equal match, the AF signals the CAN controller to retain the message, and provides it with an ID Index value to store in its the Receive Frame Status register register.

If the Acceptance Filter does not find a match in the appropriate individual Identifier table, it then searches the Identifier Range table for the size of Identifier signalled by the CAN controller. If the AF finds a match to a range in the table, it similarly signals the CAN controller to retain the message, and provides it with an ID Index value to store in its the Receive Frame Status register register. If the Acceptance Filter does not find a match in either the individual or Range table for the size of Identifier received, it signals the CAN controller to discard/ignore the received message.

ACCEPTANCE FILTER REGISTERS

Acceptance Filter Mode Register (AFMR - 0xE003 C000)

Table 146: Acceptance Filter Modes Register (AFMR - 0xE003 C000)

AFMR	Name	Function	Reset Value
0	AccOff	1: if AccBP is 0, the Acceptance Filter is not operational. All Rx messages on all CAN buses are ignored.	1
1	AccBP	1: all Rx messages are accepted on enabled CAN controllers. Software must set this bit before modifying the contents of any of the registers described below, and before modifying the contents of Lookup Table RAM in any way other than setting or clearing Disable bits in Standard Identifier entries. When both this bit and AccOff are 0, the Acceptance filter operates to screen received CAN Identifiers.	0
2	eFCAN	1: the Acceptance Filter itself will take care of receiving and storing messages for selected Standard ID values on selected CAN buses. See “FullCAN Mode“ on page 210. 0: software must read all messages for all enabled IDs on all enabled CAN buses, from the receiving CAN controllers.	0

Standard Frame Individual Start Address Register (SFF_sa - 0xE003 C004)

Table 147: Standard Frame Start Address Register (SFF_sa - 0xE003 C004)

SFF_sa	Name	Function	Reset Value
10:2		The start address of the table of individual Standard Identifiers in AF Lookup RAM. If the table is empty, write the same value in this register and the SFF_GRP_sa register described below. For compatibility with possible future devices, please write zeroes in bits 31:11 and 1:0 of this register. If the eFCAN bit in the AFMR is 1, this value also indicates the size of the table of Standard IDs which the Acceptance Filter will search and (if found) automatically store received messages in Acceptance Filter RAM.	0

Standard Frame Group Start Address Register (SFF_GRP_sa - 0xE003 C008)

Table 148: Standard Frame Group Start Address Reg (SFF_GRP_sa - 0xE003 C008)

SFF_GRP_sa	Name	Function	Reset Value
11:2		The start address of the table of grouped Standard Identifiers in AF Lookup RAM. If the table is empty, write the same value in this register and the EFF_sa register described below. The largest value that should be written to this register is 0x800, when only the Standard Individual table is used, and the last word (address 0x7FC) in AF Lookup Table RAM is used. For compatibility with possible future devices, please write zeroes in bits 31:12 and 1:0 of this register.	0

Extended Frame Start Address Register (EFF_sa - 0xE003 C00C)

Table 149: Extended Frame Start Address Register (EFF_sa - 0xE003 C00C)

EFF_sa	Name	Function	Reset Value
10:2		The start address of the table of individual Extended Identifiers in AF Lookup RAM. If the table is empty, write the same value in this register and the EFF_GRP_sa register described below. The largest value that should be written to this register is 0x800, when both Extended Tables are empty and the last word (address 0x7FC) in AF Lookup Table RAM is used. For compatibility with possible future devices, please write zeroes in bits 31:11 and 1:0 of this register.	0

Extended Frame Group Start Address Register (EFF_GRP_sa - 0xE003 C010)

Table 150: Extended Frame Group Start Addr Register (EFF_GRP_sa - 0xE003 C010)

EFF_GRP_sa	Name	Function	Reset Value
11:2		The start address of the table of grouped Extended Identifiers in AF Lookup RAM. If the table is empty, write the same value in this register and the ENDofTable register described below. The largest value that should be written to this register is 0x800, when this table is empty and the last word (address 0x7FC) in AF Lookup Table RAM is used. For compatibility with possible future devices, please write zeroes in bits 31:12 and 1:0 of this register.	0

End of AF Tables Register (ENDofTable - 0xE003 C014)

Table 151: End of AF Tables Register (ENDofTable - 0xE003 C014)

ENDofTable	Name	Function	Reset Value
11:2		The address above the last active address in the last active AF table. For compatibility with possible future devices, please write zeroes in bits 31:12 and 1:0 of this register. If the eFCAN bit in the AFMR is 0, the largest value that should be written to this register is 0x800, which allows the last word (address 0x7FC) in AF Lookup Table RAM to be used. If the eFCAN bit in the AFMR is 1, this value marks the start of the area of Acceptance Filter RAM, into which the Acceptance Filter will automatically receive messages for selected IDs on selected CAN buses. In this case, the maximum value that should be written to this register is 0x800 minus 6 times the value in SFF_sa. This allows 12 bytes of message storage between this address and the end of Acceptance Filter RAM, for each Standard ID that is specified between the start of Acceptance Filter RAM, and the next active AF table.	0

LUT Error Address Register (LUTerrAd - 0xE003 C018)

Table 152: LUT Error Address Register (LUTerrAd - 0xE003 C018)

LUTerrAd	Name	Function	Reset Value
10:2		It the LUT Error bit (below) is 1, this read-only field contains the address in AF Lookup Table RAM, at which the Acceptance Filter encountered an error in the content of the tables.	0

LUT Error Register (LUTerr - 0xE003 C01C)

Table 153: LUT Error Register (LUTerr - 0xE003 C01C)

LUTerr	Name	Function	Reset Value
0		This read-only bit is set to 1 if the Acceptance Filter encounters an error in the content of the tables in AF RAM. It is cleared when software reads the LUTerrAd register. This condition is ORed with the “other CAN” interrupts from the CAN controllers, to produce the request for a VIC interrupt channel.	0

EXAMPLES OF ACCEPTANCE FILTER TABLES AND ID INDEX VALUES

Suppose that the five Acceptance Filter address registers contain the values shown in the third column below. In this case each table contains the decimal number of words and entries shown in the next two columns, and the ID Index field of the CANRFS register can return the decimal values shown in the rightmost column, for CAN messages whose Identifiers match the entries in that table.

Table 154: Example of Acceptance Filter Tables and ID Index Values

Table	Register	Value	# words	# entries	ID Indexes
Standard Individual	SFF_sa	0x040	8_{10}	16_{10}	$0\text{-}15_{10}$
Standard Group	SFF_GRP_sa	0x060	4_{10}	4_{10}	$16\text{-}19_{10}$
Extended Individual	EFF_sa	0x070	36_{10}	36_{10}	$20\text{-}55_{10}$
Extended Group	EFF_GRP_sa	0x100	4_{10}	2_{10}	$56\text{-}57_{10}$
	ENDofTable	0x110			

Figure 38 below is a more detailed and graphic example of the address registers, table layout, and ID Index values. It shows:

- a Standard Individual table starting at the start of Acceptance Filter RAM and containing 26 Identifiers, followed by
- a Standard Group table containing 12 ranges of Identifiers, followed by
- an Extended Individual table containing 3 Identifiers, followed by
- an Extended Group table containing 2 ranges of Identifiers.

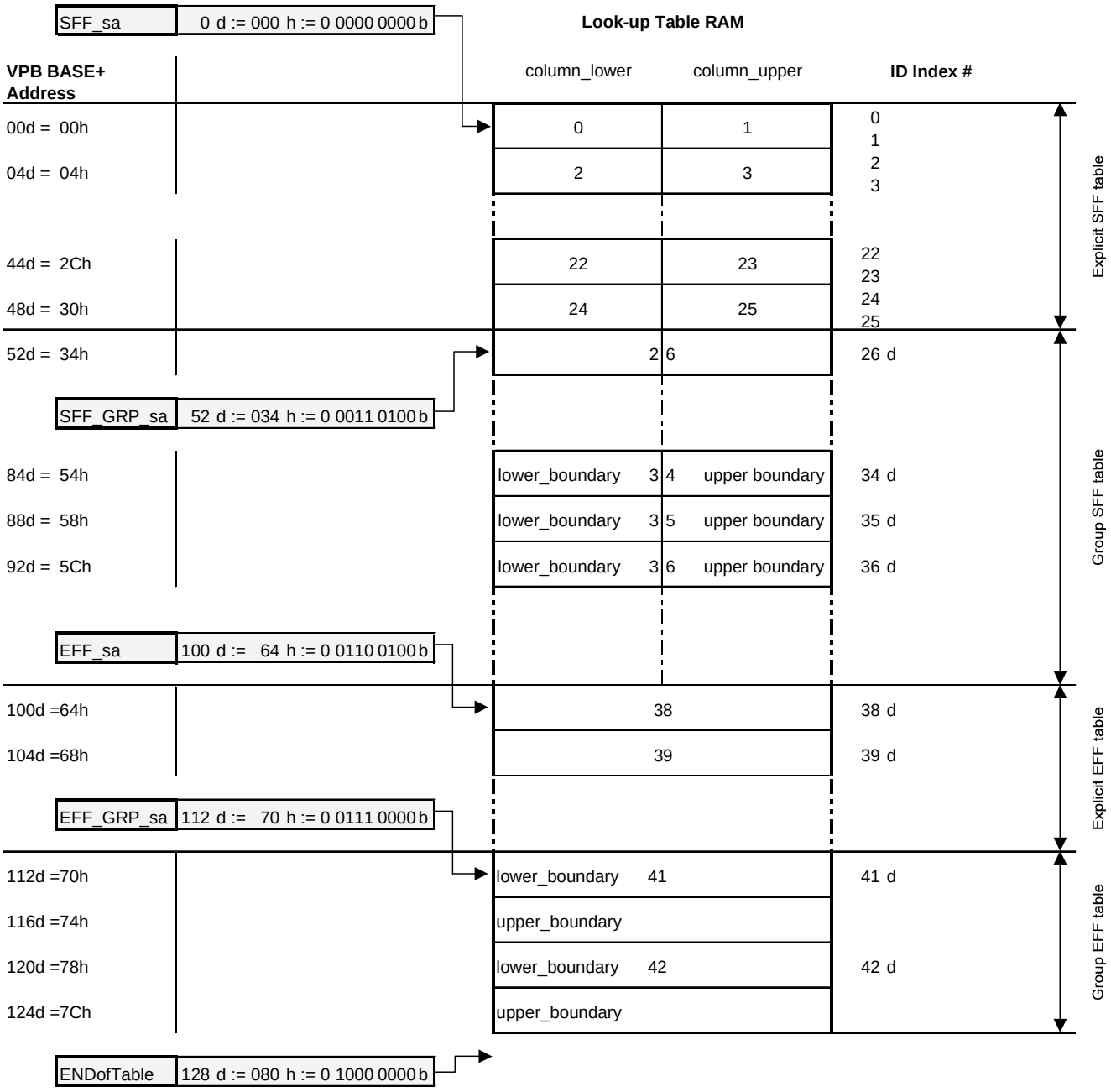


Figure 38: Detailed Example of Acceptance Filter Tables and ID Index Values

FULLCAN MODE

When fullCAN mode is enabled, the Acceptance Filter itself takes care of receiving and storing messages for selected Standard ID values on selected CAN buses, in the style of “FullCAN” controllers.

In order to set this bit and use this mode, two other conditions must be met with respect to the contents of Acceptance Filter RAM and the pointers into it:

1. The Standard Frame Individual Start Address Register (SFF_sa) must be greater than or equal to the number of IDs for which automatic receive storage is to be done, times two. SFF_sa must be rounded up to a multiple of 4 if necessary.
2. The EndOfTable register must be less than or equal to 0x800 minus 6 times the SFF_sa value, to allow 12 bytes of message storage for each ID for which automatic receive storage will be done.

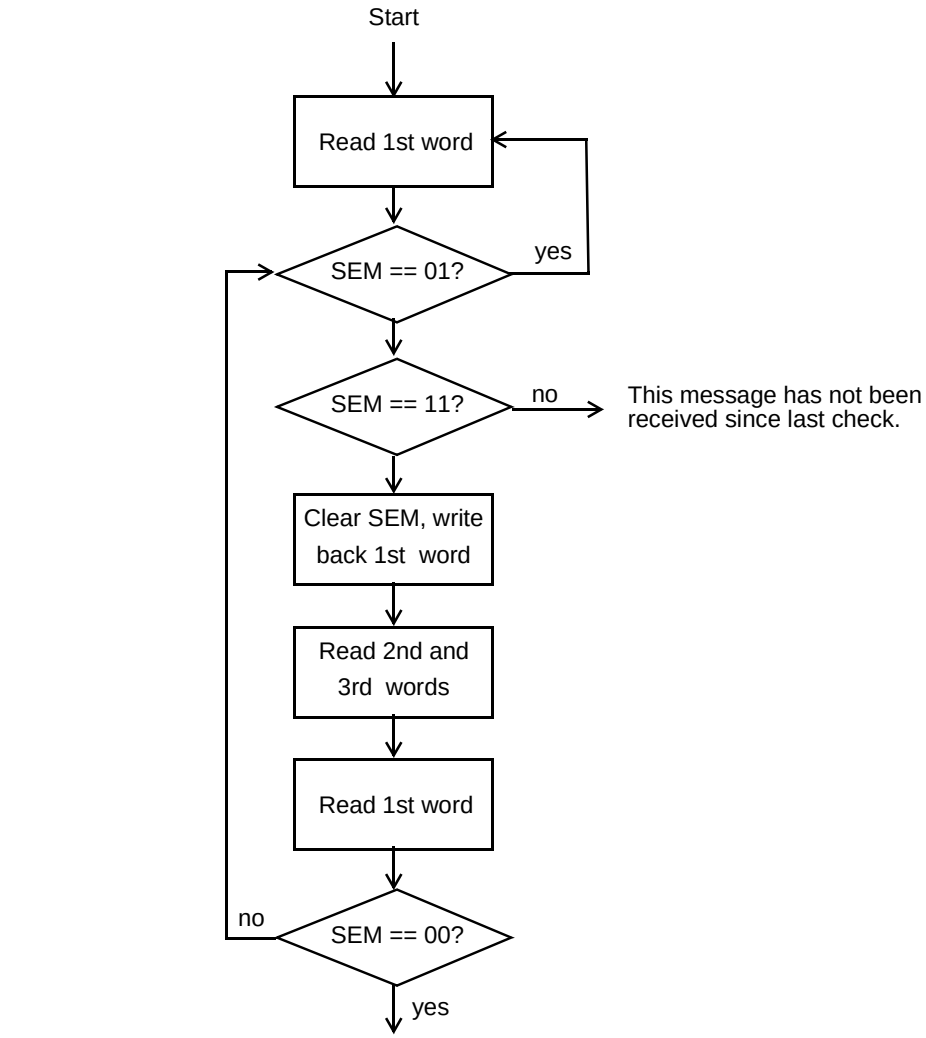
When these conditions are met and eFCAN is set:

- The area between the start of Acceptance Filter RAM and the SFF_sa address, is used for a table of individual Standard IDs and CAN Controller/bus identification, sorted in ascending order and in the same format as in the Individual Standard ID table (see Figure 35 on page 205). Entries can be marked as “disabled” as in the other Standard tables. If there are an odd number of “FullCAN” ID’s, at least one entry in this table must be so marked.
- The first (SFF_sa)/2 IDindex values are assigned to these automatically-stored ID’s. That is, IDindex values stored in the Rx Frame Status Register, for IDs not handled in this way, are increased by (SFF_sa)/2 compared to the values they would have when eFCAN is 0.
- When a Standard ID is received, the Acceptance Filter searches this table before the Standard Individual and Group tables.
- When a message is received for a controller and ID in this table, the Acceptance filter reads the received message out of the CAN controller and stores it in Acceptance Filter RAM, starting at (EndOfTable) + its IDindex*12.
- The format of such messages is shown in Table 155.

Table 155: Format of Automatically Stored Rx Message

Address	31							24	23							16	15					10		8	7							0
0	FF	RTR	0000				SEM		0000				DLC				00000					ID										
+4	Rx Data 4								Rx Data 3								Rx Data 2								Rx Data 1							
+8	Rx Data 8								Rx Data 7								Rx Data 6								Rx Data 5							

The FF, RTR, and DLC fields are as described in Table 133, “CAN Rx Frame Status Register (CANRFS - 0xE00x x020),” on page 199. Hardware sets the SEM field to 01 when it begins to update a message, and to 11 when it finishes doing so. Software should clear SEM to 00 as part of accessing a message. Software must access the three words in a message in a particular way to ensure that they are all from the same received message. Figure below shows how software should use the SEM field to ensure this.



Semaphore Procedure for Reading an Auto-Stored Message

15. TIMER0 AND TIMER1

Timer0 and Timer1 are functionally identical except for the peripheral base address.

FEATURES

- A 32-bit Timer/Counter with a programmable 32-bit Prescaler.
- Up to four 32-bit capture channels per timer, that can take a snapshot of the timer value when an input signal transitions. A capture event may also optionally generate an interrupt.
- Four 32-bit match registers that allow:
  - Continuous operation with optional interrupt generation on match.
  - Stop timer on match with optional interrupt generation.
  - Reset timer on match with optional interrupt generation.
- Up to four external outputs corresponding to match registers, with the following capabilities:
  - Set low on match.
  - Set high on match.
  - Toggle on match.
  - Do nothing on match.

APPLICATIONS

- Interval Timer for counting internal events.
- Pulse Width Demodulator via Capture inputs.
- Free running timer.

DESCRIPTION

The Timer is designed to count cycles of the peripheral clock (pclk) and optionally generate interrupts or perform other actions at specified timer values, based on four match registers. It also includes four capture inputs to trap the timer value when an input signal transitions, optionally generating an interrupt.

PIN DESCRIPTION

Table 156 gives a brief summary of each of the Timer related pins.

Table 156: Pin summary

Pin name	Pin direction	Pin Description
CAP0.3..0 CAP1.3..0	Input	**Capture Signals**- A transition on a capture pin can be configured to load one of the Capture Registers with the value in the Timer Counter and optionally generate an interrupt. Capture functionality can be selected from a number of pins. Also, if for example 2 pins are selected to provide CAP0.2 function in parallel, their inputs will be logically ored and this value will be processed as a single input. CAP0.0 can be selected from/on up to 3 pins at the same time. CAP0.1 can be selected from/on up to 2 pins at the same time. CAP0.2 can be selected from/on up to 3 pins at the same time. CAP0.3 can be selected from/on 1 pin. CAP1.0 can be selected from/on 1 pin. CAP1.1 can be selected from/on 1 pin. CAP1.2 can be selected from/on up to 2 pins at the same time. CAP1.3 can be selected from/on up to 2 pins at the same time.
MAT0.3...0 MAT1.0...0	Output	**External Match Output 0/1-** When a match register 0/1 (MR3:0) equals the timer counter (TC) this output can either toggle, go low, go high, or do nothing. The External Match Register (EMR) controls the functionality of this output. Match Output functionality can be selected on a number of pins in parallel. It is also possible for example, to have 2 pins selected at the same time so that they provide MAT1.3 function in parallel. MAT0.0 can be selected on up to 2 pins at the same time. MAT0.1 can be selected on up to 2 pins at the same time. MAT0.2 can be selected on up to 2 pins at the same time. MAT0.3 can be selected on 1 pin. MAT1.0 can be selected on 1 pin. MAT1.1 can be selected on 1 pin. MAT1.2 can be selected on up to 2 pins at the same time. MAT1.3 can be selected on up to 2 pins at the same time.

REGISTER DESCRIPTION

Each Timer contains the registers shown in Table 157. More detailed descriptions follow.

Table 157: TIMER0 and TIMER1 Register Map

Generic Name	Description	Access	Reset Value*	TIMER0 Address & Name	TIMER1 Address & Name
IR	Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of eight possible interrupt sources are pending.	R/W	0	0xE0004000 T0IR	0xE0008000 T1IR
TCR	Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.	R/W	0	0xE0004004 T0TCR	0xE0008004 T1TCR
TC	Timer Counter. The 32-bit TC is incremented every PR+1 cycles of pclk. The TC is controlled through the TCR.	RW	0	0xE0004008 T0TC	0xE0008008 T1TC
PR	Prescale Register. The TC is incremented every PR+1 cycles of pclk.	R/W	0	0xE000400C T0PR	0xE000800C T1PR
PC	Prescale Counter. The 32-bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented.	R/W	0	0xE0004010 T0PC	0xE0008010 T1PC
MCR	Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.	R/W	0	0xE0004014 T0MCR	0xE0008014 T1MCR
MR0	Match Register 0. MR0 can be enabled through the MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt every time MR0 matches the TC.	R/W	0	0xE0004018 T0MR0	0xE0008018 T1MR0
MR1	Match Register 1. See MR0 description.	R/W	0	0xE000401C T0MR1	0xE000801C T1MR1
MR2	Match Register 2. See MR0 description.	R/W	0	0xE0004020 T0MR2	0xE0008020 T1MR2
MR3	Match Register 3. See MR0 description.	R/W	0	0xE0004024 T0MR3	0xE0008024 T1MR3
CCR	Capture Control Register. The CCR controls which edges of the capture inputs are used to load the Capture Registers and whether or not an interrupt is generated when a capture takes place.	R/W	0	0xE0004028 T0CCR	0xE0008028 T1CCR
CR0	Capture Register 0. CR0 is loaded with the value of TC when there is an event on the CAP0.0(CAP1.0) input.	RO	0	0xE000402C T0CR0	0xE000802C T1CR0
CR1	Capture Register 1. See CR0 description.	RO	0	0xE0004030 T0CR1	0xE0008030 T1CR1
CR2	Capture Register 2. See CR0 description.	RO	0	0xE0004034 T0CR2	0xE0008034 T1CR2
CR3	Capture Register 3. See CR0 description.	RO	0	0xE0004038 T0CR3	0xE0008038 T1CR3
EMR	External Match Register. The EMR controls the external match pins MAT0.0-3 (MAT1.0-3).	R/W	0	0xE000403C T0EMR	0xE000803C T1EMR

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

Interrupt Register (IR: TIMER0 - T0IR: 0xE0004000; TIMER1 - T1IR: 0xE0008000)

The Interrupt Register consists of four bits for the match interrupts and four bits for the capture interrupts. If an interrupt is generated then the corresponding bit in the IR will be high. Otherwise, the bit will be low. Writing a logic one to the corresponding IR bit will reset the interrupt. Writing a zero has no effect.

Table 158: Interrupt Register (IR: TIMER0 - T0IR: 0xE0004000; TIMER1 - T1IR: 0xE0008000)

IR	Function	Description	Reset Value
0	MR0 Interrupt	Interrupt flag for match channel 0.	0
1	MR1 Interrupt	Interrupt flag for match channel 1.	0
2	MR2 Interrupt	Interrupt flag for match channel 2.	0
3	MR3 Interrupt	Interrupt flag for match channel 3.	0
4	CR0 Interrupt	Interrupt flag for capture channel 0 event.	0
5	CR1 Interrupt	Interrupt flag for capture channel 1 event.	0
6	CR2 Interrupt	Interrupt flag for capture channel 2 event.	0
7	CR3 Interrupt	Interrupt flag for capture channel 3 event.	0

Timer Control Register (TCR: TIMER0 - T0TCR: 0xE0004004; TIMER1 - T1TCR: 0xE0008004)

The Timer Control Register (TCR) is used to control the operation of the Timer Counter.

Table 159: Timer Control Register (TCR: TIMER0 - T0TCR: 0xE0004004; TIMER1 - T1TCR: 0xE0008004)

TCR	Function	Description	Reset Value
0	Counter Enable	When one, the Timer Counter and Prescale Counter are enabled for counting. When zero, the counters are disabled.	0
1	Counter Reset	When one, the Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of pclk. The counters remain reset until TCR[1] is returned to zero.	0

Timer Counter (TC: TIMER0 - T0TC: 0xE0004008; TIMER1 - T1TC: 0xE0008008)

The 32-bit Timer Counter is incremented when the Prescale Counter reaches its terminal count. Unless it is reset before reaching its upper limit, the TC will count up through the value 0xFFFFFFFF and then wrap back to the value 0x00000000. This event does not cause an interrupt, but a Match register can be used to detect an overflow if needed.

Prescale Register (PR: TIMER0 - T0PR: 0xE000400C; TIMER1 - T1PR: 0xE000800C)

The 32-bit Prescale Register specifies the maximum value for the Prescale Counter.

Prescale Counter Register (PC: TIMER0 - T0PC: 0xE0004010; TIMER1 - T1PC: 0xE0008010)

The 32-bit Prescale Counter controls division of pclk by some constant value before it is applied to the Timer Counter. This allows control of the relationship of the resolution of the timer versus the maximum time before the timer overflows. The Prescale Counter is incremented on every pclk. When it reaches the value stored in the Prescale Register, the Timer Counter is incremented and the Prescale Counter is reset on the next pclk. This causes the TC to increment on every pclk when PR = 0, every 2 pclks when PR = 1, etc.

Match Registers (MR0 - MR3)

The Match register values are continuously compared to the Timer Counter value. When the two values are equal, actions can be triggered automatically. The action possibilities are to generate an interrupt, reset the Timer Counter, or stop the timer. Actions are controlled by the settings in the MCR register.

Match Control Register (MCR: TIMER0 - T0MCR: 0xE0004014; TIMER1 - T1MCR: 0xE0008014)

The Match Control Register is used to control what operations are performed when one of the Match Registers matches the Timer Counter. The function of each of the bits is shown in Table 160.

Table 160: Match Control Register (MCR: TIMER0 - T0MCR: 0xE0004014; TIMER1 - T1MCR: 0xE0008014)

MCR	Function	Description	Reset Value
0	Interrupt on MR0	When one, an interrupt is generated when MR0 matches the value in the TC. When zero this interrupt is disabled.	0
1	Reset on MR0	When one, the TC will be reset if MR0 matches it. When zero this feature is disabled.	0
2	Stop on MR0	When one, the TC and PC will be stopped and TCR[0] will be set to 0 if MR0 matches the TC. When zero this feature is disabled.	0
3	Interrupt on MR1	When one, an interrupt is generated when MR1 matches the value in the TC. When zero this interrupt is disabled.	0
4	Reset on MR1	When one, the TC will be reset if MR1 matches it. When zero this feature is disabled.	0
5	Stop on MR1	When one, the TC and PC will be stopped and TCR[0] will be set to 0 if MR1 matches the TC. When zero this feature is disabled.	0
6	Interrupt on MR2	When one, an interrupt is generated when MR2 matches the value in the TC. When zero this interrupt is disabled.	0
7	Reset on MR2	When one, the TC will be reset if MR2 matches it. When zero this feature is disabled.	0
8	Stop on MR2	When one, the TC and PC will be stopped and TCR[0] will be set to 0 if MR2 matches the TC. When zero this feature is disabled.	0
9	Interrupt on MR3	When one, an interrupt is generated when MR3 matches the value in the TC. When zero this interrupt is disabled.	0
10	Reset on MR3	When one, the TC will be reset if MR3 matches it. When zero this feature is disabled.	0
11	Stop on MR3	When one, the TC and PC will be stopped and TCR[0] will be set to 0 if MR3 matches the TC. When zero this feature is disabled.	0

Capture Registers (CR0 - CR3)

Each Capture register is associated with a device pin and may be loaded with the Timer Counter value when a specified event occurs on that pin. The settings in the Capture Control Register register determine whether the capture function is enabled, and whether a capture event happens on the rising edge of the associated pin, the falling edge, or on both edges.

Capture Control Register (CCR: TIMER0 - T0CCR: 0xE0004028; TIMER1 - T1CCR: 0xE0008028)

The Capture Control Register is used to control whether one of the four Capture Registers is loaded with the value in the Timer Counter when the capture event occurs, and whether an interrupt is generated by the capture event. Setting both the rising and falling bits at the same time is a valid configuration, resulting in a capture event for both edges. In the description below, "n" represents the Timer number, 0 or 1.

Table 161: Capture Control Register (CCR: TIMER0 - T0CCR: 0xE0004028; TIMER1 - T1CCR: 0xE0008028)

CCR	Function	Description	Reset Value
0	Capture on CAPn.0 rising edge	When one, a sequence of 0 then 1 on CAPn.0 will cause CR0 to be loaded with the contents of the TC. When zero, this feature is disabled.	0
1	Capture on CAPn.0 falling edge	When one, a sequence of 1 then 0 on CAPn.0 will cause CR0 to be loaded with the contents of TC. When zero, this feature is disabled.	0
2	Interrupt on CAPn.0 event	When one, a CR0 load due to a CAPn.0 event will generate an interrupt. When zero, this feature is disabled.	0
3	Capture on CAPn.1 rising edge	When one, a sequence of 0 then 1 on CAPn.1 will cause CR1 to be loaded with the contents of the TC. When zero, this feature is disabled.	0
4	Capture on CAPn.1 falling edge	When one, a sequence of 1 then 0 on CAPn.1 will cause CR1 to be loaded with the contents of TC. When zero, this feature is disabled.	0
5	Interrupt on CAPn.1 event	When one, a CR1 load due to a CAPn.1 event will generate an interrupt. When zero, this feature is disabled.	0
6	Capture on CAPn.2 rising edge	When one, a sequence of 0 then 1 on CAPn.2 will cause CR2 to be loaded with the contents of the TC. When zero, this feature is disabled.	0
7	Capture on CAPn.2 falling edge	When one, a sequence of 1 then 0 on CAPn.2 will cause CR2 to be loaded with the contents of TC. When zero, this feature is disabled.	0
8	Interrupt on CAPn.2 event	When one, a CR2 load due to a CAPn.2 event will generate an interrupt. When zero, this feature is disabled.	0
9	Capture on CAPn.3 rising edge	When one, a sequence of 0 then 1 on CAPn.3 will cause CR3 to be loaded with the contents of TC. When zero, this feature is disabled.	0
10	Capture on CAPn.3 falling edge	When one, a sequence of 1 then 0 on CAPn.3 will cause CR3 to be loaded with the contents of TC. When zero, this feature is disabled.	0
11	Interrupt on CAPn.3 event	When one, a CR3 load due to a CAPn.3 event will generate an interrupt. When zero, this feature is disabled.	0

External Match Register (EMR: TIMER0 - T0EMR: 0xE000403C; TIMER1 - T1EMR: 0xE000803C)

The External Match Register provides both control and status of the external match pins M(0-3).

Table 162: External Match Register (EMR: TIMER0 - T0EMR: 0xE000403C; TIMER1 - T1EMR: 0xE000803C)

EMR	Function	Description	Reset Value
0	External Match 0	This bit reflects the state of output MAT0.0/MAT1.0, whether or not this output is connected to its pin. When a match occurs for MR0, this output of the timer can either toggle, go low, go high, or do nothing. Bits EMR[4:5] control the functionality of this output.	0
1	External Match 1	This bit reflects the state of output MAT0.1/MAT1.1, whether or not this output is connected to its pin. When a match occurs for MR1, this output of the timer can either toggle, go low, go high, or do nothing. Bits EMR[6:7] control the functionality of this output.	0
2	External Match 2	This bit reflects the state of output MAT0.2/MAT1.2, whether or not this output is connected to its pin. When a match occurs for MR2, this output of the timer can either toggle, go low, go high, or do nothing. Bits EMR[8:9] control the functionality of this output.	0
3	External Match 3	This bit reflects the state of output MAT0.3/MAT1.3, whether or not this output is connected to its pin. When a match occurs for MR3, this output of the timer can either toggle, go low, go high, or do nothing. Bits EMR[10:11] control the functionality of this output.	0
5:4	External Match Control 0	Determines the functionality of External Match 0. Table 163 shows the encoding of these bits.	0
7:6	External Match Control 1	Determines the functionality of External Match 1. Table 163 shows the encoding of these bits.	0
9:8	External Match Control 2	Determines the functionality of External Match 2. Table 163 shows the encoding of these bits.	0
11:10	External Match Control 3	Determines the functionality of External Match 3. Table 163 shows the encoding of these bits.	0

Table 163: External Match Control

EMR[11:10], EMR[9:8], EMR[7:6], or EMR[5:4]	Function
0 0	Do Nothing
0 1	Clear corresponding External Match output to 0 (LOW if pinned out)
1 0	Set corresponding External Match output to 1(HIGH if pinned out)
1 1	Toggle corresponding External Match output

EXAMPLE TIMER OPERATION

Figure 39 shows a timer configured to reset the count and generate an interrupt on match. The prescaler is set to 2 and the match register set to 6. At the end of the timer cycle where the match occurs, the timer count is reset. This gives a full length cycle to the match value. The interrupt indicating that a match occurred is generated in the next clock after the timer reached the match value.

Figure 40 shows a timer configured to stop and generate an interrupt on match. The prescaler is again set to 2 and the match register set to 6. In the next clock after the timer reaches the match value, the timer enable bit in TCR is cleared, and the interrupt indicating that a match occurred is generated.

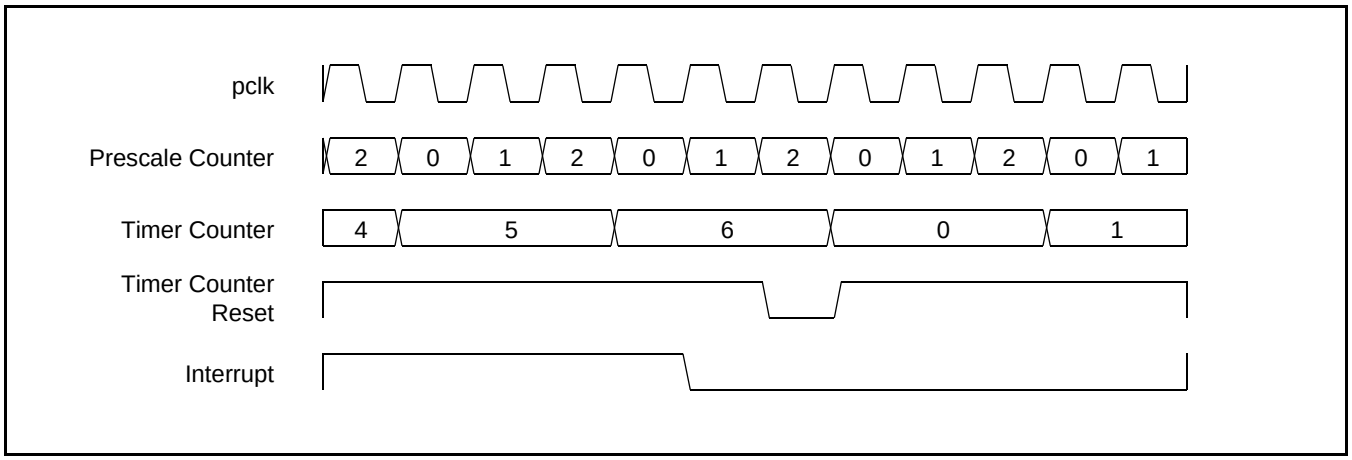


Figure 39: A timer cycle in which PR=2, MRx=6, and both interrupt and reset on match are enabled.

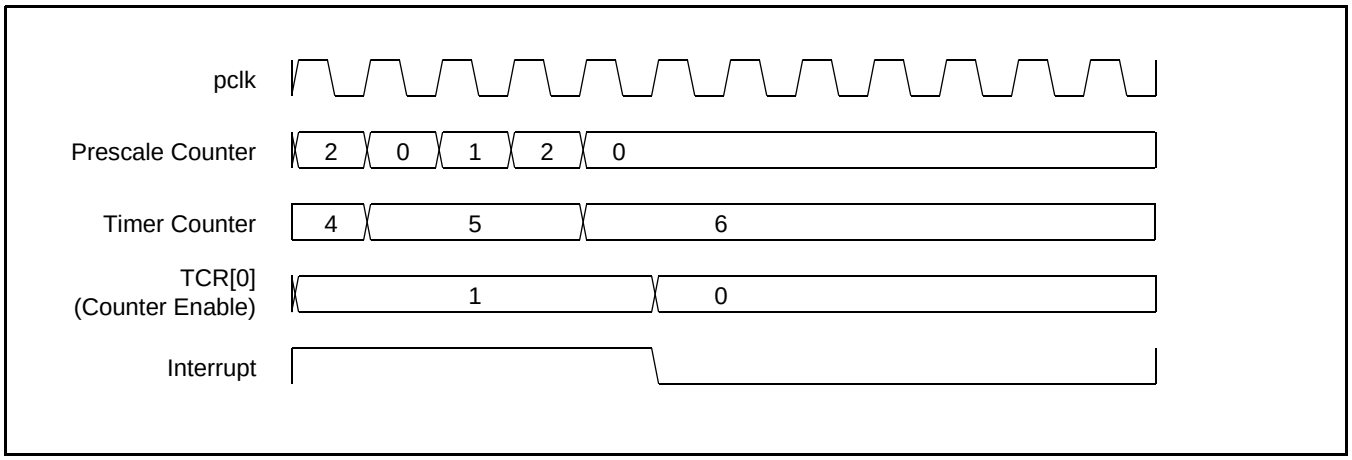


Figure 40: A timer cycle in which PR=2, MRx=6, and both interrupt and stop on match are enabled.

ARCHITECTURE

The block diagram for TIMER0 and TIMER1 is shown in Figure 41.

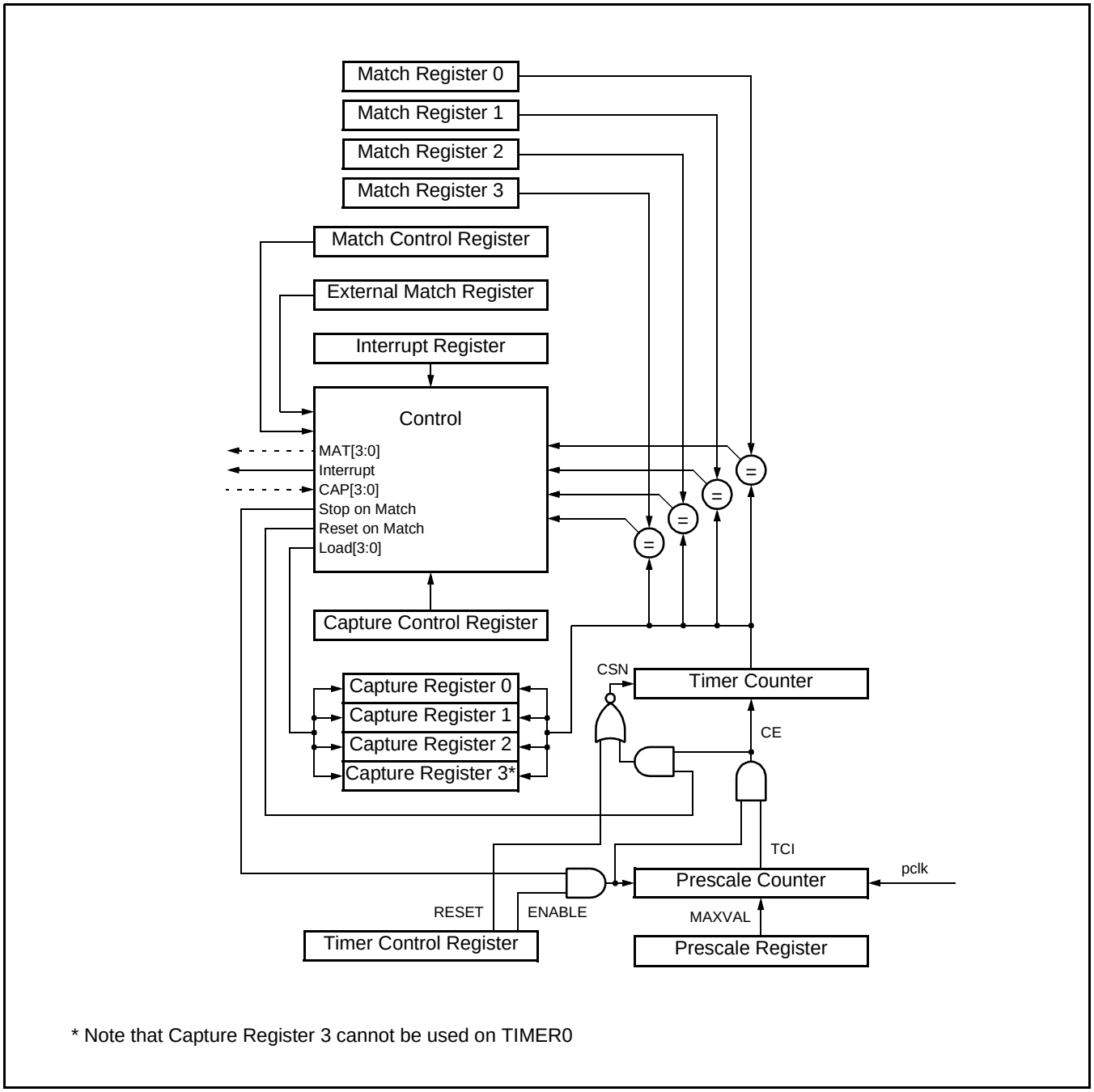


Figure 41: Timer block diagram

16. PULSE WIDTH MODULATOR (PWM)

LPC2119/2129/2194/2292/2294 Pulse Width Modulator is based on standard Timer 0/1 described in previous chapter. Application can choose among PWM and match functions available .

FEATURES

- Seven match registers allow up to 6 single edge controlled or 3 double edge controlled PWM outputs, or a mix of both types. The match registers also allow:
  - Continuous operation with optional interrupt generation on match.
  - Stop timer on match with optional interrupt generation.
  - Reset timer on match with optional interrupt generation.
- An external output for each match register with the following capabilities:
  - Set low on match.
  - Set high on match.
  - Toggle on match.
  - Do nothing on match.
- Supports single edge controlled and/or double edge controlled PWM outputs. Single edge controlled PWM outputs all go high at the beginning of each cycle unless the output is a constant low. Double edge controlled PWM outputs can have either edge occur at any position within a cycle. This allows for both positive going and negative going pulses.
- Pulse period and width can be any number of timer counts. This allows complete flexibility in the trade-off between resolution and repetition rate. All PWM outputs will occur at the same repetition rate.
- Double edge controlled PWM outputs can be programmed to be either positive going or negative going pulses.
- Match register updates are synchronized with pulse outputs to prevent generation of erroneous pulses. Software must "release" new match values before they can become effective.
- May be used as a standard timer if the PWM mode is not enabled.
- A 32-bit Timer/Counter with a programmable 32-bit Prescaler.
- Four 32-bit capture channels take a snapshot of the timer value when an input signal transitions. A capture event may also optionally generate an interrupt.

DESCRIPTION

The PWM is based on the standard Timer block and inherits all of its features, although only the PWM function is pinned out on the LPC2119/2129/2194/2292/2294. The Timer is designed to count cycles of the peripheral clock (pclk) and optionally generate interrupts or perform other actions when specified timer values occur, based on seven match registers. It also includes four capture inputs to save the timer value when an input signal transitions, and optionally generate an interrupt when those events occur. The PWM function is in addition to these features, and is based on match register events.

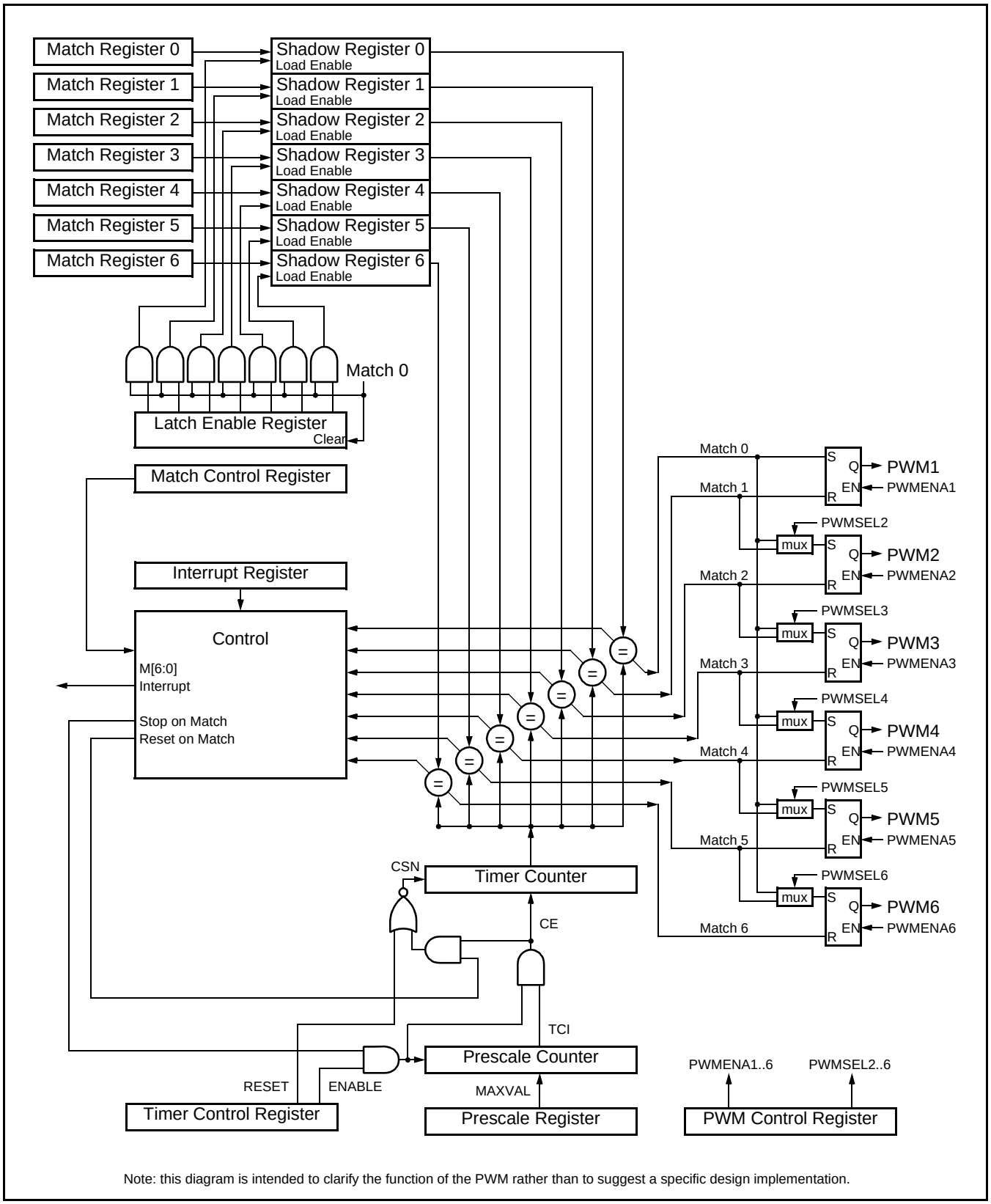
The ability to separately control rising and falling edge locations allows the PWM to be used for more applications. For instance, multi-phase motor control typically requires three non-overlapping PWM outputs with individual control of all three pulse widths and positions.

Two match registers can be used to provide a single edge controlled PWM output. One match register (PWMMR0) controls the PWM cycle rate, by resetting the count upon match. The other match register controls the PWM edge position. Additional single edge controlled PWM outputs require only one match register each, since the repetition rate is the same for all PWM outputs. Multiple single edge controlled PWM outputs will all have a rising edge at the beginning of each PWM cycle, when an PWMMR0 match occurs.

Three match registers can be used to provide a PWM output with both edges controlled. Again, the PWMMR0 match register controls the PWM cycle rate. The other match registers control the two PWM edge positions. Additional double edge controlled PWM outputs require only two match registers each, since the repetition rate is the same for all PWM outputs.

With double edge controlled PWM outputs, specific match registers control the rising and falling edge of the output. This allows both positive going PWM pulses (when the rising edge occurs prior to the falling edge), and negative going PWM pulses (when the falling edge occurs prior to the rising edge).

Figure 42 shows the block diagram of the PWM. The portions that have been added to the standard timer block are on the right hand side and at the top of the diagram.



Note: this diagram is intended to clarify the function of the PWM rather than to suggest a specific design implementation.

Figure 42: PWM block diagram

A sample of how PWM values relate to waveform outputs is shown in Figure 43. PWM output logic is shown in Figure 42 that allows selection of either single or double edge controlled PWM outputs via the muxes controlled by the PWMSELn bits. The match register selections for various PWM outputs is shown in Table 164. This implementation supports up to N-1 single edge PWM outputs or (N-1)/2 double edge PWM outputs, where N is the number of match registers that are implemented. PWM types can be mixed if desired.

The waveforms below show a single PWM cycle and demonstrate PWM outputs under the following conditions:

- The timer is configured for PWM mode.
- Match 0 is configured to reset the timer/counter when a match event occurs.
- Control bits PWMSEL2 and PWMSEL4 are set.
- The Match register values are as follows:
  MR0= 100(PWM rate)
  MR1= 41, MR2 = 78(PWM2 output)
  MR3= 53, MR4 = 27(PWM4 output)
  MR5= 65(PWM5 output)

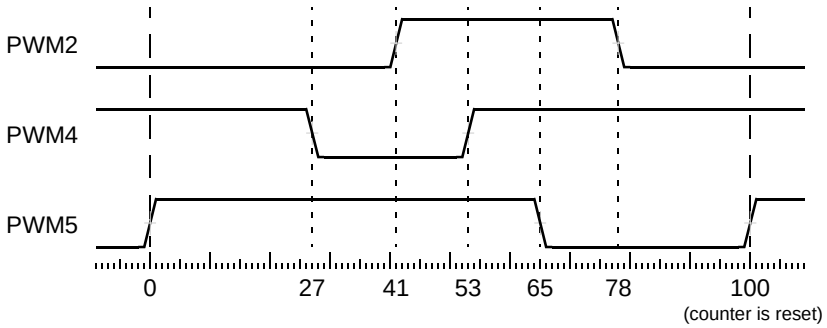


Figure 43: Sample PWM waveforms

Table 164: Set and Reset inputs for PWM Flip-Flops

PWM Channel	Single Edge PWM (PWMSELn = 0)		Double Edge PWM (PWMSELn = 1)	
	Set by	Reset by	Set by	Reset by
1	Match 0	Match 1	Match 0 [1]	Match 1 [1]
2	Match 0	Match 2	Match 1	Match 2
3	Match 0	Match 3	Match 2 [2]	Match 3 [2]
4	Match 0	Match 4	Match 3	Match 4
5	Match 0	Match 5	Match 4 [2]	Match 5 [2]
6	Match 0	Match 6	Match 5	Match 6

Notes:
1. Identical to single edge mode in this case since Match 0 is the neighboring match register. Essentially, PWM1 cannot be a double edged output.
2. It is generally not advantageous to use PWM channels 3 and 5 for double edge PWM outputs because it would reduce the number of double edge PWM outputs that are possible. Using PWM 2, PWM4, and PWM6 for double edge PWM outputs provides the most pairings.

Rules for Single Edge Controlled PWM Outputs

1. All single edge controlled PWM outputs go high at the beginning of a PWM cycle unless their match value is equal to 0.
2. Each PWM output will go low when its match value is reached. If no match occurs (i.e. the match value is greater than the PWM rate), the PWM output remains continuously high.

Rules for Double Edge Controlled PWM Outputs

Five rules are used to determine the next value of a PWM output when a new cycle is about to begin:

1. The match values for the next PWM cycle are used at the end of a PWM cycle (a time point which is coincident with the beginning of the next PWM cycle), except as noted in rule 3.
2. A match value equal to 0 or the current PWM rate (the same as the Match channel 0 value) have the same effect, except as noted in rule 3. For example, a request for a falling edge at the beginning of the PWM cycle has the same effect as a request for a falling edge at the end of a PWM cycle.
3. When match values are changing, if one of the "old" match values is equal to the PWM rate, it is used again once if the neither of the new match values are equal to 0 or the PWM rate, and there was no old match value equal to 0.
4. If both a set and a clear of a PWM output are requested at the same time, clear takes precedence. This can occur when the set and clear match values are the same as in, or when the set or clear value equals 0 and the other value equals the PWM rate.
5. If a match value is out of range (i.e. greater than the PWM rate value), no match event occurs and that match channel has no effect on the output. This means that the PWM output will remain always in one state, allowing always low, always high, or "no change" outputs.

PIN DESCRIPTION

Table 165 gives a brief summary of each of PWM related pins.

Table 165: Pin summary

Pin name	Pin direction	Pin Description
PWM1	Output	Output from PWM channel 1.
PWM2	Output	Output from PWM channel 2.
PWM3	Output	Output from PWM channel 3.
PWM4	Output	Output from PWM channel 4.
PWM5	Output	Output from PWM channel 5.
PWM6	Output	Output from PWM channel 6.

REGISTER DESCRIPTION

The PWM function adds new registers and registers bits as shown in Table 166 below.

Table 166: Pulse Width Modulator Register Map

Name	Description	Access	Reset Value*	Address
PWMIR	PWM Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of the possible interrupt sources are pending.	R/W	0	0xE0014000
PWMTCR	PWM Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.	R/W	0	0xE0014004
PWMTC	PWM Timer Counter. The 32-bit TC is incremented every PR+1 cycles of pclk. The TC is controlled through the TCR.	RW	0	0xE0014008
PWMPR	PWM Prescale Register. The TC is incremented every PR+1 cycles of pclk.	R/W	0	0xE001400C
PWMPC	PWM Prescale Counter. The 32-bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented.	R/W	0	0xE0014010
PWMMCR	PWM Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.	R/W	0	0xE0014014
PWMMR0	PWM Match Register 0. MR0 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR0 and the TC sets all PWM outputs that are in single-edge mode, and sets PWM1 if it is in double-edge mode.	R/W	0	0xE0014018
PWMMR1	PWM Match Register 1. MR1 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR1 and the TC clears PWM1 in either single-edge mode or double-edge mode, and sets PWM2 if it is in double-edge mode.	R/W	0	0xE001401C
PWMMR2	PWM Match Register 2. MR2 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR2 and the TC clears PWM2 in either single-edge mode or double-edge mode, and sets PWM3 if it is in double-edge mode.	R/W	0	0xE0014020
PWMMR3	PWM Match Register 3. MR3 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR3 and the TC clears PWM3 in either single-edge mode or double-edge mode, and sets PWM4 if it is in double-edge mode.	R/W	0	0xE0014024
PWMMR4	PWM Match Register 4. MR4 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR4 and the TC clears PWM4 in either single-edge mode or double-edge mode, and sets PWM5 if it is in double-edge mode.	R/W	0	0xE0014040

Table 166: Pulse Width Modulator Register Map

Name	Description	Access	Reset Value*	Address
PWMMR5	PWM Match Register 5. MR5 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR5 and the TC clears PWM5 in either single-edge mode or double-edge mode, and sets PWM6 if it is in double-edge mode.	R/W	0	0xE0014044
PWMMR6	PWM Match Register 6. MR6 can be enabled through MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt when it matches the TC. In addition, a match between MR6 and the TC clears PWM6 in either single-edge mode or double-edge mode.	R/W	0	0xE0014048
PWMPCR	PWM Control Register. Enables PWM outputs and selects PWM channel types as either single edge or double edge controlled.	R/W	0	0xE001404C
PWMLER	PWM Latch Enable Register. Enables use of new PWM match values.	R/W	0	0xE0014050

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

PWM Interrupt Register (PWMIR - 0xE0014000)

The PWM Interrupt Register consists of eleven bits (Table 167), seven for the match interrupts and four reserved for the future use. If an interrupt is generated then the corresponding bit in the PWMIR will be high. Otherwise, the bit will be low. Writing a logic one to the corresponding IR bit will reset the interrupt. Writing a zero has no effect.

Table 167: PWM Interrupt Register (PWMIR - 0xE0014000)

PWMIR	Function	Description	Reset Value
0	PWMMR0 Interrupt	Interrupt flag for PWM match channel 0.	0
1	PWMMR1 Interrupt	Interrupt flag for PWM match channel 1.	0
2	PWMMR2 Interrupt	Interrupt flag for PWM match channel 2.	0
3	MR3 Interrupt	Interrupt flag for PWM match channel 3.	0
4	Reserved.	Application must not write 1 to this bit.	0
5	Reserved.	Application must not write 1 to this bit.	0
6	Reserved.	Application must not write 1 to this bit.	0
7	Reserved.	Application must not write 1 to this bit.	0
8	PWMMR4 Interrupt	Interrupt flag for PWM match channel 4.	0
9	PWMMR5 Interrupt	Interrupt flag for PWM match channel 5.	0
10	PWMMR6 Interrupt	Interrupt flag for PWM match channel 6.	0

PWM Timer Control Register (PWMTCR - 0xE0014004)

The PWM Timer Control Register (PWMTCR) is used to control the operation of the PWM Timer Counter. The function of each of the bits is shown in Table 168.

Table 168: PWM Timer Control Register (PWMTCR - 0xE0014004)

PWMTCR	Function	Description	Reset Value
0	Counter Enable	When one, the PWM Timer Counter and PWM Prescale Counter are enabled for counting. When zero, the counters are disabled.	0
1	Counter Reset	When one, the PWM Timer Counter and the PWM Prescale Counter are synchronously reset on the next positive edge of pclk. The counters remain reset until TCR[1] is returned to zero.	0
2	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	PWM Enable	When one, PWM mode is enabled. PWM mode causes shadow registers to operate in connection with the Match registers. A program write to a Match register will not have an effect on the Match result until the corresponding bit in PWMLER has been set, followed by the occurrence of a PWM Match 0 event. Note that the PWM Match register that determines the PWM rate (PWM Match 0) must be set up prior to the PWM being enabled. Otherwise a Match event will not occur to cause shadow register contents to become effective.	0

PWM Timer Counter (PWMTC - 0xE0014008)

The 32-bit PWM Timer Counter is incremented when the Prescale Counter reaches its terminal count. Unless it is reset before reaching its upper limit, the PWMTC will count up through the value 0xFFFFFFFF and then wrap back to the value 0x00000000. This event does not cause an interrupt, but a Match register can be used to detect an overflow if needed.

PWM Prescale Register (PWMPR - 0xE001400C)

The 32-bit PWM Prescale Register specifies the maximum value for the PWM Prescale Counter.

PWM Prescale Counter Register (PWMPC - 0xE0014010)

The 32-bit PWM Prescale Counter controls division of pclk by some constant value before it is applied to the PWM Timer Counter. This allows control of the relationship of the resolution of the timer versus the maximum time before the timer overflows. The PWM Prescale Counter is incremented on every pclk. When it reaches the value stored in the PWM Prescale Register, the PWM Timer Counter is incremented and the PWM Prescale Counter is reset on the next pclk. This causes the PWM TC to increment on every pclk when PWMPR = 0, every 2 pclks when PWMPR = 1, etc.

PWM Match Registers (PWMMR0 - PWMMR6)

ThePWM Match register values are continuously compared to the PWM Timer Counter value. When the two values are equal, actions can be triggered automatically. The action possibilities are to generate an interrupt, reset the PWM Timer Counter, or stop the timer. Actions are controlled by the settings in the PWMMCR register.

PWM Match Control Register (PWMMCR - 0xE0014014)

The PWM Match Control Register is used to control what operations are performed when one of the PWM Match Registers matches the PWM Timer Counter. The function of each of the bits is shown in Table 169.

Table 169: PWM Match Control Register (PWMMCR - 0xE0014014)

PWMMCR	Function	Description	Reset Value
0	Interrupt on PWMMR0	When one, an interrupt is generated when PWMMR0 matches the value in the PWMTC. When zero this interrupt is disabled.	0
1	Reset on PWMMR0	When one, the PWMTC will be reset if PWMMR0 matches it. When zero this feature is disabled.	0
2	Stop on PWMMR0	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR0 matches the PWMTC. When zero this feature is disabled.	0
3	Interrupt on PWMMR1	When one, an interrupt is generated when PWMMR1 matches the value in the PWMTC. When zero this interrupt is disabled.	0
4	Reset on PWMMR1	When one, the PWMTC will be reset if PWMMR1 matches it. When zero this feature is disabled.	0
5	Stop on PWMMR1	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR1 matches the PWMTC. When zero this feature is disabled	0
6	Interrupt on PWMMR2	When one, an interrupt is generated when PWMMR2 matches the value in the PWMTC. When zero this interrupt is disabled.	0
7	Reset on PWMMR2	When one, the PWMTC will be reset if PWMMR2 matches it. When zero this feature is disabled.	0
8	Stop on PWMMR2	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR2 matches the PWMTC. When zero this feature is disabled	0
9	Interrupt on PWMMR3	When one, an interrupt is generated when PWMMR3 matches the value in the PWMTC. When zero this interrupt is disabled	0
10	Reset on PWMMR3	When one, the PWMTC will be reset if PWMMR3 matches it. When zero this feature is disabled.	0
11	Stop on PWMMR3	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR3 matches the PWMTC. When zero this feature is disabled	0
12	Interrupt on PWMMR4	When one, an interrupt is generated when PWMMR4 matches the value in the PWMTC. When zero this interrupt is disabled.	0
13	Reset on PWMMR4	When one, the PWMTC will be reset if PWMMR4 matches it. When zero this feature is disabled.	0
14	Stop on PWMMR4	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR4 matches the PWMTC. When zero this feature is disabled	0
15	Interrupt on PWMMR5	When one, an interrupt is generated when PWMMR5 matches the value in the PWMTC. When zero this interrupt is disabled.	0
16	Reset on PWMMR5	When one, the PWMTC will be reset if PWMMR5 matches it. When zero this feature is disabled.	0

Table 169: PWM Match Control Register (PWMMCR - 0xE0014014)

PWMMCR	Function	Description	Reset Value
17	Stop on PWMMR5	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR5 matches the PWMTC. When zero this feature is disabled	0
18	Interrupt on PWMMR6	When one, an interrupt is generated when PWMMR6 matches the value in the PWMTC. When zero this interrupt is disabled.	0
19	Reset on PWMMR6	When one, the PWMTC will be reset if PWMMR6 matches it. When zero this feature is disabled.	0
20	Stop on PWMMR6	When one, the PWMTC and PWMPC will be stopped and PWMTCR[0] will be set to 0 if PWMMR6 matches the PWMTC. When zero this feature is disabled	0

PWM Control Register (PWMPCR - 0xE001404C)

The PWM Control Register is used to enable and select the type of each PWM channel. The function of each of the bits are shown in Table 170.

Table 170: PWM Control Register (PWMPCR - 0xE001404C)

PWMPCR	Function	Description	Reset Value
1:0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
2	PWMSEL2	When zero, selects single edge controlled mode for PWM2. When one, selects double edge controlled mode for the PWM2 output.	0
3	PWMSEL3	When zero, selects single edge controlled mode for PWM3. When one, selects double edge controlled mode for the PWM3 output.	0
4	PWMSEL4	When zero, selects single edge controlled mode for PWM4. When one, selects double edge controlled mode for the PWM4 output.	0
5	PWMSEL5	When zero, selects single edge controlled mode for PWM5. When one, selects double edge controlled mode for the PWM5 output.	0
6	PWMSEL6	When zero, selects single edge controlled mode for PWM6. When one, selects double edge controlled mode for the PWM6 output.	0
8:7	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
9	PWMENA1	When one, enables the PWM1 output. When zero, disables the PWM1 output.	0
10	PWMENA2	When one, enables the PWM2 output. When zero, disables the PWM2 output.	0
11	PWMENA3	When one, enables the PWM3 output. When zero, disables the PWM3 output.	0
12	PWMENA4	When one, enables the PWM4 output. When zero, disables the PWM4 output.	0
13	PWMENA5	When one, enables the PWM5 output. When zero, disables the PWM5 output.	0
14	PWMENA6	When one, enables the PWM6 output. When zero, disables the PWM6 output.	0
15	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

PWM Latch Enable Register (PWMLER - 0xE0014050)

ThePWM Latch Enable Register is used to control the update of the PWM Match registers when they are used for PWM generation. When software writes to the location of a PWM Match register while the Timer is in PWM mode, the value is held in a shadow register. When a PWM Match 0 event occurs (normally also resetting the timer in PWM mode), the contents of shadow registers will be transferred to the actual Match registers if the corresponding bit in the Latch Enable Register has been set. At that point, the new values will take effect and determine the course of the next PWM cycle. Once the transfer of new values has taken place, all bits of the LER are automatically cleared. Until the corresponding bit in the PWMLER is set and a PWM Match 0 event occurs, any value written to the PWM Match registers has no effect on PWM operation.

For example, if PWM2 is configured for double edge operation and is currently running, a typical sequence of events for changing the timing would be:

- Write a new value to the PWM Match1 register.
- Write a new value to the PWM Match2 register.
- Write to the PWMLER, setting bits 1 and 2 at the same time.
- The altered values will become effective at the next reset of the timer (when a PWM Match 0 event occurs).

The order of writing the two PWM Match registers is not important, since neither value will be used until after the write to PWMLER. This insures that both values go into effect at the same time, if that is required. A single value may be altered in the same way if needed.

The function of each of the bits in the PWMLER is shown in Table 171.

Table 171: PWM Latch Enable Register (PWMLER - 0xE0014050)

PWMLER	Function	Description	Reset Value
0	Enable PWM Match 0 Latch	Writing a one to this bit allows the last value written to the PWM Match 0 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
1	Enable PWM Match 1 Latch	Writing a one to this bit allows the last value written to the PWM Match 1 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
2	Enable PWM Match 2 Latch	Writing a one to this bit allows the last value written to the PWM Match 2 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
3	Enable PWM Match 3 Latch	Writing a one to this bit allows the last value written to the PWM Match 3 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
4	Enable PWM Match 4 Latch	Writing a one to this bit allows the last value written to the PWM Match 4 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
5	Enable PWM Match 5 Latch	Writing a one to this bit allows the last value written to the PWM Match 5 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
6	Enable PWM Match 6 Latch	Writing a one to this bit allows the last value written to the PWM Match 6 register to be become effective when the timer is next reset by a PWM Match event. See the description of the PWM Match Control Register (PWMMCR).	0
7	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

17. A/D CONVERTER

FEATURES

- 10 bit successive approximation analog to digital converter.
- Input multiplexing among 4 pins (LPC2119/2129/2194) or 8 pins (LPC2292/2294)
- Power down mode
- Measurement range 0 to 3 V
- 10 bit conversion time >= 2.44 uS
- Burst conversion mode for single or multiple inputs
- Optional conversion on transition on input pin or Timer Match signal

DESCRIPTION

Basic clocking for the A/D converter is provided by the VPB clock. A programmable divider is included to scale this clock to the 4.5 MHz (max) clock needed by the successive approximation process. A fully accurate conversion requires 11 of these clocks.

PIN DESCRIPTIONS

Table 172: A/D Pin Description

Pin Name	Type	Pin Description
Ain7:0	Input	**Analog Inputs.** The A/D converter cell can measure the voltage on any of these 8 input signals, but the 64-pin packages restrict the choice to Ain3:0. Note that these analog inputs are always connected to their pins, even if the Pin Multiplexing Register assigns them to port pins. A simple self-test of the A/D Converter can be done by driving these pins as port outputs. **Note:** if the A/D converter is used, signal levels on analog input pins must not be above the level of V_{3A} at any time. Otherwise, A/D converter readings will be invalid. If the A/D converter is not used in an application then the pins associated with A/D inputs can be used as 5V tolerant digital IO pins.
V_{3A}, V_{SSA}	Power	**Analog Power and Ground.** These should be nominally the same voltages as V_3 and V_{SSD}, but should be isolated to minimize noise and error.

REGISTER DESCRIPTION

The base address of the A/D Converter is 0xE003 4000 (page 51). The A/D Converter includes 2 registers as shown in Table 173.

Table 173: A/D Registers

Name	Description	Access	Reset Value	Address
ADCR	A/D Control Register. The ADCR register must be written to select the operating mode before A/D conversion can occur.	Read/Write	0x0000 0001	0xE003 4000
ADDR	A/D Data Register. This register contains the ADC's DONE bit and (when DONE is 1) the 10-bit result of the conversion.	Read/Write	NA	0xE003 4004

A/D Control Register (ADCR - 0xE0034000)

Table 174: A/D Control Register (ADCR - 0xE0034000)

ADCR	Name	Description	Reset Value
7:0	SEL	Selects which of the Ain3:0 (LPC2119/2129/2194) or Ain7:0 (LPC2292/2294) pins is (are) to be sampled and converted. Only bits 3:0 should be set to 1 in the 48 or 64 pin package. In software-controlled mode, only one of these bits should be 1. In hardware scan mode, any value containing 1 to 8 ones (1 to 4 ones in the 48 or 64 pin package) can be used. All zeroes is equivalent to 0x01.	0x01
15:8	CLKDIV	The VPB clock (PCLK) is divided by (this value plus one) to produce the clock for the A/D converter, which should be less than or equal to 4.5 MHz. Typically, software should program the smallest value in this field that yields a clock of 4.5 MHz or slightly less, but in certain cases (such as a high-impedance analog source) a slower clock may be desirable.	0
16	BURST	If this bit is 0, conversions are software controlled and require 11 clocks. If this bit is 1, the AD converter does repeated conversions at the rate selected by the CLKS field, scanning (if necessary) through the pins selected by 1s in the SEL field. The first conversion after the start corresponds to the least-significant 1 in the SEL field, then higher numbered 1-bits (pins) if applicable. Repeated conversions can be terminated by clearing this bit, but the conversion that's in progress when this bit is cleared will be completed..	0
19:17	CLKS	This field selects the number of clocks used for each conversion in Burst mode, and the number of bits of accuracy of the result in the LS bits of ADDR, between 11 clocks (10 bits) and 4 clocks (3 bits): 000=11 clocks/10 bits, 001=10 clocks/9 bits,..., 111=4 clocks/3 bits	000
21	PDN	1: the A/D converter is operational 0: the A/D converter is in power down mode	0
23:22	TEST1:0	These bits are used in device testing. 00=normal operation, 01=digital test mode, 10=DAC test mode, and 11=simple conversion test mode.	0
26:24	START	When the BURST bit is 0, these bits control whether and when an A/D conversion is started: 000: no start (this value should be used when clearing PDN to 0) 001: start conversion now 010: start conversion when the edge selected by bit 27 occurs on P0.16/EINT0/MAT0.2/ CAP0.2 011: start conversion when the edge selected by bit 27 occurs on P0.22/TD3/CAP0.0/MAT0.0 *Note: for choices 100-111 the MAT signal need not be pinned out:* 100: start conversion when the edge selected by bit 27 occurs on MAT0.1 101: start conversion when the edge selected by bit 27 occurs on MAT0.3 110: start conversion when the edge selected by bit 27 occurs on MAT1.0 111: start conversion when the edge selected by bit 27 occurs on MAT1.1	000
27	EDGE	This bit is significant only when the START field contains 010-111. In these cases: 0: start conversion on a falling edge on the selected CAP/MAT signal 1: start conversion on a rising edge on the selected CAP/MAT signal	0

A/D Data Register (ADDR - 0xE0034004)

ADDR	Name	Description	Reset Value
31	DONE	This bit is set to 1 when an A/D conversion completes. It is cleared when this register is read and when the ADCR is written. If the ADCR is written while a conversion is still in progress, this bit is set and a new conversion is started.	0
30	OVERUN	This bit is 1 in burst mode if the results of one or more conversions was (were) lost and overwritten before the conversion that produced the result in the LS bits. In non-FIFO operation, this bit is cleared by reading this register.	0
29:27		These bits always read as zeroes. They could be used for expansion of the CHN field in future compatible A/D converters that can convert more channels.	0
26:24	CHN	These bits contain the channel from which the LS bits were converted.	X
23:16		These bits always read as zeroes. They allow accumulation of successive A/D values without AND-masking, for at least 256 values without overflow into the CHN field.	0
15:6	V/V_{3A}	When DONE is 1, this field contains a binary fraction representing the voltage on the Ain pin selected by the SEL field, divided by the voltage on the VddA pin. Zero in the field indicates that the voltage on the Ain pin was less than, equal to, or close to that on V_{SSA}, while 0x3FF indicates that the voltage on Ain was close to, equal to, or greater than that on V_{3A}. For testing, data written to this field is captured in a shift register that is clocked by the A/D converter clock. The MS bit of this register sources the DINSERI input of the A/D converter, which is used only when TEST1:0 are 10.	X
5:0		These bits always read as zeroes. They provide compatible expansion room for future, higher-resolution A/D converters.	0

Table 175: A/D Data Register (ADDR - 0xE0034004)

OPERATION

Hardware-Triggered Conversion

If the BURST bit in the ADCR is 0 and the START field contains 010-111, the A/D converter will start a conversion when a transition occurs on a selected pin or Timer Match signal . The choices include conversion on a specified edge of any of 4 Match signals, or conversion on a specified edge of either of 2 Capture/Match pins. The pin state from the selected pad or the selected Match signal, XORed with ADCR bit 27, is used in the edge detection logic.

Clock Generation

It is highly desirable that the clock divider for the 4.5 MHz conversion clock be held in a Reset state when the A/D converter is idle, so that the sampling clock can begin immediately when 01 is written to the START field of the ADCR, or the selected edge occurs on the selected signal. This feature also saves power, particularly if the A/D converter is used infrequently.

Interrupts

An interrupt request is asserted to the Vectored Interrupt Controller (VIC) when the DONE bit is 1. Software can use the Interrupt Enable bit for the A/D Converter in the VIC to control whether this assertion results in an interrupt. DONE is negated when the ADDR is read.

Accuracy vs. Digital Receiver

While the A/D converter can be used to measure the voltage on any AIN pin, regardless of the pin's setting in the Pin Select register (Pin Connect Block on page 126), selecting the AIN function improves the conversion accuracy by disabling the pin's digital receiver.

18. REAL TIME CLOCK

FEATURES

- Measures the passage of time to maintain a calendar and clock.
- Ultra Low Power design to support battery powered systems.
- Provides Seconds, Minutes, Hours, Day of Month, Month, Year, Day of Week, and Day of Year.
- Programmable Reference Clock Divider allows adjustment of the RTC to match various crystal frequencies.

DESCRIPTION

The Real Time Clock (RTC) is designed to provide a set of counters to measure time during system power on and off operation. The RTC has been designed to use little power, making it suitable for battery powered systems where the CPU is not running continuously (Idle mode).

ARCHITECTURE

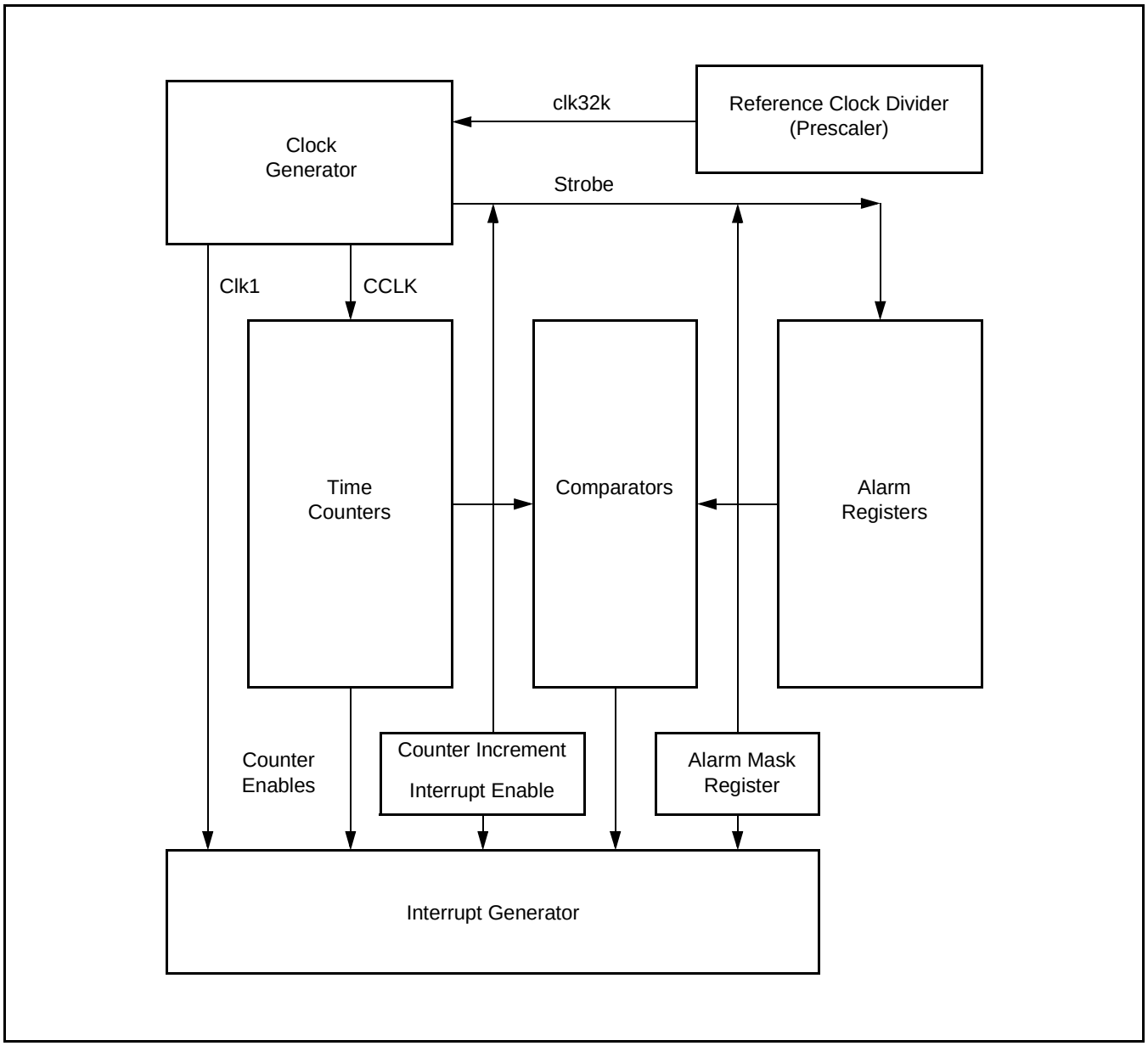


Figure 44: RTC block diagram

REGISTER DESCRIPTION

The RTC includes a number of registers. The address space is split into four sections by functionality. The first eight addresses are the Miscellaneous Register Group. The second set of eight locations are the Time Counter Group. The third set of eight locations contain the Alarm Register Group. The remaining registers control the Reference Clock Divider.

The Real Time Clock includes the register shown in Table 176. Detailed descriptions of the registers follow.

Table 176: Real Time Clock Register Map

Name	Size	Description	Access	Reset Value	Address
ILR	2	Interrupt Location Register	R/W	*	0xE0024000
CTC	15	Clock Tick Counter.	RO	*	0xE0024004
CCR	4	Clock Control Register	R/W	*	0xE0024008
CIIR	8	Counter Increment Interrupt Register	R/W	*	0xE002400C
AMR	8	Alarm Mask Register	R/W	*	0xE0024010
CTIME0	(32)	Consolidated Time Register 0	RO	*	0xE0024014
CTIME1	(32)	Consolidated Time Register 1	RO	*	0xE0024018
CTIME2	(32)	Consolidated Time Register 2	RO	*	0xE002401C
SEC	6	Seconds Register	R/W	*	0xE0024020
MIN	6	Minutes Register	R/W	*	0xE0024024
HOUR	5	Hours Register	R/W	*	0xE0024028
DOM	5	Day of Month Register	R/W	*	0xE002402C
DOW	3	Day of Week Register	R/W	*	0xE0024030
DOY	9	Day of Year Register	R/W	*	0xE0024034
MONTH	4	Months Register	R/W	*	0xE0024038
YEAR	12	Years Register	R/W	*	0xE002403C
ALSEC	6	Alarm value for Seconds	R/W	*	0xE0024060
ALMIN	6	Alarm value for Minutes	R/W	*	0xE0024064
ALHOUR	5	Alarm value for Hours	R/W	*	0xE0024068
ALDOM	5	Alarm value for Day of Month	R/W	*	0xE002406C
ALDOW	3	Alarm value for Day of Week	R/W	*	0xE0024070
ALDOY	9	Alarm value for Day of Year	R/W	*	0xE0024074
ALMON	4	Alarm value for Months	R/W	*	0xE0024078
ALYEAR	12	Alarm value for Year	R/W	*	0xE002407C
PREINT	13	Prescale value, integer portion	R/W	0	0xE0024080
PREFRAC	15	Prescale value, fractional portion	R/W	0	0xE0024084

* Registers in the RTC other than those that are part of the Prescaler are not affected by chip Reset. These registers must be initialized by software if the RTC is enabled.

RTC INTERRUPTS

Interrupt generation is controlled through the Interrupt Location Register (ILR), Counter Increment Interrupt Register (CIIR), the alarm registers, and the Alarm Mask Register (AMR). Interrupts are generated only by the transition into the interrupt state. The ILR separately enables CIIR and AMR interrupts. Each bit in CIIR corresponds to one of the time counters. If CIIR is enabled for a particular counter, then every time the counter is incremented an interrupt is generated. The alarm registers allow the user to specify a date and time for an interrupt to be generated. The AMR provides a mechanism to mask alarm compares. If all non-masked alarm registers match the value in their corresponding time counter, then an interrupt is generated.

MISCELLANEOUS REGISTER GROUP

Table 177 summarizes the registers located from 0 to 7 of A[6:2]. More detailed descriptions follow.

Table 177: Miscellaneous Registers

Address	Name	Size	Description	Access
0xE0024000	ILR	2	Interrupt Location. Reading this location indicates the source of an interrupt. Writing a one to the appropriate bit at this location clears the associated interrupt.	RW
0xE0024004	CTC	15	Clock Tick Counter. Value from the clock divider.	RO
0xE0024008	CCR	4	Clock Control Register. Controls the function of the clock divider.	RW
0xE002400C	CIIR	8	Counter Increment Interrupt. Selects which counters will generate an interrupt when they are incremented.	RW
0xE0024010	AMR	8	Alarm Mask Register. Controls which of the alarm registers are masked.	RW
0xE0024014	CTIME0	32	Consolidated Time Register 0	RO
0xE0024018	CTIME1	32	Consolidated Time Register 1	RO
0xE002401C	CTIME2	32	Consolidated Time Register 2	RO

Interrupt Location (ILR - 0xE0024000)

The Interrupt Location Register is a 2-bit register that specifies which blocks are generating an interrupt (see Table 178). Writing a one to the appropriate bit clears the corresponding interrupt. Writing a zero has no effect. This allows the programmer to read this register and write back the same value to clear only the interrupt that is detected by the read.

Table 178: Interrupt Location Register Bits (ILR - 0xE0024000)

ILR	Function	Description
0	RTCCIF	When one, the Counter Increment Interrupt block generated an interrupt. Writing a one to this bit location clears the counter increment interrupt.
1	RTCALF	When one, the alarm registers generated an interrupt. Writing a one to this bit location clears the alarm interrupt.

Clock Tick Counter (CTC - 0xE0024004)

The Clock Tick Counter is read only. It can be reset to zero through the Clock Control Register (CCR). The CTC consists of the bits of the clock divider counter.

Table 179: Clock Tick Counter Bits (CTC - 0xE0024004)

CTC	Function	Description
0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
15:1	Clock Tick Counter	Prior to the Seconds counter, the CTC counts 32,768 clocks per second. Due to the RTC Prescaler, these 32,768 time increments may not all be of the same duration. Refer to the Reference Clock Divider (Prescaler) description for details.

Clock Control Register (CCR - 0xE0024008)

The clock register is a 4-bit register that controls the operation of the clock divide circuit. Each bit of the clock register is described in Table 180.

Table 180: Clock Control Register Bits (CCR - 0xE0024008)

CCR	Function	Description
0	CLKEN	Clock Enable. When this bit is a one the time counters are enabled. When it is a zero, they are disabled so that they may be initialized.
1	CTCRST	CTC Reset. When one, the elements in the Clock Tick Counter are reset. The elements remain reset until CCR[1] is changed to zero.
3:2	CTTEST	Test Enable. These bits should always be zero during normal operation.

Counter Increment Interrupt

The Counter Increment Interrupt Register (CIIR) gives the ability to generate an interrupt every time a counter is incremented. This interrupt remains valid until cleared by writing a one to bit zero of the Interrupt Location Register (ILR[0]).

Table 181: Counter Increment Interrupt Register Bits (CIIR - 0xE002400C)

CIIR	Function	Description
0	IMSEC	When one, an increment of the Second value generates an interrupt.
1	IMMIN	When one, an increment of the Minute value generates an interrupt.
2	IMHOUR	When one, an increment of the Hour value generates an interrupt.
3	IMDOM	When one, an increment of the Day of Month value generates an interrupt.
4	IMDOW	When one, an increment of the Day of Week value generates an interrupt.
5	IMDOY	When one, an increment of the Day of Year value generates an interrupt.
6	IMMON	When one, an increment of the Month value generates an interrupt.
7	IMYEAR	When one, an increment of the Year value generates an interrupt.

Alarm Mask

The Alarm Mask Register (AMR) allows the user to mask any of the alarm registers. Table 182 shows the relationship between the bits in the AMR and the alarms. For the alarm function, every non-masked alarm register must match the corresponding time counter for an interrupt to be generated. The interrupt is generated only when the counter comparison first changes from no match to match. The interrupt is removed when a one is written to the appropriate bit of the Interrupt Location Register (ILR). If all mask bits are set, then the alarm is disabled.

Table 182: Alarm Mask Register Bits (AMR - 0xE0024010)

AMR	Function	Description
0	AMRSEC	When one, the Second value is not compared for the alarm.
1	AMRMIN	When one, the Minutes value is not compared for the alarm.
2	AMRHOUR	When one, the Hour value is not compared for the alarm.
3	AMRDOM	When one, the Day of Month value is not compared for the alarm.
4	AMRDOW	When one, the Day of Week value is not compared for the alarm.
5	AMRDOY	When one, the Day of Year value is not compared for the alarm.
6	AMRMON	When one, the Month value is not compared for the alarm.
7	AMRYEAR	When one, the Year value is not compared for the alarm.

CONSOLIDATED TIME REGISTERS

The values of the Time Counters can optionally be read in a consolidated format which allows the programmer to read all time counters with only three read operations. The various registers are packed into 32-bit values as shown in Tables 183, 184, and 185. The least significant bit of each register is read back at bit 0, 8, 16, or 24.

The Consolidated Time Registers are read only. To write new values to the Time Counters, the Time Counter addresses should be used.

Consolidated Time Register 0 (CTIME0 - 0xE0024014)

The Consolidated Time Register 0 contains the low order time values: Seconds, Minutes, Hours, and Day of Week.

Table 183: Consolidated Time Register 0 Bits (CTIME0 - 0xE0024014)

CTIME0	Function	Description
31:27	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
26:24	Day of Week	Day of week value in the range of 0 to 6.
23:21	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
20:16	Hours	Hours value in the range of 0 to 23.
15:14	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
13:8	Minutes	Minutes value in the range of 0 to 59.
7:6	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
5:0	Seconds	Seconds value in the range of 0 to 59.

Consolidated Time Register 1 (CTIME1 - 0xE0024018)

The Consolidate Time Register 1 contains the Day of Month, Month, and Year values.

Table 184: Consolidated Time Register 1 Bits (CTIME1 - 0xE0024018)

CTIME1	Function	Description
31:28	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
27:16	Year	Year value in the range of 0 to 4095.
15:12	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
11:8	Month	Month value in the range of 1 to 12.
7:5	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.
4:0	Day of Month	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year).

Consolidated Time Register 2 (CTIME2 - 0xE002401C)

The Consolidate Time Register 2 contains just the Day of Year value.

Table 185: Consolidated Time Register 2 Bits (CTIME2 - 0xE002401C)

CTIME2	Function	Description
11:0	Day of Year	Day of year value in the range of 1 to 365 (366 for leap years).
31:12	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.

TIME COUNTER GROUP

The time value consists of the eight counters shown in Tables 186 and 187. These counters can be read or written at the locations shown in Table 187.

Table 186: Time Counter Relationships and Values

Counter	Size	Enabled by	Min value	Maximum value
Second	6	Clk1 (see Figure 44)	0	59
Minute	6	Second	0	59
Hour	5	Minute	0	23
Day of Month	5	Hour	1	28,29,30, or 31
Day of Week	3	Hour	0	6
Day of Year	9	Hour	1	365 or 366 (for leap year)
Month	4	Day of Month	1	12
Year	12	Month or Day of Year	0	4095

Table 187: Time Counter registers

Address	Name	Size	Description	Access
0xE0024020	SEC	6	Seconds value in the range of 0 to 59.	R/W
0xE0024024	MIN	6	Minutes value in the range of 0 to 59.	R/W
0xE0024028	HOUR	5	Hours value in the range of 0 to 23.	R/W
0xE002402C	DOM	5	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year). [1]	R/W
0xE0024030	DOW	3	Day of week value in the range of 0 to 6. [1]	R/W
0xE0024034	DOY	9	Day of year value in the range of 1 to 365 (366 for leap years). [1]	R/W
0xE0024038	MONTH	4	Month value in the range of 1 to 12.	R/W
0xE002403C	YEAR	12	Year value in the range of 0 to 4095.	R/W

Notes:
1. These values are simply incremented at the appropriate intervals and reset at the defined overflow point. They are not calculated and must be correctly initialized in order to be meaningful.

Leap Year Calculation

The RTC does a simple bit comparison to see if the two lowest order bits of the year counter are zero. If true, then the RTC considers that year a leap year. The RTC considers all years evenly divisible by 4 as leap years. This algorithm is accurate from the year 1901 through the year 2099, but fails for the year 2100, which is not a leap year. The only effect of leap year on the RTC is to alter the length of the month of February for the month, day of month, and year counters.

ALARM REGISTER GROUP

The alarm registers are shown in Table 188. The values in these registers are compared with the time counters. If all the unmasked (See “Alarm Mask” on page 247.) alarm registers match their corresponding time counters then an interrupt is generated. The interrupt is cleared when a one is written to bit one of the Interrupt Location Register (ILR[1]).

Table 188: Alarm Registers

Address	Name	Size	Description	Access
0xE0024060	ALSEC	6	Alarm value for Seconds	R/W
0xE0024064	ALMIN	6	Alarm value for Minutes	R/W
0xE0024068	ALHOUR	5	Alarm value for Hours	R/W
0xE002406C	ALDOM	5	Alarm value for Day of Month	R/W
0xE0024070	ALDOW	3	Alarm value for Day of Week	R/W
0xE0024074	ALDOY	9	Alarm value for Day of Year	R/W
0xE0024078	ALMON	4	Alarm value for Months	R/W
0xE002407C	ALYEAR	12	Alarm value for Years	R/W

RTC USAGE NOTES

Since the RTC operates from the VPB clock (pclk), any interruption of that clock will cause the time to drift away from the time value it would have provided otherwise. The variance could be to actual clock time if the RTC was initialized to that, or simply an error in elapsed time since the RTC was activated.

No provision is made in the LPC2119/2129/2194/2292/2294 to retain RTC status upon power loss, or to maintain time incrementation if the clock source is lost, interrupted, or altered. Loss of chip power will result in complete loss of all RTC register contents. Entry to Power Down mode will cause a lapse in the time update. Altering the RTC timebase during system operation (by reconfiguring the PLL, the VPB timer, or the RTC prescaler) will result in some form of accumulated time error.

REFERENCE CLOCK DIVIDER (PRESCALER)

The reference clock divider (hereafter referred to as the Prescaler) allows generation of a 32.768 kHz reference clock from any peripheral clock frequency greater than or equal to 65.536 kHz (2 x 32.768 kHz). This permits the RTC to always run at the proper rate regardless of the peripheral clock rate. Basically, the Prescaler divides the peripheral clock (pclk) by a value which contains both an integer portion and a fractional portion. The result is not a continuous output at a constant frequency, some clock periods will be one pclk longer than others. However, the overall result can always be 32,768 counts per second.

The reference clock divider consists of a 13-bit integer counter and a 15-bit fractional counter. The reasons for these counter sizes are as follows:

1. For frequencies that are expected to be supported by the LPC2119/2129/2194/2292/2294, a 13-bit integer counter is required. This can be calculated as 160 MHz divided by 32,768 minus 1 = 4881 with a remainder of 26,624. Thirteen bits are needed to hold the value 4881, but actually supports frequencies up to 268.4 MHz (32,768 x 8192).
2. The remainder value could be as large as 32,767, which requires 15 bits.

Table 189: Reference Clock Divider registers

Address	Name	Size	Description	Access
0xE0024080	PREINT	13	Prescale Value, integer portion	R/W
0xE0024084	PREFRAC	15	Prescale Value, fractional portion	R/W

Prescaler Integer Register (PREINT - 0xE0024080)

This is the integer portion of the prescale value, calculated as:

PREINT = int (pclk / 32768) - 1. The value of PREINT must be greater than or equal to 1.

Table 190: Prescaler Integer Register (PREINT - 0xE0024080)

PREINT	Function	Description	Reset Value
15:13	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
12:0	Prescaler Integer	Contains the integer portion of the RTC prescaler value.	0

Prescaler Fraction Register (PREFRAC - 0xE0024084)

This is the fractional portion of the prescale value, and may be calculated as:

PREFRAC = pclk - ((PREINT +1) x 32768).

Table 191: Prescaler Fraction Register (PREFRAC - 0xE0024084)

PREFRAC	Function	Description	Reset Value
15	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
14:0	Prescaler Fraction	Contains the fractional portion of the RTC prescaler value.	0

Example of Prescaler Usage

In a simplistic case, the pclk frequency is 65.537 kHz. So:

PREINT = int (pclk / 32768) - 1 = 1 and PREFRAC = pclk - ((PREINT +1) x 32768) = 1

With this prescaler setting, exactly 32,768 clocks per second will be provided to the RTC by counting 2 pclks 32,767 times, and 3 pclks once.

In a more realistic case, the pclk frequency is 10 MHz. Then,

PREINT = int (pclk / 32768) - 1 = 304 and PREFRAC = pclk - ((PREINT +1) x 32768) = 5,760.

In this case, 5,760 of the prescaler output clocks will be 306 (305+1) pclks long, the rest will be 305 pclks long.

In a similar manner, any pclk rate greater than 65.536 kHz (as long as it is an even number of cycles per second) may be turned into a 32 kHz reference clock for the RTC. The only caveat is that if PREFRAC does not contain a zero, then not all of the 32,768 per second clocks are of the same length. Some of the clocks are one pclk longer than others. While the longer pulses are distributed as evenly as possible among the remaining pulses, this "jitter" could possibly be of concern in an application that wishes to observe the contents of the Clock Tick Counter (CTC) directly.

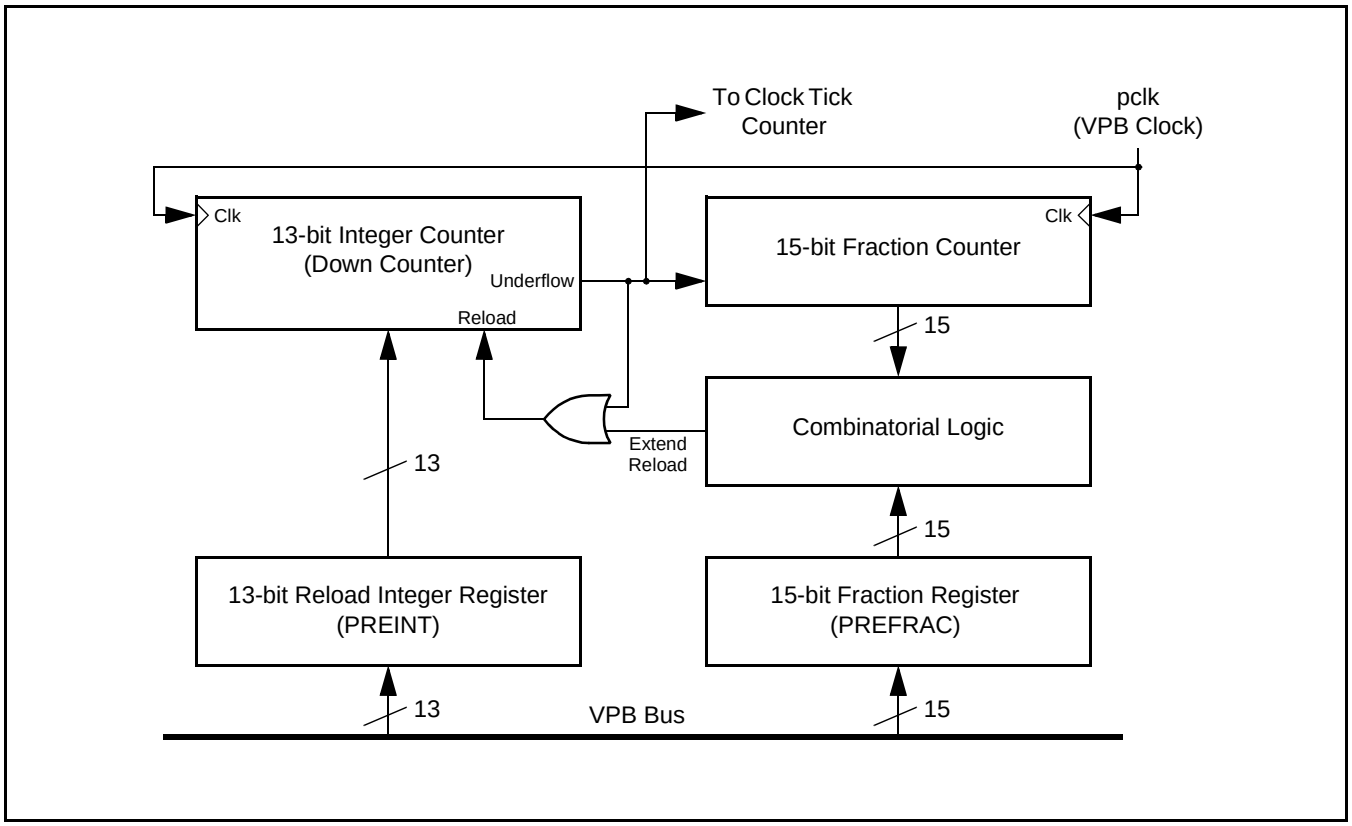


Figure 45: RTC Prescaler block diagram

Prescaler Operation

The Prescaler block labelled "Combination Logic" in Figure 45 determines when the decrement of the 13-bit PREINT counter is extended by one pclk. In order to both insert the correct number of longer cycles, and to distribute them evenly, the Combinatorial Logic associates each bit in PREFRAC with a combination in the 15-bit Fraction Counter. These associations are shown in (???).

For example, if PREFRAC bit 14 is a one (representing the fraction 1/2), then half of the cycles counted by the 13-bit counter need to be longer. When there is a 1 in the LSB of the Fraction Counter, the logic causes every alternate count (whenever the LSB of the Fraction Counter=1) to be extended by one pclk, evenly distributing the pulse widths. Similarly, a one in PREFRAC bit 13 (representing the fraction 1/4) will cause every fourth cycle (whenever the two LSBs of the Fraction Counter=10) counted by the 13-bit counter to be longer.

Table 192: Prescaler cases where the Integer Counter reload value is incremented

Fraction Counter	PREFRAC Bit														
	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
--- ---- ---- ---1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
--- ---- ---- --10	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-
--- ---- ---- -100	-	-	1	-	-	-	-	-	-	-	-	-	-	-	-
--- ---- ---- 1000	-	-	-	1	-	-	-	-	-	-	-	-	-	-	-
--- ---- ---1 0000	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-
--- ---- --10 0000	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-
--- ---- -100 0000	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-
--- ---- 1000 0000	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-
--- ---1 0000 0000	-	-	-	-	-	-	-	-	1	-	-	-	-	-	-
--- --10 0000 0000	-	-	-	-	-	-	-	-	-	1	-	-	-	-	-
--- -100 0000 0000	-	-	-	-	-	-	-	-	-	-	1	-	-	-	-
--- 1000 0000 0000	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-
--1 0000 0000 0000	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-
-10 0000 0000 0000	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-
100 0000 0000 0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1

19. WATCHDOG

FEATURES

- Internally resets chip if not periodically reloaded
- Debug mode
- Enabled by software but requires a hardware reset or a Watchdog reset/interrupt to be disabled
- Incorrect/Incomplete feed sequence causes reset/interrupt if enabled
- Flag to indicate Watchdog reset
- Programmable 32-bit timer with internal pre-scaler
- Selectable time period from (t_{pclk} x 256 x 4) to (t_{pclk} x 2^{32} x 4) in multiples of t_{pclk} x 4

APPLICATIONS

The purpose of the Watchdog is to reset the microcontroller within a reasonable amount of time if it enters an erroneous state. When enabled, the Watchdog will generate a system reset if the user program fails to "feed" (or reload) the Watchdog within a predetermined amount of time.

For interaction of the on-chip watchdog and other peripherals, especially the reset and boot-up procedures, please read "Reset" and "Boot Control on 144-pin Package" sections of this document.

DESCRIPTION

The Watchdog consists of a divide by 4 fixed pre-scaler and a 32-bit counter. The clock is fed to the timer via a pre-scaler. The timer decrements when clocked. The minimum value from which the counter decrements is 0xFF. Setting a value lower than 0xFF causes 0xFF to be loaded in the counter. Hence the minimum Watchdog interval is (t_{pclk} x 256 x 4) and the maximum Watchdog interval is (t_{pclk} x 2^{32} x 4) in multiples of (t_{pclk} x 4). The Watchdog should be used in the following manner:

- Set the Watchdog timer constant reload value in WDTC register.
- Setup mode in WDMOD register.
- Start the Watchdog by writing 0xAA followed by 0x55 to the WDFEED register.
- Watchdog should be fed again before the Watchdog counter underflows to prevent reset/interrupt.

When the Watchdog counter underflows, the program counter will start from 0x00000000 as in the case of external reset. The Watchdog time-out flag (WDTOF) can be examined to determine if the Watchdog has caused the reset condition. The WDTOF flag must be cleared by software.

REGISTER DESCRIPTION

The Watchdog contains 4 registers as shown in Table 193 below.

Table 193: Watchdog Register Map

Name	Description	Access	Reset Value*	Address
WDMOD	Watchdog mode register. This register contains the basic mode and status of the Watchdog Timer.	Read/Set	0	0xE0000000
WDTC	Watchdog timer constant register. This register determines the time-out value.	Read/Write	0xFF	0xE0000004
WDFEED	Watchdog feed sequence register. Writing AAh followed by 55h to this register reloads the Watchdog timer to its preset value.	Write Only	NA	0xE0000008
WDTV	Watchdog timer value register. This register reads out the current value of the Watchdog timer.	Read Only	0xFF	0xE000000C

*Reset Value refers to the data stored in used bits only. It does not include reserved bits content.

Watchdog Mode Register (WDMOD - 0xE0000000)

The WDMOD register controls the operation of the Watchdog as per the combination of WDEN and RESET bits.

WDEN	WDRESET	
0	X	Debug/Operate without the Watchdog running
1	0	Debug with the Watchdog interrupt but no WDRESET
1	1	Operate with the Watchdog interrupt and WDRESET

Once the WDEN and/or WDRESET bits are set they can not be cleared by software. Both flags are cleared by an external reset or a Watchdog timer underflow.

WDTOF The Watchdog time-out flag is set when the Watchdog times out. This flag is cleared by software.

WDINT The Watchdog interrupt flag is set when the Watchdog times out. This flag is cleared when any reset occurs.

Table 194: Watchdog Mode Register (WDMOD - 0xE0000000)

WDMOD	Function	Description	Reset Value
0	WDEN	Watchdog interrupt enable bit (Set only)	0
1	WDRESET	Watchdog reset enable bit (Set Only)	0
2	WDTOF	Watchdog time-out flag	0 (Only after external reset)
3	WDINT	Watchdog interrupt flag (Read Only)	0
7:4	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Watchdog Timer Constant Register (WDTC - 0xE0000004)

The WDTC register determines the time-out value. Every time a feed sequence occurs the WDTC content is reloaded in to the Watchdog timer. It's a 32-bit register with 8 LSB set to 1 on reset. Writing values below 0xFF will cause 0xFF to be loaded to the WDTC. Thus the minimum time-out interval is t_{pclk} x 256 x 4.

WDTC	Function	Description	Reset Value
31:0	Count	Watchdog time-out interval	0xFF

Watchdog Feed Register (WDFEED - 0xE0000008)

Writing 0xAA followed by 0x55 to this register will reload the Watchdog timer to the WDTC value. This operation will also start the Watchdog if it is enabled via the WDMOD register. Setting the WDEN bit in the WDMOD register is not sufficient to enable the Watchdog. A valid feed sequence must first be completed before the Watchdog is capable of generating an interrupt/reset. Until then, the Watchdog will ignore feed errors. Once 0xAA is written to the WDFEED register the next operation in the Watchdog register space should be a **WRITE** (0x55) to the WDFFED register otherwise the Watchdog is triggered. The interrupt/reset will be generated during the second **pclk** following an incorrect access to a watchdog timer register during a feed sequence.

Table 195: Watchdog Feed Register (WDFEED - 0xE0000008)

WDFEED	Function	Description	Reset Value
7:0	Feed	Feed value should be 0xAA followed by 0x55	undefined

Watchdog Timer Value Register (WDTV - 0xE000000C)

The WDTV register is used to read the current value of Watchdog timer.

Table 196: Watchdog Timer Value Register (WDTV - 0xE000000C)

WDTV	Function	Description	Reset Value
31:0	Count	Current timer value	0xFF

USAGE NOTES ON WATCHDOG RESET AND EXTERNAL START

When LPC2292/2294 is conditioned by components attached to the BOOT1:0 pins to start execution in off-chip memory, and is programmed to enable the Watchdog Timer to reset the part if it is not periodically serviced, care must be taken to avoid problems due to the interaction of these features. First, the BOOT1 and/or BOOT0 pin(s) must be biased to ground using pulldown resistors, not transistors driven from RESET low, because RESET is not driven low during a Watchdog Reset. Second, if either or both of the BOOT1:0 pins are used as inputs in the application, the application designer must ensure that the external driver will not be enabled during an internal Reset generated by the Watchdog Timer. (One way to do this is to use one of the CS3:0 outputs to enable the driver.) If these two conditions cannot be met, an external Watchdog facility can be used.

BLOCK DIAGRAM

The block diagram of the Watchdog is shown below in the Figure 46.

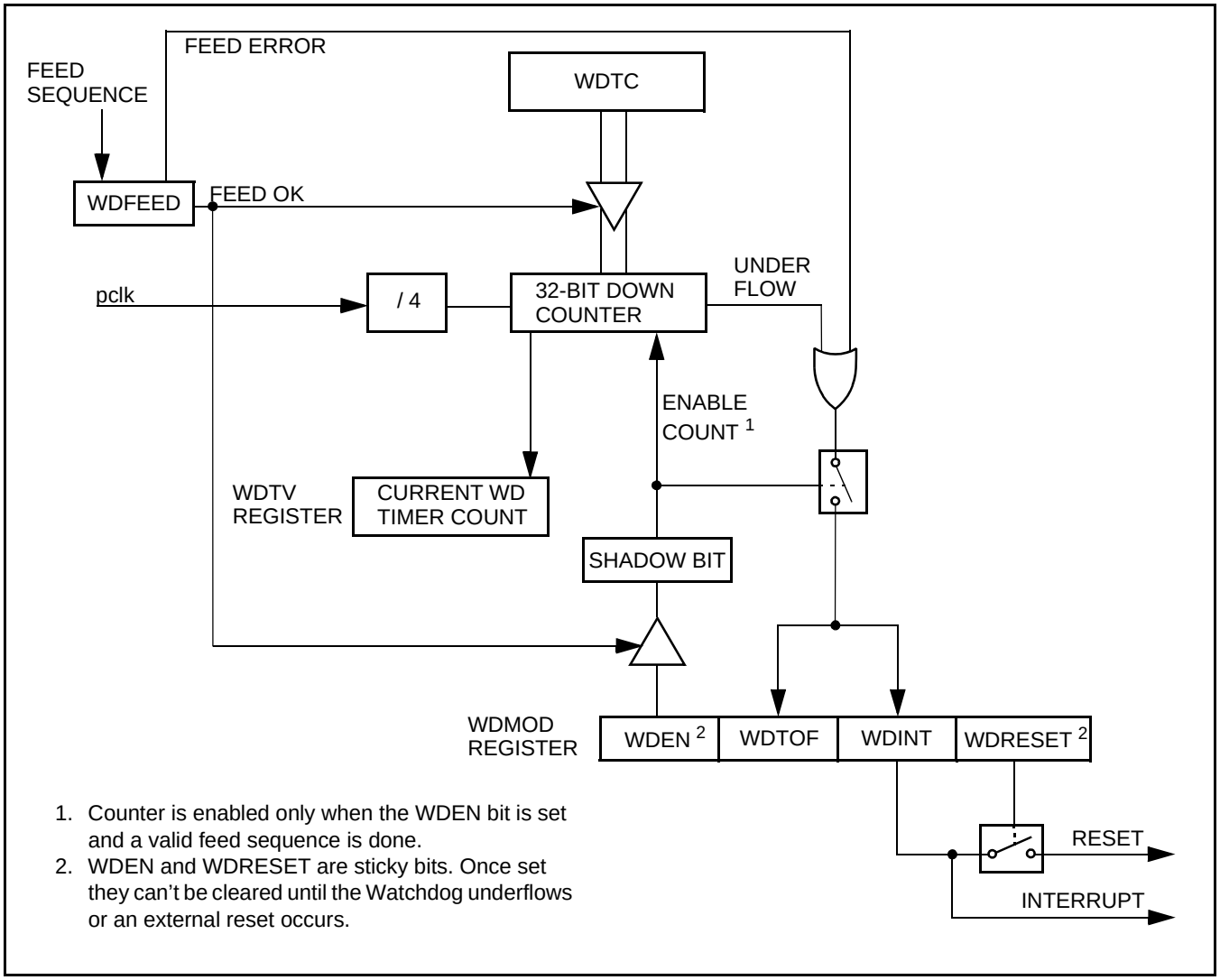


Figure 46: Watchdog Block Diagram

20. FLASH MEMORY SYSTEM AND PROGRAMMING

This chapter describes the Flash Memory System and the Boot Loader. It also includes In-System Programming (ISP) and In-Application Programming (IAP) interfaces.

FLASH MEMORY SYSTEM

The Flash Memory System contains 16 sectors for 128 kB part and 17 sectors for 256 kB part. Flash memory begins at address 0 and continues upward. Details may be found in the LPC2119/2129/2292/2294 Memory Addressing chapter.

On-chip Flash memory is capable of withstanding at least 10,000 erase and write cycles over the whole temperature range.

FLASH BOOT LOADER

The Boot Loader controls initial operation after reset, and also provides the means to accomplish programming of the Flash memory. This could be initial programming of a blank device, erasure and re-programming of a previously programmed device, or programming of the Flash memory by the application program in a running system.

FEATURES

- In-System Programming: In-System programming (**ISP**) is programming as well as reprogramming the on-chip flash memory, using the boot loader software and a serial port while the part may reside in the end-user system.
- In Application Programming: In-Application (**IAP**) programming is performing erase and write operation on the on-chip flash memory, as directed by the end-user application code.

APPLICATIONS

The flash boot loader provides both In-System and In-Application programming interfaces for programming the on-chip flash memory.

DESCRIPTION

The flash boot loader code is executed every time the part is powered on or reset. The loader can execute the ISP command handler or the user application code. A LOW level after reset at the P0.14 pin is considered as the external hardware request to start the ISP command handler. This pin is sampled in software. Asuming that proper signal is present on X1 pin when the rising edge on RST pin is generated, it may take up to 3 ms before P0.14 is sampled and the decision on wether to continue with user code or ISP handler is made. If P0.14 is sampled low and the watchdog overflow flag is set, the external hardware request to start the ISP command handler is ignored. If there is no request for the ISP command handler execution (P0.14 is sampled HIGH after reset), a search is made for a valid user program. If a valid user program is found then the execution control is transferred to it. If a valid user program is not found, the auto-baud routine is invoked.

Pin P0.14 that is used as hardware request for ISP requires special attention. Since P0.14 is in high impedance mode after reset, it is important that the user provides external hardware (a pull-up resistor or other device) to put the pin in a defined state. Otherwise unintended entry into ISP mode may occur.

Memory map after any reset:

The boot sector is 8 kB in size and resides in the top portion (starting from 0x0001 E000 in 128 kB Flash part and from 0x0003 E000 in 256 kb Flash part) of the on-chip flash memory. After any reset the entire boot sector is also mapped to the top

of the on-chip memory space i.e. the boot sector is also visible in the memory region starting from the address 0x7FFF E000. The flash boot loader is designed to run from this memory area but both the ISP and IAP software use parts of the on-chip RAM. The RAM usage is described later in this chapter. The interrupt vectors residing in the boot sector of the on-chip flash memory also become active after reset i.e. the bottom 64 bytes of the boot sector are also visible in the memory region starting from the address 0x0000 0000. The reset vector contains a jump instruction to the entry point of the flash boot loader software.

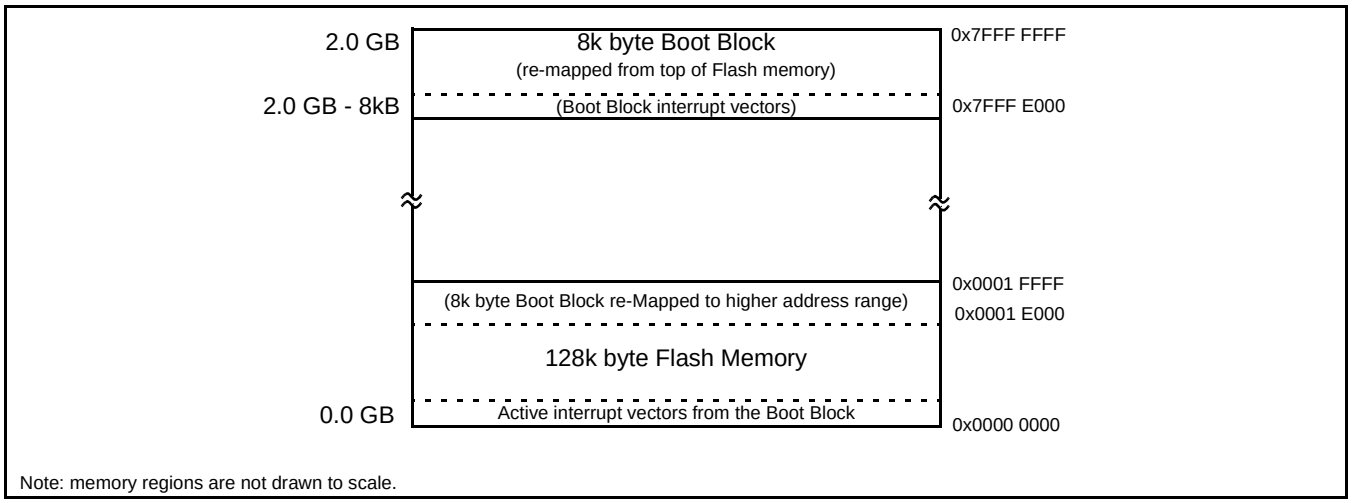


Figure 47: Map of lower memory after any reset (128 kB Flash part).

Criterion for valid user code: The reserved ARM interrupt vector location (0x0000 0014) should contain the 2's complement of the check-sum of the remaining interrupt vectors. This causes the checksum of all of the vectors together to be 0. The boot loader code disables the overlaying of the interrupt vectors from the boot block, then calculates the checksum of the interrupt vectors in sector 0 of the flash. If the signatures match then the execution control is transferred to the user code by loading the program counter with 0x 0000 0000. Hence the user flash reset vector should contain a jump instruction to the entry point of the user application code.

If the signature is not valid, the auto-baud routine synchronizes with the host via serial port 0. The host should send a synchronization character('?') and wait for a response. The host side serial port settings should be 8 data bits, 1 stop bit and no parity. The auto-baud routine measures the bit time of the received synchronization character in terms of its own frequency and programs the baud rate generator of the serial port. It also sends an ASCII string ("Synchronized<CR><LF>") to the host. In response to this the host should send the received string ("Synchronized<CR><LF>"). The auto-baud routine looks at the received characters to verify synchronization. If synchronization is verified then "OK<CR><LF>" string is sent to the host. The host should respond by sending the crystal frequency (in kHz) at which the part is running. For example if the part is running at 10 MHz a valid response from the host should be "10000<CR><LF>". "OK<CR><LF>" string is sent to the host after receiving the crystal frequency. If synchronization is not verified then the auto-baud routine waits again for a synchronization character. For auto-baud to work correctly, the crystal frequency should be greater than or equal to 10 MHz. The on-chip PLL is not used by the boot code.

Once the crystal frequency is received the part is initialized and the ISP command handler is invoked. For safety reasons an "Unlock" command is required before executing commands resulting in flash erase/write operations and the "Go" command. The rest of the commands can be executed without the unlock command. The "Unlock" command is required to be executed once per ISP session. Unlock command is explained in the "ISP Commands" section.

Communication Protocol

All ISP commands should be sent as single ASCII strings. Strings should be terminated with Carriage Return (CR) and/or Line Feed (LF) control characters. Extra <CR> and <LF> characters are ignored. All ISP responses are sent as <CR><LF> terminated ASCII strings. Data is sent and received in UU-encoded format.

ISP Command Format

"Command Parameter_0 Parameter_1 ... Parameter_n<CR><LF>" "Data" (Applicable only in case of Write commands)

ISP Response Format

"Return_Code<CR><LF>Response_0<CR><LF>Response_1<CR><LF> ... Response_n<CR><LF>" "Data" (Applicable in case of Read commands)

ISP Data Format

The data stream is in UU-encode format. The UU-encode algorithm converts 3 bytes of binary data in to 4 bytes of printable ASCII character set. It is more efficient than Hex format, which converts 1 byte of binary data in to 2 bytes of ASCII hex. The sender should send the check-sum after transmitting 20 UU-encoded lines. The length of any UU-encoded line should not exceed 61 characters(bytes) i.e. it can hold 45 data bytes. The receiver should compare it with the check-sum of the received bytes. If the check-sum matches then the receiver should respond with "OK<CR><LF>" to continue further transmission. If the check-sum does not match the receiver should respond with "RESEND<CR><LF>". In response the sender should retransmit the bytes.

A description of UU-encode is available at *http://www.wotsit.org.*

ISP Flow control

A software XON/XOFF flow control scheme is used to prevent data loss due to buffer overrun. When the data arrives rapidly, the ASCII control character DC3 (stop) is sent to stop the flow of data. Data flow is resumed by sending the ASCII control character DC1 (start). The host should also support the same flow control scheme.

ISP Command Abort

Commands can be aborted by sending the ASCII control character "ESC". This feature is not documented as a command under "ISP Commands" section. Once the escape code is received the ISP command handler waits for a new command.

Interrupts during ISP

Boot block Interrupt vectors located in the boot sector of the flash are active after any reset.

Interrupts during IAP

The on-chip flash memory is not accessible during erase/write operations. When the user application code starts executing the interrupt vectors from the user flash area are active. The user should either disable interrupts, or ensure that user interrupt vectors are active in RAM and that the interrupt handlers reside in RAM, before making a flash erase/write IAP call. The IAP code does not use or disable interrupts.

RAM used by ISP command handler

ISP commands use on-chip RAM from 0x4000 0120 to 0x4000 01FF. The user could use this area, but the contents may be lost upon reset. Flash programming commands use the top 32 bytes of on-chip RAM. The stack is located at RAM top - 32. The maximum stack usage is 256 bytes and it grows downwards.

RAM used by IAP command handler

Flash programming commands use top 32 bytes of on-chip RAM. The maximum stack usage in the user allocated stack space is 128 bytes and it grows downwards.

RAM used by RealMonitor

The RealMonitor uses on-chip RAM from 0x4000 0040 to 0x4000 011F. The user could use this area if RealMonitor based debug is not required. The Flash boot loader does not initialize the stack for the RealMonitor.

BOOT PROCESS FLOWCHART

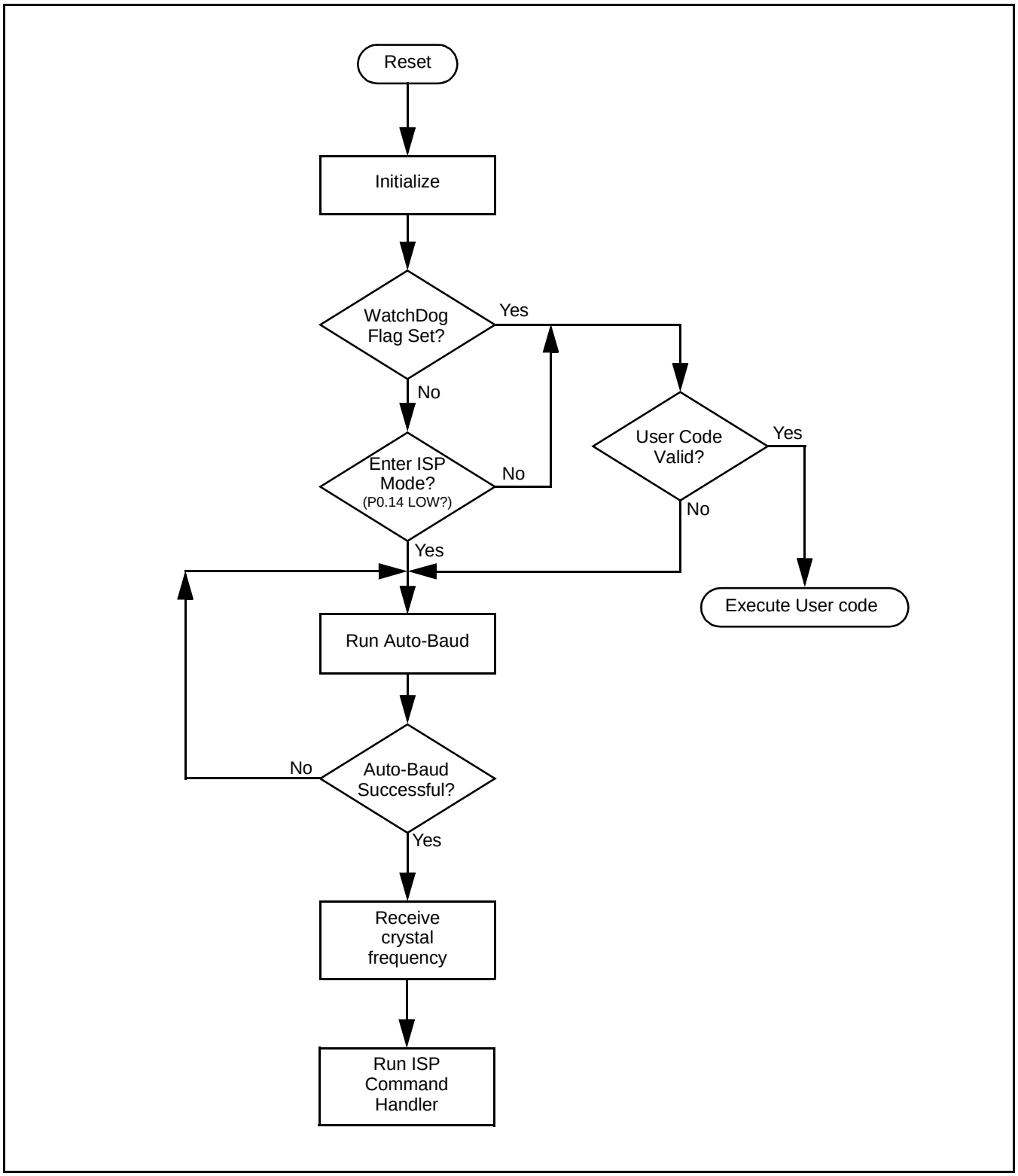


Figure 48: Boot Process flowchart (Bootloader revisions before 1.61)

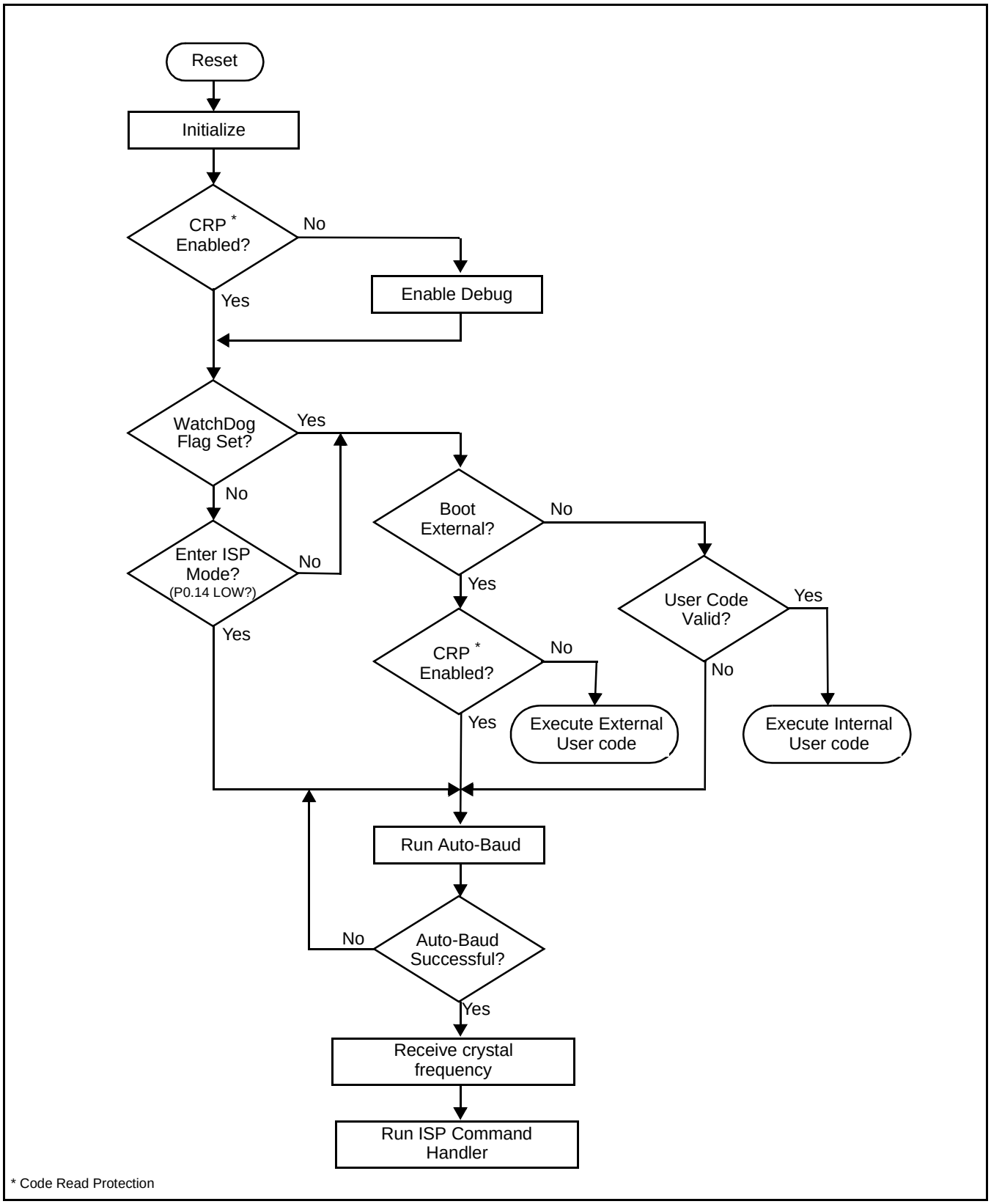


Figure 49: Boot Process flowchart (Bootloader revisions 1.61 and later)

SECTOR NUMBERS

Some IAP and ISP commands operate on "sectors" and specify sector numbers. The following table indicates the correspondence between sector numbers and memory addresses for LPC2119/2129/2194/2292/2294 device(s). IAP, ISP and RealMonitor routines are located in the Boot Sector. The boot sector is present in all devices. ISP and IAP commands do not allow write/erase/go operation on the boot sector. In a device having 128K of Flash, only 120 kB is available for the user program. Devices with the total of 256 kB of Flash, allow user code of up to 248 kB.

Table 197: Sectors in a device with 128K bytes of Flash

Sector Number	Memory Addresses and Sector Sizes			
	128 kB part	Sector size [kB]	256 kB part	Sector size [kB]
0	0x0000 0000 - 1FFF	8	0x0000 0000 - 1FFF	8
1	0x0000 2000 - 3FFF	8	0x0000 2000 - 3FFF	8
2	0x0000 4000 - 5FFF	8	0x0000 4000 - 5FFF	8
3	0x0000 6000 - 7FFF	8	0x0000 6000 - 7FFF	8
4	0x0000 8000 - 9FFF	8	0x0000 8000 - 9FFF	8
5	0x0000 A000 - BFFF	8	0x0000 A000 - BFFF	8
6	0x0000 C000 - DFFF	8	0x0000 C000 - DFFF	8
7	0x0000 E000 - FFFF	8	0x0000 E000 - FFFF	8
8	0x0001 0000 - 1FFF	8	0x0001 0000 - FFFF	64
9	0x0001 2000 - 3FFF	8	0x0002 0000 - FFFF	64
10 (0x0A)	0x0001 4000 - 5FFF	8	0x0003 0000 - 1FFF	8
11 (0x0B)	0x0001 6000 - 7FFF	8	0x0003 2000 - 3FFF	8
12 (0x0C)	0x0001 8000 - 9FFF	8	0x0003 4000 - 5FFF	8
13 (0x0D)	0x0001 A000 - BFFF	8	0x0003 6000 - 7FFF	8
14 (0x0E)	0x0001 C000 - DFFF	8	0x0003 8000 - 9FFF	8
15 (0x0F)	0x0001 E000 - FFFF*	8	0x0003 A000 - BFFF	8
16 (0x10)			0x0003 C000 - DFFF	8
17 (0x11)			0x0003 E000 - FFFF*	8

* Boot Block always resides on the top of the on-chip available Flash memory. In case of 128 kB Flash, it is the 16th sector (sector with logical number 15), and in case of 256 kB Flash, it is the 18th sector (sector with logical number 17). Flash memory sector where Boot Block resides is not available for user to store code.

CODE READ PROTECTION

This feature is available as of Bootloader revision 1.61.

Code read protection is enabled by programming the flash address location 0x1FC (User flash sector 0) with value 0x87654321 (2271560481 Decimal). Address 0x1FC is used to allow some room for the fiq exception handler. When the code read protection is enabled the JTAG debug port, external memory boot and the following ISP commands are disabled:

- Read Memory
- Write to RAM
- Go
- Copy RAM to Flash

The ISP commands mentioned above terminate with return code CODE_READ_PROTECTION_ENABLED.

The ISP erase command only allows erasure of all user sectors when the code read protection is enabled. This limitation does not exist if the code read protection is not enabled. IAP commands are not affected by the code read protection.

ISP Commands

The following commands are accepted by the ISP command handler. Detailed return codes are supported for each command. The command handler sends the return code INVALID_COMMAND when an undefined command is received. Commands and return codes are in ASCII format.

CMD_SUCCESS is sent by ISP command handler only when received ISP command has been completely executed and the new ISP command can be given by the host. Exceptions from this rule are "Set Baud Rate", "Write to RAM", "Read Memory", and "Go" commands.

Table 198: ISP Command Summary

ISP Command	Usage	Described in
Unlock	U <Unlock Code>	Table 199
Set Baud Rate	B <Baud Rate> <stop bit>	Table 200
Echo	A <setting>	Table 202
Write to RAM	W <start address> <number of bytes>	Table 203
Read Memory	R <address> <number of bytes>	Table 204
Prepare sector(s) for write operation	P <start sector number> <end sector number>	Table 205
Copy RAM to Flash	C <Flash address> <RAM address> <number of bytes>	Table 206
Go	G <address> <Mode>	Table 207
Erase sector(s)	E <start sector number> <end sector number>	Table 208
Blank check sector(s)	I <start sector number> <end sector number>	Table 209
Read Part ID	J	Table 210
Read Boot code version	K	Table 211
Compare	M <address1> <address2> <number of bytes>	Table 212

Unlock <Unlock code>

Table 199: ISP Unlock command description

Command	U
Input	Unlock code: 23130
Return Code	CMD_SUCCESS | INVALID_CODE | PARAM_ERROR
Description	This command is used to unlock flash Write/Erase & Go commands.
Example	"U 23130<CR><LF>" unlocks the flash Write/Erase & Go commands.

Set Baud Rate <Baud Rate> <stop bit>

Table 200: ISP Set Baud Rate command description

Command	B
Input	Baud Rate: 9600 | 19200 | 38400 | 57600 | 115200 | 230400 Stop bit: 1 | 2
Return Code	CMD_SUCCESS | INVALID_BAUD_RATE | INVALID_STOP_BIT | PARAM_ERROR
Description	This command is used to change the baud rate. The new baud rate is effective after the command handler sends the CMD_SUCCESS return code.
Example	"B 57600 1<CR><LF>" sets the serial port to baud rate 57600 bps and 1 stop bit.

Table 201: Correlation between possible ISP baudrates and external crystal frequency (in MHz)

ISP Baudrate .vs. External Crystal Frequency	9600	19200	38400	57600	115200	230400
10.0000	+	+	+			
11.0592	+	+		+		
12.2880	+	+	+			
14.7456	+	+	+	+	+	+
15.3600	+					
18.4320	+	+		+		
19.6608	+	+	+			
24.5760	+	+	+			
25.0000	+	+	+			

Echo <setting>

Table 202: ISP Echo command description

Command	A
Input	Setting: ON=1 | OFF=0
Return Code	CMD_SUCCESS | PARAM_ERROR
Description	The default setting for echo command is ON. When ON the ISP command handler sends the received serial data back to the host.
Example	"A 0<CR><LF>" turns echo off.

Write to RAM <start address> <number of bytes>

The host should send the data only after receiving the CMD_SUCCESS return code. The host should send the check-sum after transmitting 20 UU-encoded lines. The checksum is generated by adding raw data (before UU-encoding) bytes and is reset after transmitting 20 UU-encoded lines. The length of any UU-encoded line should not exceed 61 characters(bytes) i.e. it can hold 45 data bytes. When the data fits in less then 20 UU-encoded lines then the check-sum should be of actual number of bytes sent. The ISP command handler compares it with the check-sum of the received bytes. If the check-sum matches then the ISP command handler responds with "OK<CR><LF>" to continue further transmission. If the check-sum does not match then the ISP command handler responds with "RESEND<CR><LF>". In response the host should retransmit the bytes.

Table 203: ISP Write to RAM command description

Command	W
Input	Start Address: RAM address where data bytes are to be written. This address should be a word boundary. Number of Bytes: Number of bytes to be written. Count should be a multiple of 4.
Return Code	CMD_SUCCESS | ADDR_ERROR (Address not a word boundary) | ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not multiple of 4) | PARAM_ERROR CODE_READ_PROTECTION_ENABLED
Description	This command is used to download data to RAM. The data should be in UU-encoded format. As of Bootloader rev. 1.61 this command is blocked when code read protection is enabled.
Example	"W 1073742336 4<CR><LF>" writes 4 bytes of data to address 0x4000 0200.

Read Memory <address> <number of bytes>

The data stream is followed by the command success return code. The check-sum is sent after transmitting 20 UU-encoded lines. The checksum is generated by adding raw data (before UU-encoding) bytes and is reset after transmitting 20 UU-encoded lines. The length of any UU-encoded line should not exceed 61 characters(bytes) i.e. it can hold 45 data bytes. When the data fits in less then 20 UU-encoded lines then the check-sum is of actual number of bytes sent. The host should compare it with the check-sum of the received bytes. If the check-sum matches then the host should respond with "OK<CR><LF>" to continue further transmission. If the check-sum does not match then the host should respond with "RESEND<CR><LF>". In response the ISP command handler sends the data again.

Table 204: ISP Read Memory command description

Command	R
Input	Start Address: Address from where data bytes are to be read. This address should be a word boundary. Number of Bytes: Number of bytes to be read. Count should be a multiple of 4.
Return Code	CMD_SUCCESS followed by <actual data (UU-encoded)> | ADDR_ERROR (Address not on word boundary) | ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not multiple of 4) | PARAM_ERROR CODE_READ_PROTECTION_ENABLED
Description	This command is used to read data from RAM or Flash memory. As of Bootloader rev. 1.61 this command is blocked when code read protection is enabled.
Example	"R 1073741824 4<CR><LF>" reads 4 bytes of data from address 0x4000 0000.

Prepare sector(s) for write operation <start sector number> <end sector number>

This command makes flash write/erase operation a two step process.

Table 205: ISP Prepare sector(s) for write operation command description

Command	P
Input	Start Sector Number End Sector Number: Should be greater than or equal to start sector number.
Return Code	CMD_SUCCESS | BUSY | INVALID_SECTOR | PARAM_ERROR
Description	This command must be executed before executing "Copy RAM to Flash" or "Erase Sector(s)" command. Successful execution of the "Copy RAM to Flash" or "Erase Sector(s)" command causes relevant sectors to be protected again. The boot sector can not be prepared by this command. To prepare a single sector use the same "Start" and "End" sector numbers.
Example	"P 0 0<CR><LF>" prepares the flash sector 0.

Copy RAM to Flash <Flash address> <RAM address> <number of bytes>

Table 206: ISP Copy RAM to Flash command description

Command	C
Input	Flash Address(DST): Destination Flash address where data bytes are to be written. The destination address should be a 512 byte boundary. RAM Address(SRC): Source RAM address from where data bytes are to be read. Number of Bytes: Number of bytes to be written. Should be 512 | 1024 | 4096 | 8192.
Return Code	CMD_SUCCESS | SRC_ADDR_ERROR (Address not on word boundary) | DST_ADDR_ERROR (Address not on correct boundary) | SRC_ADDR_NOT_MAPPED | DST_ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not 512 | 1024 | 4096 | 8192) | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | BUSY | CMD_LOCKED | PARAM_ERROR CODE_READ_PROTECTION_ENABLED
Description	This command is used to program the flash memory. The affected sectors should be prepared first by calling "Prepare Sector for Write Operation" command. The affected sectors are automatically protected again once the copy command is successfully executed. The boot sector can not be written by this command. As of Bootloader rev. 1.61 this command is blocked when code read protection is enabled.
Example	"C 0 1073774592 512<CR><LF>" copies 512 bytes from the RAM address 0x4000 8000 to the flash address 0.

Go <address> <Mode>

Table 207: ISP Go command description

Command	G
Input	Address: Flash or RAM address from which the code execution is to be started. This address should be on a word boundary. Mode: T (Execute program in Thumb Mode) | A (Execute program in ARM mode)
Return Code	CMD_SUCCESS | ADDR_ERROR | ADDR_NOT_MAPPED | CMD_LOCKED | PARAM_ERROR CODE_READ_PROTECTION_ENABLED
Description	This command is used to execute (call) a program residing in RAM or Flash memory. It may not be possible to return to ISP command handler once this command is successfully executed. If executed code has ended with return instruction, ISP handler will resume with execution. As of Bootloader rev. 1.61 this command is blocked when code read protection is enabled.
Example	"G 0 A<CR><LF>" branches to address 0x0000 0000 in ARM mode.

Erase sector(s) <start sector number> <end sector number>

Table 208: ISP Erase sector command description

Command	E
Input	Start Sector Number End Sector Number: Should be greater than or equal to start sector number.
Return Code	CMD_SUCCESS | BUSY | INVALID_SECTOR | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | CMD_LOCKED | PARAM_ERROR CODE_READ_PROTECTION_ENABLED
Description	This command is used to erase a sector or multiple sectors of on-chip Flash memory. The boot sector can not be erased by this command. To erase a single sector use the same "Start" and "End" sector numbers. As of Bootloader rev. 1.61 this command is blocked when code read protection is enabled.
Example	"E 2 3<CR><LF>" erases the flash sectors 2 and 3.

Blank check sector(s) <start sector number> <end sector number>

Table 209: ISP Blank check sector(s) command description

Command	I
Input	Start Sector Number End Sector Number: Should be greater than or equal to start sector number.
Return Code	CMD_SUCCESS | SECTOR_NOT_BLANK (followed by <Offset of the first non blank word location> <Contents of non blank word location>) | INVALID_SECTOR | PARAM_ERROR
Description	This command is used to blank check a sector or multiple sectors of on-chip Flash memory. To blank check a single sector use the same "Start" and "End" sector numbers.
Example	"I 2 3<CR><LF>" blank checks the flash sectors 2 and 3. **Blank check on sector 0 always fails as first 64 bytes are re-mapped to flash boot sector.**

Read Part ID

Table 210: ISP Read Part ID command description

Command	J
Input	None
Return Code	CMD_SUCCESS followed by part identification number in ASCII format.
Description	This command is used to read the part identification number.
Example	"J<CR><LF>".

Read Boot code version

Table 211: ISP Read Boot Code version command description

Command	K
Input	None
Return Code	CMD_SUCCESS followed by 2 bytes of boot code version number in ASCII format. It is to be interpreted as <byte1(Major)>.<byte0(Minor)>
Description	This command is used to read the boot code version number.
Example	"K<CR><LF>".

Compare <address1> <address2> <number of bytes>

Table 212: ISP Compare command description

Command	M
Input	Address1(DST): Starting Flash or RAM address from where data bytes are to be compared. This address should be on word boundary. Address2(SRC): Starting Flash or RAM address from where data bytes are to be compared. This address should be on word boundary. Number of Bytes: Number of bytes to be compared. Count should be in multiple of 4.
Return Code	CMD_SUCCESS | (Source and destination data is same) COMPARE_ERROR | (Followed by the offset of first mismatch) COUNT_ERROR (Byte count is not multiple of 4) | ADDR_ERROR | ADDR_NOT_MAPPED | PARAM_ERROR
Description	This command is used to compare the memory contents at two locations.
Example	"M 8192 1073741824 4<CR><LF>" compares 4 bytes from the RAM address 0x4000 0000 to the 4 bytes from the flash address 0x2000. **Compare result may not be correct when source or destination address contains any of the first 64 bytes starting from address zero. First 64 bytes are re-mapped to flash boot sector.**

Table 213: ISP Return Codes Summary

Return Code	Mnemonic	Description
0	CMD_SUCCESS	Command is executed successfully. Sent by ISP handler only when command given by the host has been completely and successfully executed.
1	INVALID_COMMAND	Invalid command.
2	SRC_ADDR_ERROR	Source address is not on word boundary.
3	DST_ADDR_ERROR	Destination address is not on a correct boundary.
4	SRC_ADDR_NOT_MAPPED	Source address is not mapped in the memory map. Count value is taken in to consideration where applicable.
5	DST_ADDR_NOT_MAPPED	Destination address is not mapped in the memory map. Count value is taken in to consideration where applicable.
6	COUNT_ERROR	Byte count is not multiple of 4 or is not a permitted value.
7	INVALID_SECTOR	Sector number is invalid or end sector number is greater than start sector number.
8	SECTOR_NOT_BLANK	Sector is not blank.
9	SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION	Command to prepare sector for write operation was not executed.
10	COMPARE_ERROR	Source and destination data not equal.
11	BUSY	Flash programming hardware interface is busy
12	PARAM_ERROR	Insufficient number of parameters or invalid parameter.
13	ADDR_ERROR	Address is not on word boundary.
14	ADDR_NOT_MAPPED	Address is not mapped in the memory map. Count value is taken in to consideration where applicable.
15	CMD_LOCKED	Command is locked.
16	INVALID_CODE	Unlock code is invalid.
17	INVALID_BAUD_RATE	Invalid baud rate setting.
18	INVALID_STOP_BIT	Invalid stop bit setting.
19	CODE_READ_PROTECTION_ENABLED	Code read protection enabled. Available as of Bootloader rev. 1.61

IAP Commands

For in application programming the IAP routine should be called with a word pointer in register r0 pointing to memory (RAM) containing command code and parameters. Result of the IAP command is returned in the result table pointed to by register r1. The user can reuse the command table for result by passing the same pointer in registers r0 and r1. The parameter table should be big enough to hold all the results in case if number of results are more than number of parameters. Parameter passing is illustrated in the Figure 50. The number of parameters and results vary according to the IAP command. The maximum number of parameters is 5, passed to the "Copy RAM to FLASH" command. The maximum number of results is 2, returned by the "Blank check sector(s)" command. The command handler sends the status code INVALID_COMMAND when an undefined command is received. The IAP routine resides at 0x7FFFFFF0 location and it is thumb code.

The IAP function could be called in the following way using C.

Define the IAP location entry point. Since the 0th bit of the IAP location is set there will be a change to Thumb instruction set when the program counter branches to this address.

```
#define IAP_LOCATION 0x7ffffff1
```

Define data structure or pointers to pass IAP command table and result table to the IAP function

```
unsigned long command[5];

unsigned long result[2];
```

or

```
unsigned long * command;

unsigned long * result;

command=(unsigned long *) 0x……

result= (unsigned long *) 0x……
```

Define pointer to function type, which takes two parameters and returns void. Note the IAP returns the result with the base address of the table residing in R1.

```
typedef void (*IAP)(unsigned int [],unsigned int[]);

IAP iap_entry;
```

Setting function pointer

```
iap_entry=(IAP) IAP_LOCATION;
```

Whenever you wish to call IAP you could use the following statement.

```
iap_entry (command, result);
```

The IAP call could be simplified further by using the symbol definition file feature supported by ARM Linker in ADS (ARM Developer Suite). You could also call the IAP routine using assembly code.

The following symbol definitions can be used to link IAP routine and user application:

```
#<SYMDEFS># ARM Linker, ADS1.2 [Build 826]: Last Updated: Wed May 08 16:12:23 2002

0x7fffff90 T rm_init_entry

0x7fffffa0 A rm_undef_handler
```

```
0x7ffffb0 A rm_prefetchabort_handler
0x7ffffc0 A rm_dataabort_handler
0x7ffffd0 A rm_irqhandler
0x7ffffe0 A rm_irqhandler2
0x7fffff0 T iap_entry
```

As per the ARM specification (The ARM Thumb Procedure Call Standard SWS ESPC 0002 A-05) up to 4 parameters can be passed in the r0, r1, r2 and r3 registers respectively. Additional parameters are passed on the stack. Up to 4 parameters can be returned in the r0, r1, r2 and r3 registers respectively. Additional parameters are returned indirectly via memory. Some of the IAP calls require more than 4 parameters. If the ARM suggested scheme is used for the parameter passing/returning then it might create problems due to difference in the C compiler implementation from different vendors. The suggested parameter passing scheme reduces such risk.

The flash memory is not accessible during a write or erase operation. IAP commands, which results in a flash write/erase operation, use 32 bytes of space in the top portion of the on-chip RAM for execution. The user program should not be use this space if IAP flash programming is permitted in the application.

Table 214: IAP Command Summary

IAP Command	Command Code	Described in
Prepare sector(s) for write operation	50	Table 215
Copy RAM to Flash	51	Table 216
Erase sector(s)	52	Table 217
Blank check sector(s)	53	Table 218
Read Part ID	54	Table 219
Read Boot code version	55	Table 220
Compare	56	Table 221

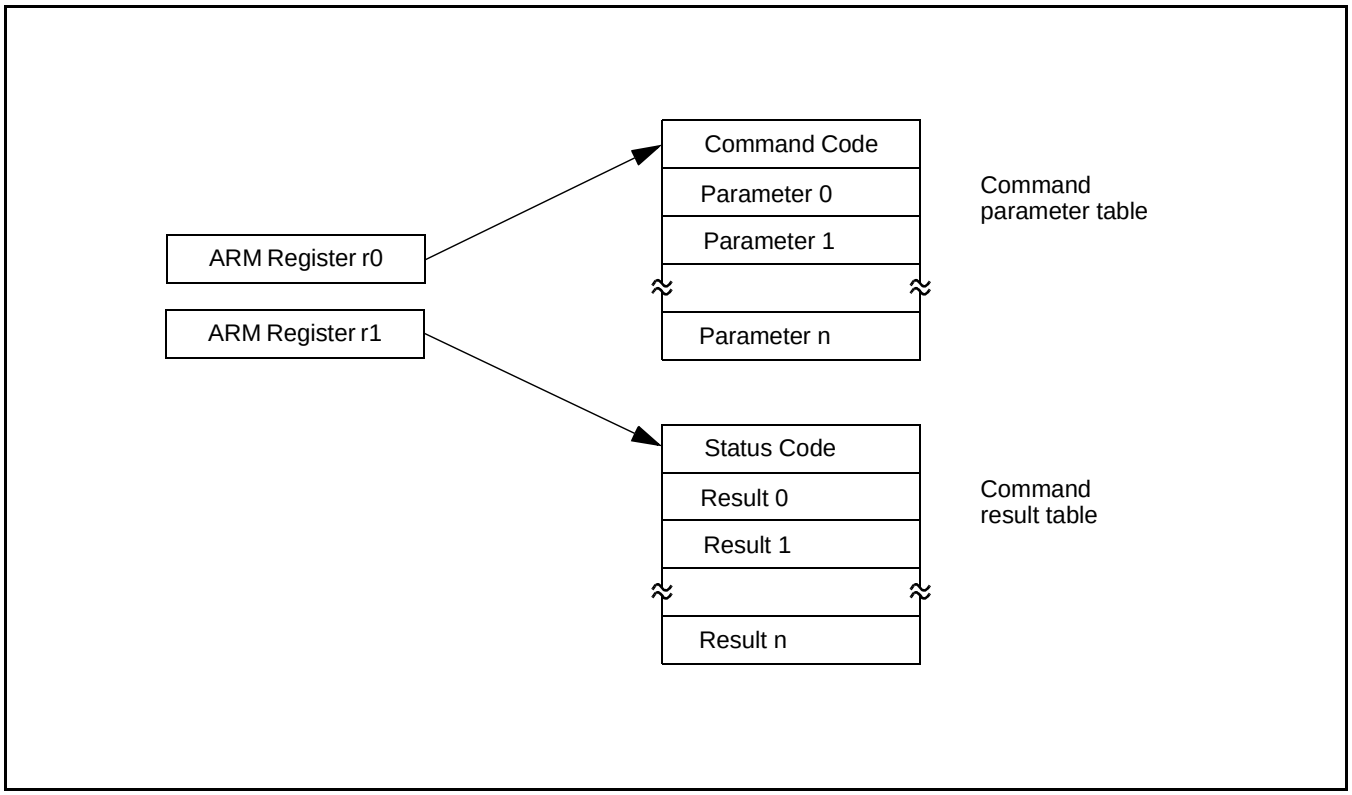


Figure 50: IAP Parameter passing

Prepare sector(s) for write operation

This command makes flash write/erase operation a two step process.

Table 215: IAP Prepare sector(s) for write operation command description

Command	Prepare sector(s) for write operation
Input	Command code: 50 Param0: Start Sector Number Param1: End Sector Number: Should be greater than or equal to start sector number.
Status Code	CMD_SUCCESS | BUSY | INVALID_SECTOR
Result	None
Description	This command must be executed before executing "Copy RAM to Flash" or "Erase Sector(s)" command. Successful execution of the "Copy RAM to Flash" or "Erase Sector(s)" command causes relevant sectors to be protected again. The boot sector can not be prepared by this command. To prepare a single sector use the same "Start" and "End" sector numbers.

Copy RAM to Flash

Table 216: IAP Copy RAM to Flash command description

Command	Copy RAM to Flash
Input	Command code: 51 Param0(DST): Destination Flash address where data bytes are to be written. The destination address should be a 512 byte boundary. Param1(SRC): Source RAM address from which data bytes are to be read. This address should be on word boundary. Param2: Number of bytes to be written. Should be 512 | 1024 | 4096 | 8192. Param3: System Clock Frequency (CCLK) in KHz.
Status Code	CMD_SUCCESS | SRC_ADDR_ERROR (Address not on word boundary) | DST_ADDR_ERROR (Address not on correct boundary) | SRC_ADDR_NOT_MAPPED | DST_ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not 512 | 1024 | 4096 | 8192) | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | BUSY
Result	None
Description	This command is used to program the flash memory. The affected sectors should be prepared first by calling "Prepare Sector for Write Operation" command. The affected sectors are automatically protected again once the copy command is successfully executed. The boot sector can not be written by this command.

Erase Sector(s)

Table 217: IAP Erase Sector(s) command description

Command	Erase Sector(s)
Input	Command code: 52 Param0: Start Sector Number Param1: End Sector Number: Should be greater than or equal to start sector number. Param2: System Clock Frequency (CCLK) in KHz.
Status Code	CMD_SUCCESS | BUSY | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | INVALID_SECTOR
Result	None
Description	This command is used to erase a sector or multiple sectors of on-chip Flash memory. The boot sector can not be erased by this command. To erase a single sector use the same "Start" and "End" sector numbers.

Blank check sector(s)

Table 218: IAP Blank check sector(s) command description

Command	Blank check sector(s)
Input	Command code: 53 Param0: Start Sector Number Param1: End Sector Number: Should be greater than or equal to start sector number.
Status Code	CMD_SUCCESS | BUSY | SECTOR_NOT_BLANK | INVALID_SECTOR
Result	Result0: Offset of the first non blank word location if the Status Code is SECTOR_NOT_BLANK. Result1: Contents of non blank word location.
Description	This command is used to blank check a sector or multiple sectors of on-chip Flash memory. To blank check a single sector use the same "Start" and "End" sector numbers.

Read Part ID

Table 219: IAP Read Part ID command description

Command	Read Part ID
Input	Command Code: 54 parameters: None
Status Code	CMD_SUCCESS
Result	Result0: Part Identification Number
Description	This command is used to read the part identification number.

Read Boot code version

Table 220: IAP Read Boot Code version command description

Command	Read boot code version
Input	Command code: 55 Parameters: None
Status Code	CMD_SUCCESS
Result	Result0: 2 bytes of boot code version number. It is to be interpreted as <byte1(Major)>.<byte0(Minor)>
Description	This command is used to read the boot code version number.

Compare

Table 221: IAP Compare command description

Command	Compare
Input	Command Code: 56 Param0(DST): Starting Flash or RAM address from where data bytes are to be compared. This address should be a word boundary. Param1(SRC): Starting Flash or RAM address from where data bytes are to be compared. This address should be a word boundary. Param2: Number of bytes to be compared. Count should be in multiple of 4.
Status Code	CMD_SUCCESS | COMPARE_ERROR | COUNT_ERROR (Byte count is not multiple of 4) | ADDR_ERROR | ADDR_NOT_MAPPED
Result	Result0: Offset of the first mismatch if the Status Code is COMPARE_ERROR.
Description	This command is used to compare the memory contents at two locations. **Compare result may not be correct when source or destination address contains any of the first 64 bytes starting from address zero. First 64 bytes can be re-mapped to RAM.**

Table 222: IAP Status Codes Summary

Status Code	Mnemonic	Description
0	CMD_SUCCESS	Command is executed successfully.
1	INVALID_COMMAND	Invalid command.
2	SRC_ADDR_ERROR	Source address is not on a word boundary.
3	DST_ADDR_ERROR	Destination address is not on a correct boundary.
4	SRC_ADDR_NOT_MAPPED	Source address is not mapped in the memory map. Count value is taken in to consideration where applicable.
5	DST_ADDR_NOT_MAPPED	Destination address is not mapped in the memory map. Count value is taken in to consideration where applicable.
6	COUNT_ERROR	Byte count is not multiple of 4 or is not a permitted value.
7	INVALID_SECTOR	Sector number is invalid.
8	SECTOR_NOT_BLANK	Sector is not blank.
9	SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION	Command to prepare sector for write operation was not executed.
10	COMPARE_ERROR	Source and destination data is not same.
11	BUSY	Flash programming hardware interface is busy.

JTAG FLASH PROGRAMMING INTERFACE

Debug tools can write parts of the flash image to the RAM and then execute the IAP call "Copy RAM to Flash" repeatedly with proper offset.

21. EMBEDDEDICE LOGIC

FEATURES

- No target resources are required by the software debugger in order to start the debugging session
- Allows the software debugger to talk via a JTAG (Joint Test Action Group) port directly to the core
- Inserts instructions directly in to the ARM7TDMI-S core
- The ARM7TDMI-S core or the System state can be examined, saved or changed depending on the type of instruction inserted
- Allows instructions to execute at a slow debug speed or at a fast system speed

APPLICATIONS

The EmbeddedICE logic provides on-chip debug support. The debugging of the target system requires a host computer running the debugger software and an EmbeddedICE protocol convertor. EmbeddedICE protocol convertor converts the Remote Debug Protocol commands to the JTAG data needed to access the ARM7TDMI-S core present on the target system.

DESCRIPTION

The ARM7TDMI-S Debug Architecture uses the existing JTAG* port as a method of accessing the core. The scan chains that are around the core for production test are reused in the debug state to capture information from the databus and to insert new information into the core or the memory. There are two JTAG-style scan chains within the ARM7TDMI-S. A JTAG-style Test Access Port Controller controls the scan chains. In addition to the scan chains, the debug architecture uses EmbeddedICE logic which resides on chip with the ARM7TDMI-S core. The EmbeddedICE has its own scan chain that is used to insert watchpoints and breakpoints for the ARM7TDMI-S core. The EmbeddedICE logic consists of two real time watchpoint registers, together with a control and status register. One or both of the watchpoint registers can be programmed to halt the ARM7TDMI-S core. Execution is halted when a match occurs between the values programmed into the EmbeddedICE logic and the values currently appearing on the address bus, databus and some control signals. Any bit can be masked so that its value does not affect the comparison. Either watchpoint register can be configured as a watchpoint (i.e. on a data access) or a break point (i.e. on an instruction fetch). The watchpoints and breakpoints can be combined such that:

- The conditions on both watchpoints must be satisfied before the ARM7TDMI core is stopped. The CHAIN functionality requires two consecutive conditions to be satisfied before the core is halted. An example of this would be to set the first breakpoint to trigger on an access to a peripheral and the second to trigger on the code segment that performs the task switching. Therefore when the breakpoints trigger the information regarding which task has switched out will be ready for examination.
- The watchpoints can be configured such that a range of addresses are enabled for the watchpoints to be active. The RANGE function allows the breakpoints to be combined such that a breakpoint is to occur if an access occurs in the bottom 256 bytes of memory but not in the bottom 32 bytes.

The ARM7TDMI-S core has a Debug Communication Channel function in-built. The debug communication channel allows a program running on the target to communicate with the host debugger or another separate host without stopping the program flow or even entering the debug state. The debug communication channel is accessed as a co-processor 14 by the program running on the ARM7TDMI-S core. The debug communication channel allows the JTAG port to be used for sending and receiving data without affecting the normal program flow. The debug communication channel data and control registers are mapped in to addresses in the EmbeddedICE logic.

* For more details refer to IEEE Standard 1149.1 - 1990 Standard Test Access Port and Boundary Scan Architecture.

PIN DESCRIPTION

Table 223: EmbeddedICE Pin Description

Pin Name	Type	Description
TMS	Input	**Test Mode Select.** The TMS pin selects the next state in the TAP state machine.
TCK	Input	**Test Clock.** This allows shifting of the data in, on the TMS and TDI pins. It is a positive edge-triggered clock with the TMS and TCK signals that define the internal state of the device.
TDI	Input	**Test Data In.** This is the serial data input for the shift register.
TDO	Output	**Test Data Output.** This is the serial data output from the shift register. Data is shifted out of the device on the negative edge of the TCK signal
nTRST	Input	**Test Reset.** The nTRST pin can be used to reset the test logic within the EmbeddedICE logic.
RTCK	Output	**Returned Test Clock.** Extra signal added to the JTAG port. Required for designs based on ARM7TDMI-S processor core. Multi-ICE (Development system from ARM) uses this signal to maintain synchronization with targets having slow or widely varying clock frequency. For details refer to "Multi-ICE System Design considerations Application Note 72 (ARM DAI 0072A)".

RESET STATE OF MULTIPLEXED PINS

On the LPC2119/2129/2194/2292/2294, the pins above are multiplexed with P1.31-26. To have them come up as a Debug port, connect a weak bias resistor (4.7 kΩ) between VSS and the P1.26/RTCK pin. To have them come up as GPIO pins, do not connect a bias resistor, and ensure that any external driver connected to P1.26/RTCK is either driving high, or is in high-impedance state, during Reset.

REGISTER DESCRIPTION

The EmbeddedICE logic contains 16 registers as shown in Table 224. below. The ARM7TDMI-S debug architecture is described in detail in "ARM7TDMI-S (rev 4) Technical Reference Manual" (ARM DDI 0234A) published by ARM Limited and is available via Internet at http://www.arm.com.

Table 224: EmbeddedICE Logic Registers

Name	Width	Description	Address
Debug Control	6	Force debug state, disable interrupts	00000
Debug Status	5	Status of debug	00001
Debug Comms Control Register	32	Debug communication control register	00100
Debug Comms Data Register	32	Debug communication data register	00101
Watchpoint 0 Address Value	32	Holds watchpoint 0 address value	01000
Watchpoint 0 Address Mask	32	Holds watchpoint 0 address mask	01001
Watchpoint 0 Data Value	32	Holds watchpoint 0 data value	01010
Watchpoint 0 Data Mask	32	Holds watchpoint 0 data Mask	01011
Watchpoint 0 Control Value	9	Holds watchpoint 0 control value	01100
Watchpoint 0 Control Mask	8	Holds watchpoint 0 control mask	01101
Watchpoint 1 Address Value	32	Holds watchpoint 1 address value	10000
Watchpoint 1 Address Mask	32	Holds watchpoint 1 address mask	10001
Watchpoint 1 Data Value	32	Holds watchpoint 1 data value	10010
Watchpoint 1 Data Mask	32	Holds watchpoint 1 data Mask	10011
Watchpoint 1 Control Value	9	Holds watchpoint 1 control value	10100
Watchpoint 1 Control Mask	8	Holds watchpoint 1 control mask	10101

BLOCK DIAGRAM

The block diagram of the debug environment is shown below in Figure 51.

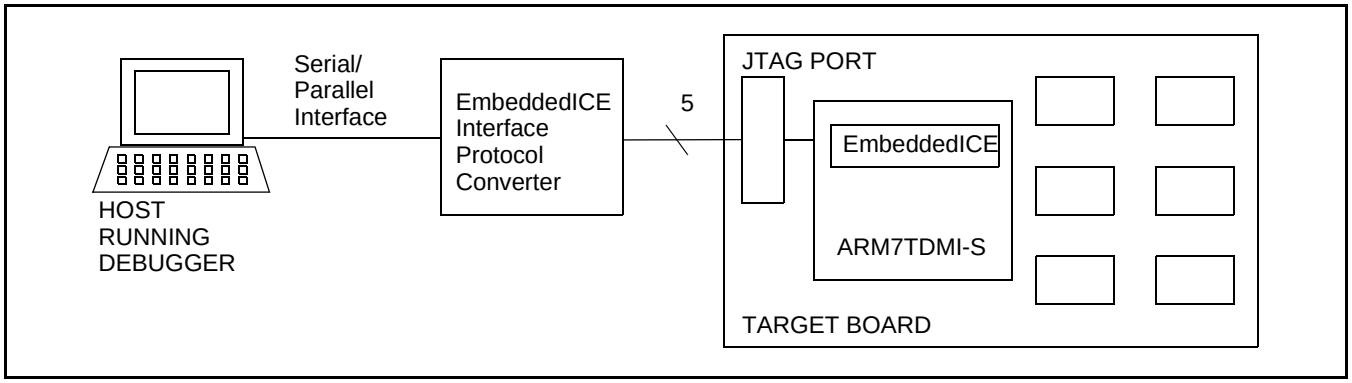


Figure 51: EmbeddedICE Debug Environment Block Diagram

22. EMBEDDED TRACE MACROCELL

FEATURES

- Closely track the instructions that the ARM core is executing
- 10 pin interface
- 1 External trigger input
- All registers are programmed through JTAG interface
- Does not consume power when trace is not being used
- THUMB instruction set support

APPLICATIONS

As the microcontroller has significant amounts of on-chip memories, it is not possible to determine how the processor core is operating simply by observing the external pins. The ETM provides real-time trace capability for deeply embedded processor cores. It outputs information about processor execution to a trace port. A software debugger allows configuration of the ETM using a JTAG interface and displays the trace information that has been captured, in a format that a user can easily understand.

DESCRIPTION

The ETM is connected directly to the ARM core and not to the main AMBA system bus. It compresses the trace information and exports it through a narrow trace port. An external Trace Port Analyzer captures the trace information under software debugger control. Trace port can broadcast the Instruction trace information. Instruction trace (or PC trace) shows the flow of execution of the processor and provides a list of all the instructions that were executed. Instruction trace is significantly compressed by only broadcasting branch addresses as well as a set of status signals that indicate the pipeline status on a cycle by cycle basis. Trace information generation can be controlled by selecting the trigger resource. Trigger resources include address comparators, counters and sequencers. Since trace information is compressed the software debugger requires a static image of the code being executed. Self-modifying code can not be traced because of this restriction.

ETM Configuration

The following standard configuration is selected for the ETM macrocell.

Table 225: ETM Configuration

Resource number/type	Small [1]
Pairs of address comparators	1
Data Comparators	0 (Data tracing is not supported)
Memory Map Decoders	4
Counters	1
Sequencer Present	No
External Inputs	2

Table 225: ETM Configuration

Resource number/type	Small [1]
External Outputs	0
FIFOFULL Present	Yes (Not wired)
FIFO depth	10 bytes
Trace Packet Width	4/8

1. For details refer to ARM documentation "Embedded Trace Macrocell Specification (ARM IHI 0014E)".

PIN DESCRIPTION

Table 226: ETM Pin Description

Pin Name	Type	Description
TRACECLK	Output	**Trace Clock.** The trace clock signal provides the clock for the trace port. PIPESTAT[2:0], TRACESYNC, and TRACEPKT[3:0] signals are referenced to the rising edge of the trace clock. This clock is not generated by the ETM block. It is to be derived from the system clock. The clock should be balanced to provide sufficient hold time for the trace data signals. Half rate clocking mode is supported. Trace data signals should be shifted by a clock phase from TRACECLK. Refer to Figure 3.14 page 3.26 and figure 3.15 page 3.27 in "ETM7 Technical Reference Manual" (ARM DDI 0158B), for example circuits that implements both half-rate-clocking and shifting of the trace data with respect to the clock. For TRACECLK timings refer to section 5.2 on page 5-13 in "Embedded Trace Macrocell Specification" (ARM IHI 0014E).
PIPESTAT[2:0]	Output	**Pipe Line status.** The pipeline status signals provide a cycle-by-cycle indication of what is happening in the execution stage of the processor pipeline.
TRACESYNC	Output	**Trace synchronization.** The trace sync signal is used to indicate the first packet of a group of trace packets and is asserted HIGH only for the first packet of any branch address.
TRACEPKT[3:0]	Output	**Trace Packet.** The trace packet signals are used to output packaged address and data information related to the pipeline status. All packets are eight bits in length. A packet is output over two cycles. In the first cycle, Packet[3:0] is output and in the second cycle, Packet[7:4] is output.
EXTIN[0]	Input	External Trigger Input.

RESET STATE OF MULTIPLEXED PINS

On the LPC2119/2129/2194/2292/2294, the ETM pin functions are multiplexed with P1.25-16. To have these pins come as a Trace port, connect a weak bias resistor (4.7 kΩ) between the P1.20/TRACESYNC pin and V_{SS}. To have them come up as port pins, do not connect a bias resistor to P1.20/TRACESYNC, and ensure that any external driver connected to P1.20/TRACESYNC is either driving high, or is in high-impedance state, during Reset.

REGISTER DESCRIPTION

The ETM contains 29 registers as shown in Table 227. below. They are described in detail in the ARM IHI 0014E document published by ARM Limited, which is available via the Internet at http://www.arm.com.

Table 227: ETM Registers

Name	Description	Access	Register encoding
ETM Control	Controls the general operation of the ETM	Read/Write	000 0000
ETM Configuration Code	Allows a debugger to read the number of each type of resource	Read Only	000 0001
Trigger Event	Holds the controlling event	Write Only	000 0010
Memory Map Decode Control	Eight-bit register, used to statically configure the memory map decoder	Write Only	000 0011
ETM Status	Holds the pending overflow status bit	Read Only	000 0100
System Configuration	Holds the configuration information using the SYSOPT bus	Read Only	000 0101
Trace Enable Control 3	Holds the trace on/off addresses	Write Only	000 0110
Trace Enable Control 2	Holds the address of the comparison	Write Only	000 0111
Trace Enable Event	Holds the enabling event	Write Only	000 1000
Trace Enable Control 1	Holds the include and exclude regions	Write Only	000 1001
FIFOFULL Region	Holds the include and exclude regions	Write Only	000 1010
FIFOFULL Level	Holds the level below which the FIFO is considered full	Write Only	000 1011
ViewData event	Holds the enabling event	Write Only	000 1100
ViewData Control 1	Holds the include/exclude regions	Write Only	000 1101
ViewData Control 2	Holds the include/exclude regions	Write Only	000 1110
ViewData Control 3	Holds the include/exclude regions	Write Only	000 1111
Address Comparator 1 to 16	Holds the address of the comparison	Write Only	001 xxxx
Address Access Type 1 to 16	Holds the type of access and the size	Write Only	010 xxxx
reserved	-	-	000 xxxx
reserved	-	-	100 xxxx
Initial Counter Value 1 to 4	Holds the initial value of the counter	Write Only	101 00xx
Counter Enable 1 to 4	Holds the counter clock enable control and event	Write Only	101 01xx
Counter reload 1 to 4	Holds the counter reload event	Write Only	101 10xx
Counter Value 1 to 4	Holds the current counter value	Read Only	101 11xx
Sequencer State and Control	Holds the next state triggering events.	-	110 00xx
External Output 1 to 4	Holds the controlling events for each output	Write Only	110 10xx
Reserved	-	-	110 11xx
Reserved	-	-	111 0xxx
Reserved	-	-	111 1xxx

BLOCK DIAGRAM

The block diagram of the ETM debug environment is shown below in Figure 52.

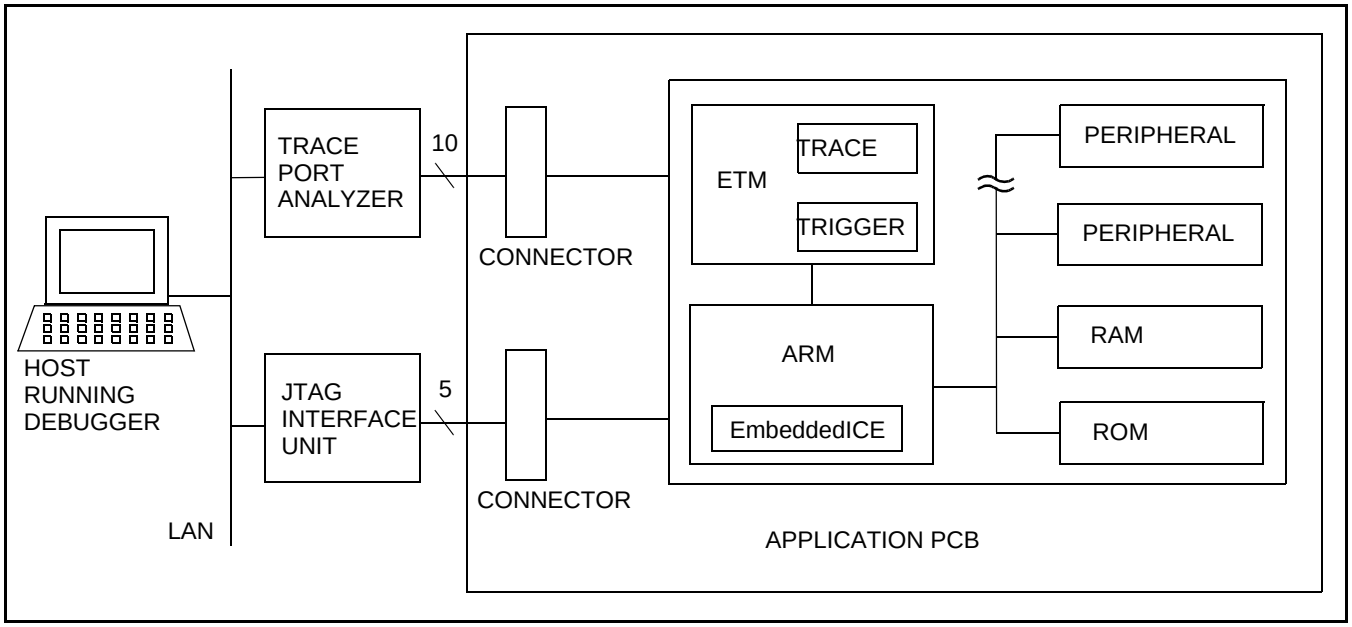


Figure 52: ETM Debug Environment Block Diagram

23. REALMONITOR

RealMonitor is a configurable software module which enables real time debug. RealMonitor is developed by ARM Inc. Information presented in this chapter is taken from the ARM document *RealMonitor Target Integration Guide (ARM DUI 0142A).* It applies to a specific configuration of RealMonitor software programmed in the on-chip flash memory of this device.

Refer to the white paper *"Real Time Debug for System-on-Chip"* available at *http://www.arm.com/support/White_Papers?OpenDocument* for background information.

FEATURES

- Allows user to establish a debug session to a currently running system without halting or resetting the system.
- Allows user time-critical interrupt code to continue executing while other user application code is being debugged.

APPLICATIONS

Real time debugging.

DESCRIPTION

RealMonitor is a lightweight debug monitor that allows interrupts to be serviced while user debug their foreground application. It communicates with the host using the DCC (Debug Communications Channel), which is present in the EmbeddedICE logic. RealMonitor provides advantages over the traditional methods for debugging applications in ARM systems. The traditional methods include:

- Angel (a target-based debug monitor).
- Multi-ICE or other JTAG unit and EmbeddedICE logic (a hardware-based debug solution).

Although both of these methods provide robust debugging environments, neither is suitable as a lightweight real-time monitor.

Angel is designed to load and debug independent applications that can run in a variety of modes, and communicate with the debug host using a variety of connections (such as a serial port or ethernet). Angel is required to save and restore full processor context, and the occurrence of interrupts can be delayed as a result. Angel, as a fully functional target-based debugger, is therefore too heavyweight to perform as a real-time monitor.

Multi-ICE is a hardware debug solution that operates using the EmbeddedICE unit that is built into most ARM processors. To perform debug tasks such as accessing memory or the processor registers, Multi-ICE must place the core into a debug state. While the processor is in this state, which can be millions of cycles, normal program execution is suspended, and interrupts cannot be serviced.

RealMonitor combines features and mechanisms from both Angel and Multi-ICE to provide the services and functions that are required. In particular, it contains both the Multi-ICE communication mechanisms (the DCC using JTAG), and Angel-like support for processor context saving and restoring. RealMonitor is pre-programmed in the on-chip Flash memory (boot sector). When enabled It allows user to observe and debug while parts of application continue to run. Refer to section How to Enable RealMonitor for details.

RealMonitor Components

As shown in Figure 53, RealMonitor is split in to two functional components:

RMHost

This is located between a debugger and a JTAG unit. The RMHost controller, RealMonitor.dll, converts generic *Remote Debug Interface* (RDI) requests from the debugger into DCC-only RDI messages for the JTAG unit. For complete details on debugging a RealMonitor-integrated application from the host, see the *ARM RMHost User Guide (ARM DUI 0137A).*

RMTarget

This is pre-programmed in the on-chip Flash memory (boot sector), and runs on the target hardware. It uses the EmbeddedICE logic, and communicates with the host using the DCC. For more details on RMTarget functionality, see the *RealMonitor Target Integration Guide (ARM DUI 0142A).*

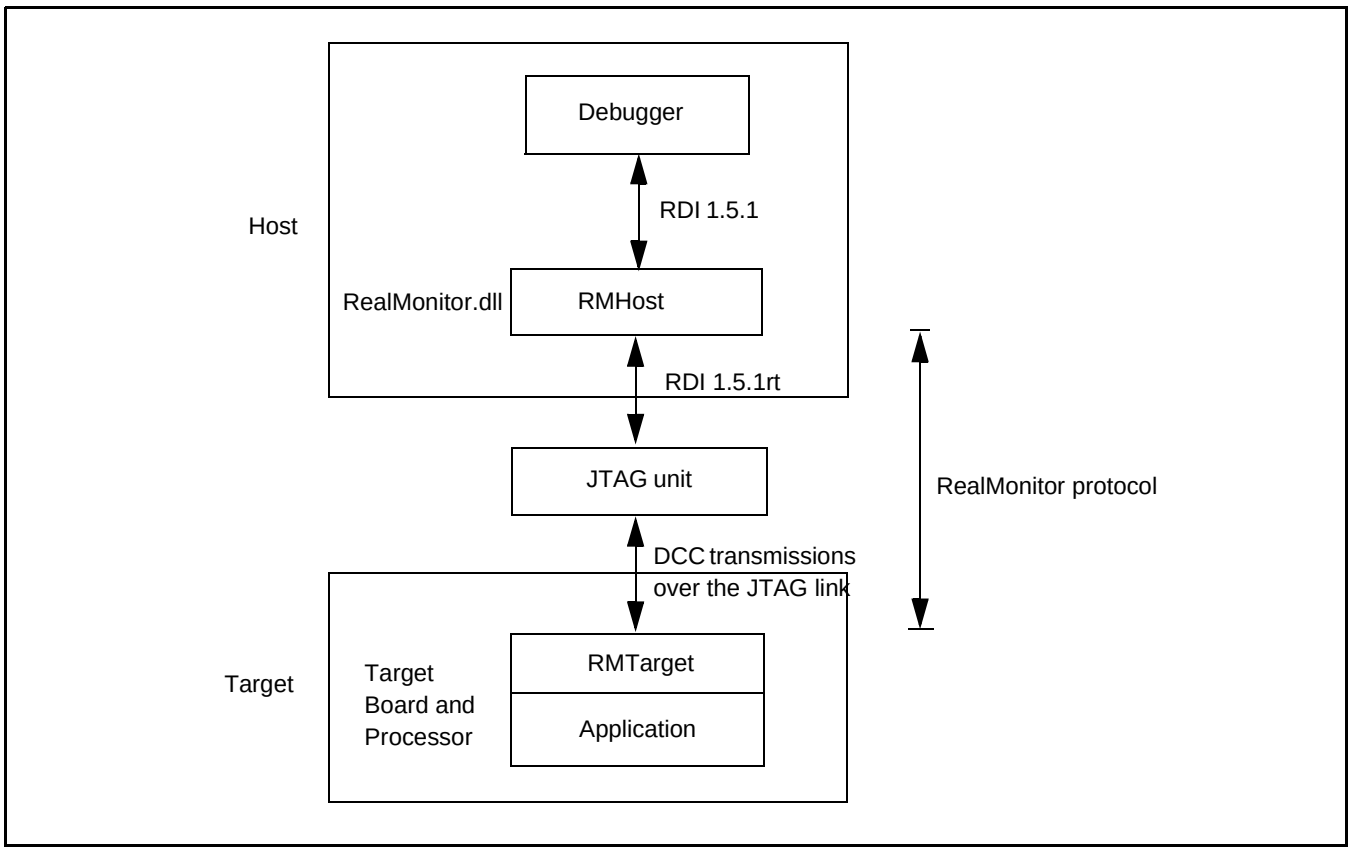


Figure 53: RealMonitor components

How RealMonitor works

In general terms, the RealMonitor operates as a state machine, as shown in Figure 54. RealMonitor switches between running and stopped states, in response to packets received by the host, or due to asynchronous events on the target. RMTarget supports the triggering of only one breakpoint, watchpoint, stop, or semihosting SWI at a time. There is no provision to allow nested events to be saved and restored. So, for example, if user application has stopped at one breakpoint, and another breakpoint occurs in an IRQ handler, RealMonitor enters a panic state. No debugging can be performed after RealMonitor enters this state.

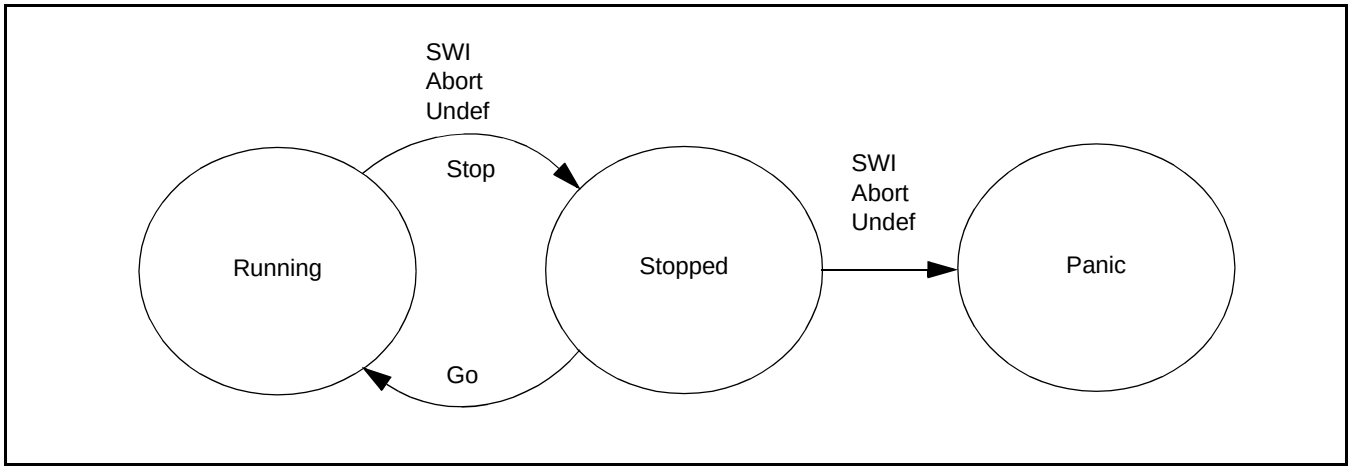


Figure 54: RealMonitor as a state machine

A debugger such as the ARM eXtended Debugger (AXD) or other RealMonitor aware debugger, that runs on a host computer, can connect to the target to send commands and receive data. This communication between host and target is illustrated in Figure 53.

The target component of RealMonitor, RMTarget, communicates with the host component, RMHost, using the Debug Communications Channel (DCC), which is a reliable link whose data is carried over the JTAG connection.

While user application is running, RMTarget typically uses IRQs generated by the DCC. This means that if user application also wants to use IRQs, it must pass any DCC-generated interrupts to RealMonitor.

To allow nonstop debugging, the EmbeddedICE-RT logic in the processor generates a Prefetch Abort exception when a breakpoint is reached, or a Data Abort exception when a watchpoint is hit. These exceptions are handled by the RealMonitor exception handlers that inform the user, by way of the debugger, of the event. This allows user application to continue running without stopping the processor. RealMonitor considers user application to consist of two parts:

- a foreground application running continuously, typically in User, System, or SVC mode
- a background application containing interrupt and exception handlers that are triggered by certain events in user system, including:
  - IRQs or FIQs
  - Data and Prefetch aborts caused by user foreground application. This indicates an error in the application being debugged. In both cases the host is notified and the user application is stopped.
  - Undef exception caused by the undefined instructions in user foreground application. This indicates an error in the application being debugged. RealMonitor stops the user application until a "Go" packet is received from the host.

When one of these exceptions occur that is not handled by user application, the following happens:

- RealMonitor enters a loop, polling the DCC. If the DCC read buffer is full, control is passed to rm_ReceiveData() (RealMonitor internal function). If the DCC write buffer is free, control is passed to rm_TransmitData() (RealMonitor internal function). If there is nothing else to do, the function returns to the caller. The ordering of the above comparisons gives reads from the DCC a higher priority than writes to the communications link.
- RealMonitor stops the foreground application. Both IRQs and FIQs continue to be serviced if they were enabled by the application at the time the foreground application was stopped.

HOW TO ENABLE REALMONITOR

The following steps must be performed to enable RealMonitor. A code example which implements all the steps can be found at the end of this section.

Adding stacks

User must ensure that stacks are set up within application for each of the processor modes used by RealMonitor. For each mode, RealMonitor requires a fixed number of words of stack space. User must therefore allow sufficient stack space for both RealMonitor and application.

RealMonitor has the following stack requirements:

Table 228: RealMonitor stack requirement

Processor Mode	RealMonitor Stack Usage (Bytes)
Undef	48
Prefetch Abort	16
Data Abort	16
IRQ	8

IRQ mode

A stack for this mode is always required. RealMonitor uses two words on entry to its interrupt handler. These are freed before nested interrupts are enabled.

Undef mode

A stack for this mode is always required. RealMonitor uses 12 words while processing an undefined instruction exception.

SVC mode

RealMonitor makes no use of this stack.

Prefetch Abort mode

RealMonitor uses four words on entry to its Prefetch abort interrupt handler.

Data Abort mode

RealMonitor uses four words on entry to its data abort interrupt handler.

User/System mode

RealMonitor makes no use of this stack.

FIQ mode

RealMonitor makes no use of this stack.

Handling exceptions

This section describes the importance of sharing exception handlers between RealMonitor and user application.

RealMonitor exception handling

To function properly, RealMonitor must be able to intercept certain interrupts and exceptions. Figure 55 illustrates how exceptions can be claimed by RealMonitor itself, or shared between RealMonitor and application. If user application requires the exception sharing, they must provide function (such as *app_IRQDispatch* ()). Depending on the nature of the exception, this handler can either:

- pass control to the RealMonitor processing routine, such as rm_irqhandler2()
- claim the exception for the application itself, such as *app_IRQHandler* ().

In a simple case where an application has no exception handlers of its own, the application can install the RealMonitor low-level exception handlers directly into the vector table of the processor. Although the irq handler must get the address of the Vectored Interrupt Controller. The easiest way to do this is to write a branch instruction (<*address* >) into the vector table, where the target of the branch is the start address of the relevant RealMonitor exception handler.

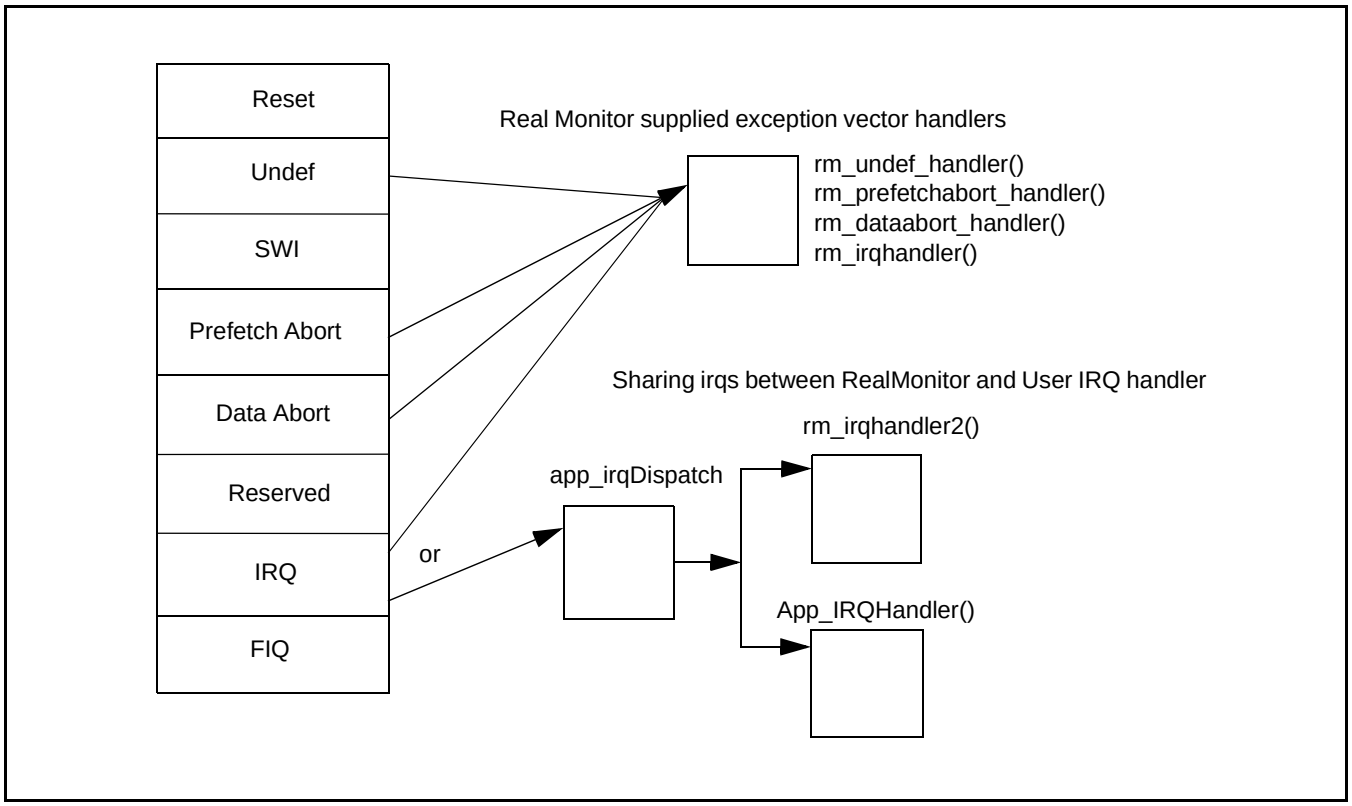


Figure 55: Exception Handlers

RMTarget initialization

While the processor is in a privileged mode, and IRQs are disabled, user must include a line of code within the start-up sequence of application to call rm_init_entry().

Code Example

The following example shows how to setup stack, VIC, initialize RealMonitor and share non vectored interrupts:

```
    IMPORT rm_init_entry
    IMPORT rm_prefetchabort_handler
    IMPORT rm_dataabort_handler
    IMPORT rm_irqhandler2
    IMPORT rm_undef_handler
    IMPORT User_Entry ;Entry point of user application.
    CODE32
    ENTRY
   ;Define exception table. Instruct linker to place code at address 0x0000 0000

    AREA exception_table, CODE

    LDR pc, Reset_Address
    LDR pc, Undefined_Address
    LDR pc, SWI_Address
    LDR pc, Prefetch_Address
    LDR pc, Abort_Address
    NOP ; Insert User code valid signature here.
    LDR pc, [pc, #-0xFF0] ;Load IRQ vector from VIC
    LDR PC, FIQ_Address

Reset_Address      DCD     __init                   ;Reset Entry point
Undefined_Address  DCD     rm_undef_handler         ;Provided by RealMonitor
SWI_Address        DCD     0                        ;User can put address of SWI handler here
Prefetch_Address   DCD     rm_prefetchabort_handler ;Provided by RealMonitor
Abort_Address      DCD     rm_dataabort_handler     ;Provided by RealMonitor
FIQ_Address        DCD     0                        ;User can put address of FIQ handler here

AREA init_code, CODE

ram_end EQU 0x4000xxxx ; Top of on-chip RAM.
__init
; /*********************************************************************
; * Set up the stack pointers for various processor modes. Stack grows
; * downwards.
; *********************************************************************/
    LDR r2, =ram_end    ;Get top of RAM
    MRS r0, CPSR        ;Save current processor mode
    ; Initialize the Undef mode stack for RealMonitor use
    BIC     r1, r0, #0x1f
    ORR     r1, r1, #0x1b
    MSR     CPSR_c, r1
    ;Keep top 32 bytes for flash programming routines.
    ;Refer to Flash Memory System and Programming chapter
    SUB sp,r2,#0x1F

    ; Initialize the Abort mode stack for RealMonitor
    BIC     r1, r0, #0x1f
    ORR     r1, r1, #0x17
    MSR     CPSR_c, r1
    ;Keep 64 bytes for Undef mode stack
    SUB sp,r2,#0x5F
```

```
    ; Initialize the IRQ mode stack for RealMonitor and User
    BIC     r1, r0, #0x1f
    ORR     r1, r1, #0x12
    MSR     CPSR_c, r1
    ;Keep 32 bytes for Abort mode stack
    SUB sp,r2,#0x7F

    ; Return to the original mode.
    MSR     CPSR_c, r0

    ; Initialize the stack for user application
    ; Keep 256 bytes for IRQ mode stack
    SUB sp,r2,#0x17F

; /*********************************************************************
; * Setup Vectored Interrupt controller. DCC Rx and Tx interrupts
; * generate Non Vectored IRQ request. rm_init_entry is aware
; * of the VIC and it enables the DBGCommRX and DBGCommTx interrupts.
; * Default vector address register is programmed with the address of
; * Non vectored app_irqDispatch mentioned in this example. User can setup
; * Vectored IRQs or FIQs here.
; *********************************************************************/

    VICBaseAddr          EQU 0xFFFFF000    ; VIC Base address
    VICDefVectAddrOffset EQU 0x34

    LDR r0, =VICBaseAddr
    LDR r1, =app_irqDispatch
    STR r1, [r0,#VICDefVectAddrOffset]

    BL rm_init_entry ;Initialize RealMonitor
    ;enable FIQ and IRQ in ARM Processor
    MRS     r1, CPSR        ; get the CPSR
    BIC     r1, r1, #0xC0   ; enable IRQs and FIQs
    MSR     CPSR_c, r1      ; update the CPSR
; /*********************************************************************
; * Get the address of the User entry point.
; *********************************************************************/
    LDR lr, =User_Entry
    MOV pc, lr
; /*********************************************************************
; * Non vectored irq handler (app_irqDispatch)
; *********************************************************************/

AREA app_irqDispatch, CODE
VICVectAddrOffset EQU 0x30
app_irqDispatch

    ;enable interrupt nesting
    STMFD sp!, {r12,r14}
    MRS r12, spsr             ;Save SPSR in to r12
    MSR cpsr_c,0x1F           ;Re-enable IRQ, go to system mode

    ;User should insert code here if non vectored Interrupt sharing is
    ;required. Each non vectored shared irq handler must return to
    ;the interrupted instruction by using the following code.
    ;MSR cpsr_c, #0x52              ;Disable irq, move to IRQ mode
```

```
;MSR spsr, r12                  ;Restore SPSR from r12
;STMFD sp!, {r0}
;LDR r0, =VICBaseAddr
;STR r1, [r0,#VICVectAddrOffset] ;Acknowledge Non Vectored irq has finished
;LDMFD sp!, {r12,r14,r0}         ;Restore registers
;SUBS pc, r14, #4                ;Return to the interrupted instruction

;user interrupt did not happen so call rm_irqhandler2. This handler
;is not aware of the VIC interrupt priority hardware so trick
;rm_irqhandler2 to return here

STMFD sp!, {ip,pc}
LDR pc, rm_irqhandler2
;rm_irqhandler2 returns here
MSR cpsr_c, #0x52               ;Disable irq, move to IRQ mode
MSR spsr, r12                   ;Restore SPSR from r12
STMFD sp!, {r0}
LDR r0, =VICBaseAddr
STR r1, [r0,#VICVectAddrOffset] ;Acknowledge Non Vectored irq has finished
LDMFD sp!, {r12,r14,r0}         ;Restore registers
SUBS pc, r14, #4                ;Return to the interrupted instruction

END
```

REALMONITOR BUILD OPTIONS

RealMonitor was built with the following options:

RM_OPT_DATALOGGING=FALSE

This option enables or disables support for any target-to-host packets sent on a non RealMonitor (third-party) channel.

RM_OPT_STOPSTART=TRUE

This option enables or disables support for all stop and start debugging features.

RM_OPT_SOFTBREAKPOINT=TRUE

This option enables or disables support for software breakpoints.

RM_OPT_HARDBREAKPOINT=TRUE

Enabled for cores with EmbeddedICE-RT. This device uses ARM-7TDMI-S Rev 4 with EmbeddedICE-RT.

RM_OPT_HARDWATCHPOINT=TRUE

Enabled for cores with EmbeddedICE-RT. This device uses ARM-7TDMI-S Rev 4 with EmbeddedICE-RT.

RM_OPT_SEMIHOSTING=FALSE

This option enables or disables support for SWI semi-hosting. Semi-hosting provides code running on an ARM target use of facilities on a host computer that is running an ARM debugger. Examples of such facilities include the keyboard input, screen output, and disk I/O.

RM_OPT_SAVE_FIQ_REGISTERS=TRUE

This option determines whether the FIQ-mode registers are saved into the registers block when RealMonitor stops.

RM_OPT_READBYTES=TRUE

RM_OPT_WRITEBYTES=TRUE

RM_OPT_READHALFWORDS=TRUE

RM_OPT_WRITEHALFWORDS=TRUE

RM_OPT_READWORDS=TRUE

RM_OPT_WRITEWORDS=TRUE

Enables/Disables support for 8/16/32 bit read/write.

RM_OPT_EXECUTECODE=FALSE

Enables/Disables support for executing code from "execute code" buffer. The code must be downloaded first.

RM_OPT_GETPC=TRUE

This option enables or disables support for the RealMonitor GetPC packet. Useful in code profiling when real monitor is used in interrupt mode.

RM_EXECUTECODE_SIZE=NA

"execute code" buffer size. Also refer to RM_OPT_EXECUTECODE option.

RM_OPT_GATHER_STATISTICS=FALSE

This option enables or disables the code for gathering statistics about the internal operation of RealMonitor.

RM_DEBUG=FALSE

This option enables or disables additional debugging and error-checking code in RealMonitor.

RM_OPT_BUILDIDENTIFIER=FALSE

This option determines whether a build identifier is built into the capabilities table of RMTarget. Capabilities table is stored in ROM.

RM_OPT_SDM_INFO=FALSE

SDM gives additional information about application board and processor to debug tools.

RM_OPT_MEMORYMAP=FALSE

This option determines whether a memory map of the board is built into the target and made available through the capabilities table

RM_OPT_USE_INTERRUPTS=TRUE

This option specifies whether RMTarget is built for interrupt-driven mode or polled mode.

RM_FIFOSIZE=NA

This option specifies the size, in words, of the data logging FIFO buffer.

CHAIN_VECTORS=FALSE

This option allows RMTarget to support vector chaining through µHAL (ARM HW abstraction API).

Purchase of Philips I²C components conveys a license under the Philips' I²C patent to use the components in the I²C system provided the system conforms to the I²C specifications defined by Philips. This specification can be ordered using the code 9398 393 40011.

Definitions

Short-form specification — The data in a short-form specification is extracted from a full data sheet with the same type number and title. For detailed information see the relevant data sheet or data handbook.

Limiting values definition — Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 60134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.

Application information — Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Disclaimers

Life support — These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes — Philips Semiconductors reserves the right to make changes in the products—including circuits, standard cells, and/or software—described or contained herein in order to improve design and/or performance. When the product is in full production (status 'Production'), relevant changes will be communicated via a Customer Product/Process Change Notification (CPCN). Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no license or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Contact information

For additional information please visit
http://www.semiconductors.philips.com. Fax: **+31 40 27 24825**

For sales offices addresses send e-mail to:
sales.addresses@www.semiconductors.philips.com.

Date of release: 5-04

Document order number: 9397 750 13262

Let's make things better.

Philips Semiconductors

PHILIPS