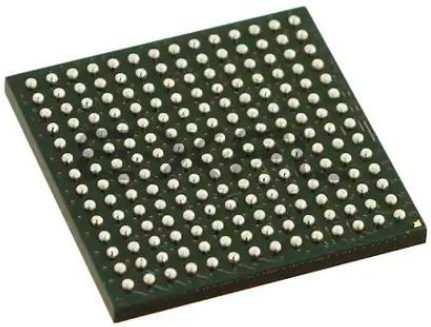

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	Coldfire V2
Core Size	32-Bit Single-Core
Speed	66MHz
Connectivity	EBI/EMI, Ethernet, I²C, SPI, UART/USART, USB
Peripherals	DMA, WDT
Number of I/O	32
Program Memory Size	16KB (4K x 32)
Program Memory Type	ROM
EEPROM Size	-
RAM Size	1K x 32
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	196-LBGA
Supplier Device Package	196-LBGA (15x15)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcf5272vm66r2j

Table xvi. USB Module Memory Map (continued)

MBAR Offset	Register Name	Old Mnemonic	New Mnemonic
0x1013	USB Function Address Register	USBFAR	**FAR**
0x1014	USB Alternate Setting Register	USBASR	**ASR**
0x1018	USB Device Request Data1 Register	USBDRR1	**DRR1**
0x101C	USB Device Request Data2 Register	USBDRR2	**DRR2**
0x1022	USB Specification Number Register	USBSPECR	**SPECR**
0x1026	USB Endpoint 0 Status Register	USBEPSR0	**EP0SR**
0x1028	USB Endpoint 0 IN Config Register	USBEPICFG0	**IEP0CFG**
0x102C	USB Endpoint 0 OUT Config Register	USBEPOCFG0	**OEP0CFG**
0x1030	USB Endpoint 1 Configuration Register	USBEPCFG1	**EP1CFG**
0x1034	USB Endpoint 2 Configuration Register	USBEPCFG2	**EP2CFG**
0x1038	USB Endpoint 3 Configuration Register	USBEPCFG3	**EP3CFG**
0x103C	USB Endpoint 4 Configuration Register	USBEPCFG4	**EP4CFG**
0x1040	USB Endpoint 5 Configuration Register	USBEPCFG5	**EP5CFG**
0x1044	USB Endpoint 6 Configuration Register	USBEPCFG6	**EP6CFG**
0x1048	USB Endpoint 7 Configuration Register	USBEPCFG7	**EP7CFG**
0x104C	USB Endpoint 0 Control Register	USBEPCTL0	**EP0CTL**
0x1052	USB Endpoint 1 Control Register	USBEPCTL1	**EP1CTL**
0x1056	USB Endpoint 2 Control Register	USBEPCTL2	**EP2CTL**
0x105A	USB Endpoint 3 Control Register	USBEPCTL3	**EP3CTL**
0x105E	USB Endpoint 4 Control Register	USBEPCTL4	**EP4CTL**
0x1062	USB Endpoint 5 Control Register	USBEPCTL5	**EP5CTL**
0x1066	USB Endpoint 6 Control Register	USBEPCTL6	**EP6CTL**
0x106A	USB Endpoint 7 Control Register	USBEPCTL7	**EP7CTL**
0x106C	USB General/Endpoint 0 Interrupt Status Register	USBEPISR0	**EP0ISR**
0x1072	USB Endpoint 1 Interrupt Status Register	USBEPISR1	**EP1ISR**
0x1076	USB Endpoint 2 Interrupt Status Register	USBEPISR2	**EP2ISR**
0x107A	USB Endpoint 3 Interrupt Status Register	USBEPISR3	**EP3ISR**
0x107E	USB Endpoint 4 Interrupt Status Register	USBEPISR4	**EP4ISR**
0x1082	USB Endpoint 5 Interrupt Status Register	USBEPISR5	**EP5ISR**
0x1086	USB Endpoint 6 Interrupt Status Register	USBEPISR6	**EP6ISR**
0x108A	USB Endpoint 7 Interrupt Status Register	USBEPISR7	**EP7ISR**
0x108C	USB Endpoint 0 Interrupt Mask Register	USBEPIMR0	**EP0IMR**

Chapter 2
ColdFire Core

This chapter provides an overview of the microprocessor core of the MCF5272. The chapter describes the V2 programming model as it is implemented on the MCF5272. It also includes a full description of exception handling, data formats, an instruction set summary, and a table of instruction timings.

2.1 Features and Enhancements

The MCF5272 is the most highly-integrated V2 standard product, containing a variety of communications and general-purpose peripherals. The V2 core was designed to maximize code density and performance while minimizing die area.

The following list summarizes MCF5272 features:

- Variable-length RISC Version 2 microprocessor core
- Two independent, decoupled pipelines—two-stage instruction fetch pipeline (IFP) and two-stage operand execution pipeline (OEP)
- Three longword FIFO buffer provides decoupling between the pipelines
- 32-bit internal address bus supporting 4 Gbytes of linear address space
- 32-bit data bus
- 16 user-accessible, 32-bit-wide, general-purpose registers
- Supervisor/user modes for system protection
- Vector base register to relocate exception-vector table
- Optimized for high-level language constructs

2.1.1 Decoupled Pipelines

The IFP prefetches instructions. The OEP decodes instructions, fetches required operands, then executes the specified function. The two independent, decoupled pipeline structures maximize performance while minimizing core size. Pipeline stages are shown in Figure 2-1 and are summarized as follows:

- Two-stage IFP (plus optional instruction buffer stage)
  - — Instruction address generation (IAG) calculates the next prefetch address.
  - — Instruction fetch cycle (IC) initiates prefetch on the processor's local instruction bus.
  - — Instruction buffer (IB) optional stage uses FIFO queue to minimize effects of fetch latency.
- Two-stage OEP
  - — Decode, select/operand fetch (DSOC) decodes the instruction and selects the required components for the effective address calculation, or the operand fetch cycle.
  - — Address generation/execute (AGEX) calculates the operand address, or performs the execution of the instruction.

Chapter 3
Hardware Multiply/Accumulate (MAC) Unit

This chapter describes the MCF5272 multiply/accumulate (MAC) unit, which executes integer multiply, multiply-accumulate, and miscellaneous register instructions. The MAC is integrated into the operand execution pipeline (OEP).

3.1 Overview

The MAC unit provides hardware support for a limited set of digital signal processing (DSP) operations used in embedded code, while supporting the integer multiply instructions in the ColdFire microprocessor family.

The MAC unit provides signal processing capabilities for the MCF5272 in a variety of applications including digital audio and servo control. Integrated as an execution unit in the processor's OEP, the MAC unit implements a three-stage arithmetic pipeline optimized for 16 x 16 multiplies. Both 16- and 32-bit input operands are supported by this design in addition to a full set of extensions for signed and unsigned integers plus signed, fixed-point fractional input operands.

The MAC unit provides functionality in three related areas:

- Signed and unsigned integer multiplies
- Multiply-accumulate operations supporting signed, unsigned, and signed fractional operands
- Miscellaneous register operations

Each of the three areas of support is addressed in detail in the succeeding sections. Logic that supports this functionality is contained in a MAC module, as shown in Figure 3-1.

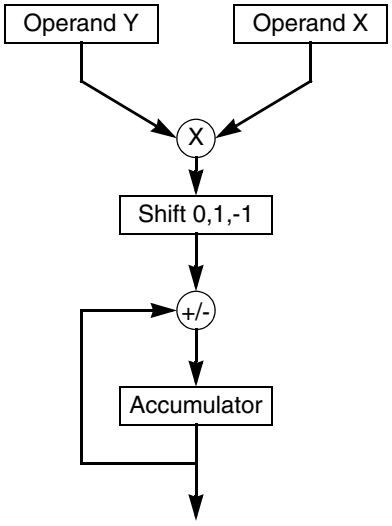

Figure 3-1. ColdFire MAC Multiplication and Accumulation

5.4.1 Revision A Shared Debug Resources

In the Revision A implementation of the debug module, certain hardware structures are shared between BDM and breakpoint functionality as shown in Table 5-4.

Table 5-4. Rev. A Shared BDM/Breakpoint Hardware

Register	BDM Function	Breakpoint Function
AATR	Bus attributes for all memory commands	Attributes for address breakpoint
ABHR	Address for all memory commands	Address for address breakpoint
DBR	Data for all BDM write commands	Data for data breakpoint

Thus, loading a register to perform a specific function that shares hardware resources is destructive to the shared function. For example, a BDM command to access memory overwrites an address breakpoint in ABHR. A BDM write command overwrites the data breakpoint in DBR.

5.4.2 Address Attribute Trigger Register (AATR)

The address attribute trigger register (AATR), Figure 5-5, defines address attributes and a mask to be matched in the trigger. The register value is compared with address attribute signals from the processor's local high-speed bus, as defined by the setting of the trigger definition register (TDR).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	RM	SZM		TTM		TMM			R	SZ		TT		TM		
Reset	0000_0000_0000_0101															
R/W	Write only. AATR is accessible in supervisor mode as debug control register 0x06 using the WDEBUG instruction and through the BDM port using the WDMREG command.															
DRc[4–0]	0x06															

Figure 5-5. Address Attribute Trigger Register (AATR)

Table 5-5 describes AATR fields.

Table 5-5. AATR Field Descriptions

Bits	Name	Description
15	RM	Read/write mask. Setting RM masks R in address comparisons.
14–13	SZM	Size mask. Setting an SZM bit masks the corresponding SZ bit in address comparisons.
12–11	TTM	Transfer type mask. Setting a TTM bit masks the corresponding TT bit in address comparisons.
10–8	TMM	Transfer modifier mask. Setting a TMM bit masks the corresponding TM bit in address comparisons.
7	R	Read/write. R is compared with the R/$\overline{W}$ signal of the processor's local bus.
6–5	SZ	Size. Compared to the processor's local bus size signals. 00 Longword 01 Byte 10 Word 11 Reserved

Table 8-5. CSOR*n* Field Descriptions (continued)

Name	Name	Description
9	RDAH	Controls the address and attribute hold time after the termination, internal or external with $\overline{TA}$, of a read cycle that hits in the chip select address space. 0 Do not hold address and attribute signals an extra cycle after chip select negates on reads. 1 Hold address and attribute signals an extra cycle after chip select negate on reads.
8	EXTBURST	Enable extended burst. Valid only for $\overline{CS7}$. Reserved bit for $\overline{CS}$[0:5]. EXTBURST should be 1 when external SDRAM is configured for a data bus narrower than the width programmed for the MCF5272. EBI must be set for SDRAM and the BW must be set to the data bus width of the external SDRAM array. Example: If the MCF5272 external physical data bus width is 32 or 16 bits but the external SDRAM is 16 bits wide, EXTBURST must be set and BW must be 10 (word) for $\overline{SDCS}$. 0 Extended bursts are not enabled. 1 Extended bursts are enabled.
7	—	Reserved, should be cleared.
6–2	WS	Wait state generator. Specifies the number of wait states (in system clocks) needed before the SIM generates an internal transfer acknowledge signal to terminate the access. 0x00 No wait states 0x01 1 wait state ... 0x1E 30 wait states 0x1F External access For example, WS = 0x0A introduces a 10-clock wait before the bus cycle terminates; 0x1F indicates a source external to the chip select module terminates the access. For SRAM and ROM accesses EBI codes 00 or 11 and WS = 0x1F, $\overline{TA}$ must be driven from an external source to terminate the bus cycle, otherwise the on-chip bus timeout monitor issues a bus error exception. For SRAM and ROM accesses with EBI = 00 or 11 and WS = 0x00–0x1E, the chip select module terminates the bus cycle after the programmed number of system clocks. For SDRAM accesses with $\overline{SDCS}$, EBI = 01, and WS = 0x1F, bus cycles are terminated under control of the SDRAM controller. The CSOR0[WS] reset default is 0x1E. The default for all other CSORs is 0x00. Caution: Never drive $\overline{TA}$ as an input to terminate SDRAM peripheral accesses.
1	RW	RW and MRW determine whether the selected memory region is read only or write only. 0 Read only 1 Write only
0	MRW	MRW must be set for value of RW be taken into consideration. 0 Memory covered by chip select is read/write. The memory covered by the chip select is neither read nor write protected. 1 RW determines whether memory covered by chip select is read only or write only. A conflict causes either a read or write protect violation.

11.5.18 Hash Table Low (HTLR)

The HTLR register, Figure 11-23, contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address.

This register is not reset and must be initialized by the user prior to operation.

	31	16
Field	HASH_LOW	
Reset	Undefined	
R/W	Read/Write	

	15	0
Field	HASH_LOW	
Reset	Undefined	
R/W	Read/Write	
Addr	MBAR + 0xC0C	

Figure 11-23. Hash Table Low (HTLR)

Table 11-26. HTLR Field Descriptions

Bits	Name	Description
31–0	HASH_LOW	The HTLR register contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of HTLR contains hash index bit 31. Bit 0 of HTLR contains hash index bit 0.

Table 12-12. EP0CTL Field Descriptions (continued)

Bits	Name	Description
7	CRC_ERR	CRC error generation enable. This bit enables CRC error generation for debug and test purpose. In order to use this feature, the DEBUG bit must be set. Enabling this bit causes a CRC error on the next data packet transmitted. The CRC_ERR bit must be set again in order to generate another CRC error. This bit only applies to IN transfers. This command bit is write-only and always returns 0 when read. 1 CRC error generation if DEBUG = 1 0 default value
6	—	Reserved, should be cleared.
5–4	OUT_LVL	Endpoint 0 OUT FIFO level for interrupt. This field selects the FIFO level to generate an OUT_LVL interrupt. The OUT_LVL interrupt is generated when the FIFO fills above the selected level. 00 FIFO **25%** Full 01 FIFO 50% Full 10 FIFO 75% Full 11 FIFO 100% Full
3–2	IN_LVL	Endpoint 0 IN FIFO level for interrupt. This field selects the FIFO level to generate an IN_LVL interrupt. The IN_LVL interrupt is generated when the FIFO falls below the selected level. 00 FIFO **25%** Empty 01 FIFO 50% Empty 10 FIFO 75% Empty 11 FIFO 100% Empty
1	IN_DONE	This bit controls the USB's response to IN tokens from the host. This bit is set at Reset and must be cleared by software when the last byte of a transfer has been written to the IN-FIFO. This bit is then subsequently set by the USB core when an end of transfer (EOT) event occurs indicating that the transfer has been completed. An end of transfer (EOT) event is indicated by one of the following: a) An IN packet is transmitted that contains less than the maximum number of bytes defined at endpoint configuration. b) A zero length IN packet is transmitted. This occurs when the previously transmitted IN packet was full, and no more data remains in the IN-FIFO. Hence a single zero length packet must be sent to indicate EOT. 0 CPU has completed writing to the IN-FIFO and transfer is in progress. The USB module will send any amount of data in the FIFO or a zero-length packet when the FIFO is empty. 1 Transfer completed or CPU Busy writing transfer into the IN-FIFO. The USB module will only send maximum size packets or NAK responses if the FIFO contains less than a maximum size packet. This bit is set at Reset and on an EOT event.
0	—	Reserved, should be cleared.

12.4.3 FIFO Configuration

The USB module allows a very flexible FIFO configuration. The FIFO configuration is very application dependent. The FIFO configuration depends on the current configuration number, alternate setting for each interface, and each endpoint's type and packet size. The USB module contains two separate 512-byte FIFO spaces: one for IN and one for OUT endpoints. The following guidelines must be adhered to when configuring the FIFO's:

- The MAX_PACKET field in the EP*n*CFG registers must match the wMaxPacketSize field in the corresponding endpoint descriptor. An incorrect setting causes USB errors and unexpected interrupts.
- EP*n*CFG[FIFO_SIZE] must be a power of 2 and must be at least two packets for non-isochronous endpoints.
- The FIFO_ADDR field in the EP*n*CFG registers must be aligned to a boundary defined by the FIFO_SIZE field. For example, a FIFO with a size equal to 64 bytes can have a starting address of 0, 64, 128, 192, 256, etc.
- The FIFO space for an endpoint defined by FIFO_SIZE and FIFO_ADDR must not overlap with the FIFO space for any other endpoint with the same direction.

An example FIFO configuration with a variety of endpoint types and packet sizes is shown in Table 12-20.

Table 12-20. Example FIFO Setup

Endpoint Number	Endpoint Direction	Endpoint Type	Max Packet Size	FIFO Size	FIFO Starting Address	EP*n*CFG Value
0	IN	Control	8	32	480	0x020101C0
0	OUT	Control	8	32	384	0x02010180
1	IN	Interrupt	17	64	384	0x04420180
2	IN	Bulk	64	128	256	0x10040100
3	OUT	Bulk	32	128	256	0x08040100
4	IN	Isochronous	5	32	448	0x014101C0
5	OUT	Isochronous	5	32	416	0x014101A0
6	IN	Isochronous	300	256	0	0x4B080000
7	OUT	Isochronous	300	256	0	0x4B080000

12.4.4 Data Flow

The handling of the data flow to and from each endpoint can be divided between isochronous and non-isochronous endpoints. Isochronous endpoints are designed to transfer streaming data which is continuous and real-time in creation, delivery, and consumption. The timely delivery of isochronous data is ensured at the expense of potential transient losses in the data stream. In other words, any error in electrical transmission in not corrected by hardware mechanisms such as retries. An example of streaming data is voice.

The other endpoint types transfer message based, or bursted data where integrity of the data is more important than timely delivery.

- Bypass capacitors should be connected between the Vdd and GND pairs with minimal trace length. These capacitors help supply the instantaneous currents of the digital circuitry, in addition to decoupling the noise that may be generated by other sections of the device or other circuitry on the power supply.
- Use short, wide, low inductance traces to connect all of the GND pins together and, with a single trace, connect all of the GND pins to the power supply ground. This helps to reduce voltage spikes in the ground circuit caused by high-speed digital current spikes. Suppressing these voltage spikes on the integrated circuit is the reason for multiple GND leads. A PCB with a ground plane connecting all of the digital and analog GND pins together is the optimal ground configuration, producing the lowest resistance and inductance in the ground circuit.
- Use short, wide, low inductance traces to connect all of the Vdd power supply pins together and, with a single trace, connect all of the Vdd pins to the 3.3V power supply. Connecting all of the digital and analog Vdd pins to the power plane would be the optimal power distribution method for a multi-layer PCB with a power plane.
- The 48-MHz oscillator must be located as close as possible to the chip package. This is required to minimize parasitic capacitance between crystal traces and ground.

12.5.3 Recommended USB Protection Circuit

Figure 12-25 shows the recommended external ESD protection circuit for the USB.

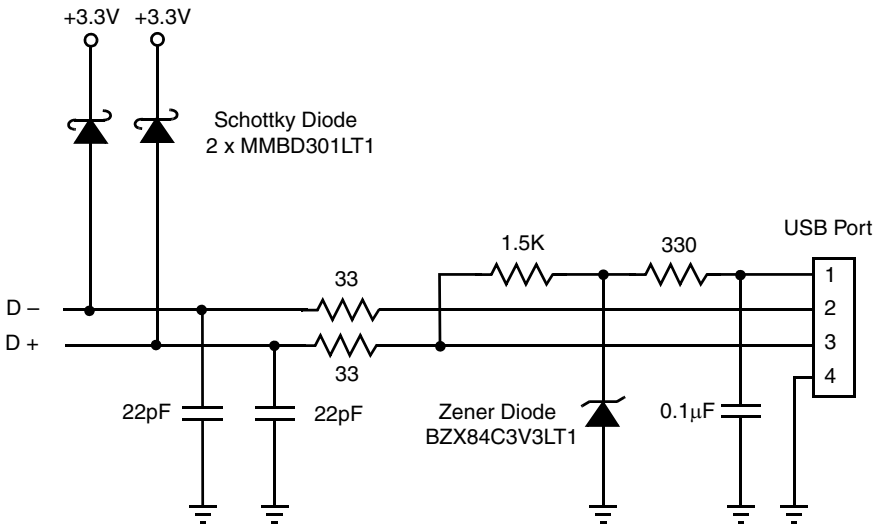

Figure 12-25. USB Protection Circuit

13.5.10 Periodic Status Registers (P0PSR–P3PSR)

All bits in these registers are read only and are set on hardware or software reset.

	15–12	11	10	9	8	7	6	5	4	3	2	1	0
P0PSR–3	—	DTUE	B2TUE	B1TUE	DROE	B2ROE	B1ROE	DTDE	B2TDE	B1TDE	DRDF	B2RDE	B1RDF
Reset	0000_0000_0000_0000												
R/W	Read Only												
Addr	MBAR + 0x384 (P0PSR); 0x386 (P1PSR); 0x388 (P2PSR); 0x38A (P3PSR)												

Figure 13-22. Periodic Status Registers (P0PSR–P3PSR)

P*n*PSR are 16-bit registers containing the interrupt status information for the B- and D-channel transmit and receive registers for each of the four ports on the MCF5272.

Table 13-5. P0PSR–P3PSR Field Descriptions

Bits	Name	Description
15–12	—	Reserved, should be cleared.
11	DTUE	D data transmit underrun error. This bit is set when the data in the PLTD transmit data register for the respective port was transferred to the transmit shadow register, which was already empty indicated by DTDE. DTUE is automatically cleared, when the P*n*PSR register has been read by the CPU.
10	B2TUE	B2 data transmit underrun error. This bit is set when the data in the P*n*B2TR transmit data register for the respective port was transferred to the transmit shadow register, which was already empty indicated by B2TDE. B2TUE is automatically cleared when the P*n*PSR register has been read by the CPU.
9	B1TUE	B1 data transmit underrun error. This bit is set when the data in the P*n*B1TR transmit data register for the respective port was transferred to the transmit shadow register, which was already empty indicated by B1TDE. B1TUE is automatically cleared when the P*n*PSR register has been read by the CPU.
8	DROE	D-Channel data receive overrun error. This bit is set when the data in the D receive shadow register for the respective port has been transferred to the receive data register P*n*DRR, which was already full indicated by DRDF. DROE is automatically cleared when the P*n*PSR register has been read by the CPU.
7	B2ROE	B2 data receive overrun error. This bit is set when the data in the B2 receive shadow register for the respective port has been transferred to the receive data register P*n*B2RR, which was already full indicated by B2RDF. B2ROE is automatically cleared when the P*n*PSR register has been read by the CPU.
6	B1ROE	B1 data receive overrun error. This bit is set when the data in the B1 receive shadow register for the respective port has been transferred to the receive data register P*n*B1RR, which was already full indicated by B1RDF. B1ROE is automatically cleared when the P*n*PSR register has been read by the CPU. Note: Overrun and Underrun conditions are caused by the B and/or D-channel receive or transmit data registers not being read or written prior to a 2-KHz super frame arriving.
5	DTDE	D data transmit data empty. This bit is set when the data in the PLTD transmit data register for the respective port has been transferred to the transmit shadow register. This bit is cleared when the CPU writes data to PLTD.
4	B2TDE	B2 data transmit data empty. This bit is set when the data in the P*n*B2TR transmit data register for the respective port has been transferred to the transmit shadow register. This bit is cleared when the CPU writes data to P*n*B2TR.
3	B1TDE	B1 data transmit data empty. This bit is set when the data in the P*n*B1TR transmit data register for the respective port has been transferred to the transmit shadow register. This bit is cleared when the CPU writes data to P*n*B1TR.

13.5.14 GCI Monitor Channel Transmit Abort Register (PGMTA)

All bits in this register are read/write and are cleared on hardware or software reset.

The PGMTA register contains the abort control bits for each of the four ports on the MCF5272 for the transmit monitor channel.

	7	6	5	4	3 … 0
Field	AR3	AR2	AR1	AR0	—
Reset	0000_0000				
R/W	Read/Write				
Addr	MBAR + 0x372				

Figure 13-26. GCI Monitor Channel Transmit Abort Register (PGMTA)

Table 13-9. PGMTA Field Descriptions

Bits	Name	Description
7	AR3	Abort request, port 3. 0 Default reset value. 1 Set by the CPU, this bit causes the monitor channel controller to transmit the end of message signal on the E bit. Automatically cleared by the monitor channel controller on receiving an abort, that is, when PGMTS[AB] is set.
6	AR2	Abort request, port 2. See AR3.
5	AR1	Abort request, port 1. See AR3.
4	AR0	Abort request, port 0. See AR3.
3–0	—	Reserved, should be cleared.

13.5.19 D-Channel Status Register (PDCSR)

All bits in this register are read only and are cleared on hardware or software reset. The register is also cleared after a read operation.

The PDCSR register contains the D-channel status bits for all four ports on the MCF5272.

	7	6	5	4	3	2	1	0
Field	—		DG1	DG0	DC3	DC2	DC1	DC0
Reset	0000_0000							
R/W	Read Only							
Addr	MBAR + 0x383							

Figure 13-31. D-Channel Status Register (PDCSR)

Table 13-14. PDCSR Field Descriptions

Bits	Name	Description
7–6	—	Reserved, should be cleared.
5	DG1	D-channel grant, port 1. 0 Default reset value. 1 In IDL mode, indicates the status of DGRANT. When the external DGNT has a logic 1, the corresponding DG1/DG0 bit is set. In GCI mode, DG1 and DG0 reflects the inverted value of the SCIT bit. The significance of this bit is the same in IDL or GCI mode, that is, in IDL mode when the DG bit is set, the D channel is granted. In GCI mode when the DG bit is set, this corresponds to the GO condition. In both cases the D channel is granted and communication may commence.
4	DG0	D-channel grant, port 0. See DG1.
3	DC3	D-channel change, port 3. 0 Default reset value. 1 Indicates that a value other than 0xFF (all ones) exists the D-channel receive register.
2	DC2	D-channel change, port 2. See DC3.
1	DC1	D-channel change, port 1. See DC3.
0	DC0	D-channel change, port 0. See DC3.

14.4.4 Transfer Length

There are two transfer length options. The user can choose a default value of 8 bits or a programmed value of 8 to 16 bits inclusive. The programmed value must be written into QMR[BITS]. The bits per transfer enable (BITSE) field in the command RAM determines whether the default value (BITSE = 0) or the BITS[3–0] value (BITSE = 1) is used. QMR[BITS] gives the required number of bits to be transferred, with 0b0000 representing 16.

14.4.5 Data Transfer

Operation is initiated by setting QDLYR[SPE]. Shortly after QDLYR[SPE] is set, the QSPI executes the command at the command RAM address pointed to by QWR[NEWQP]. Data at the pointer address in transmit RAM is loaded into the data serializer and transmitted. Data that is simultaneously received is stored at the pointer address in receive RAM.

When the proper number of bits has been transferred, the QSPI stores the working queue pointer value in QWR[CPTQP], increments the working queue pointer, and loads the next data for transfer from the transmit RAM. The command pointed to by the incremented working queue pointer is executed next unless a new value has been written to QWR[NEWQP]. If a new queue pointer value is written while a transfer is in progress, then that transfer is completed normally.

When the CONT bit in the command RAM is set, the QSPI_CS signals are asserted between transfers. When CONT is cleared, QSPI_CS[0:3] are negated between transfers. The QSPI_CS signals are not high impedance.

When the QSPI reaches the end of the queue, it asserts the SPIF flag, QIR[SPIF]. If QIR[SPIFE] is set, an interrupt request is generated when QIR[SPIF] is asserted. Then the QSPI clears QDLYR[SPE] and stops, unless wraparound mode is enabled.

Wraparound mode is enabled by setting QWR[WREN]. The queue can wrap to pointer address 0x0, or to the address specified by QWR[NEWQP], depending on the state of QWR[WRTO].

In wraparound mode, the QSPI cycles through the queue continuously, even while requesting interrupt service. QDLYR[SPE] is not cleared when the last command in the queue is executed. New receive data overwrites previously received data in the receive RAM. Each time the end of the queue is reached, QIR[SPIFE] is set. QIR[SPIF] is not automatically reset. If interrupt driven QSPI service is used, the service routine must clear QIR[SPIF] to abort the current request. Additional interrupt requests during servicing can be prevented by clearing QIR[SPIFE].

There are two recommended methods of exiting wraparound mode: clearing QWR[WREN] or setting QWR[HALT]. Exiting wraparound mode by clearing QDLYR[SPE] is not recommended because this may abort a serial transfer in progress. The QSPI sets SPIF, clears QDLYR[SPE], and stops the first time it reaches the end of the queue after QWR[WREN] is cleared. After QWR[HALT] is set, the QSPI finishes the current transfer, then stops executing commands. After the QSPI stops, QDLYR[SPE] can be cleared.

16.3.14 UART Receiver FIFO Registers (URF*n*)

The URF*n* registers contain control and status bits for the receiver FIFO. Note that some bits are read only.

	7	6	5	4	0
Field	RXS		FULL	RXB	
Reset	0000_0000				
R/W	R/W		R	R	
Address	MBAR + 0x12C (URF0), 0x16C (URF1)				

Figure 16-17. UART Receiver FIFO Registers (URF*n*)

Table 16-11 describes URF*n* fields.

Table 16-11. URF*n* Field Descriptions

Bits	Name	Description
7–6	RXS	$\overline{\text{Receiver status}}$. When written to, these bits control the meaning of UISR*n*[RxFIFO]. 00 Inhibit receiver FIFO status indication in UISR*n*. 01 Receiver FIFO Š 25% full 10 Receiver FIFO Š 50% full 11 Receiver FIFO Š 75% full When read, these bits indicate the emptiness level of the FIFO. 00 Receiver FIFO < 25% full 01 Receiver FIFO Š 25% full 10 Receiver FIFO Š 50% full 11 Receiver FIFO Š 75% full
5	FULL	Receiver $\overline{\text{FIFO full}}$. 0 Receiver FIFO is not full and can be loaded with a character. 1 Receiver FIFO is full. Characters loaded from the receiver when the FIFO is full are lost. This bit is identical to USR*n*[FFULL].
4–0	RXB	Receiver buffer data level. Indicates the number of bytes, between 0 and 24, stored in the receiver FIFO.

17.3.3 Port C Data Direction Register (PCDDR)

The PCDDR determines the signal direction of each parallel port pin programmed as a GPIO port in the PCCNT.

	15	0
Field	PCDDR	
Reset	0000_0000_0000_0000	
R/W	Read/Write	
Addr	MBAR + 0x0094	

Figure 17-6. Port C Data Direction Register (PCDDR)

17.4 Port Data Registers

These 16-bit bidirectional registers are used to read or write the logic states of the GPIO lines. This register has no effect on pins not configured for general-purpose I/O.

After a system reset, these register bits are all cleared. When any port lines are configured as outputs, a logic zero appears on those pins, unless the data register is written with an initial data value prior to setting the pin direction.

The reset values given in the following register diagrams are the port output values written to the registers during reset, and do not reflect the value of a register read cycle. Register reads always return the instantaneous value of the corresponding pins.

17.4.1 Port Data Register (P*x*DAT)

In the following description P*x*DAT refers to PADAT, PBDAT, or PCDAT.

The P*x*DAT value for inputs corresponds to the logic level at the pin; for outputs, the value corresponds to the logic level driven onto the pin. Note that P*x*DAT has no effect on pins which have not been configured for GPIO.

	15	0
Field	P*x*DAT	
Reset	Undefined	
R/W	Read/Write	
Addr	MBAR + 0x0086 (PADAT); 0x008E (PBDAT); 0x0096 (PCDAT)	

Figure 17-7. Port *x* Data Register (PADAT, PBDAT, and PCDAT)

Chapter 18
Pulse-Width Modulation (PWM) Module

This chapter describes the configuration and operation of the pulse-width modulation (PWM) module. It includes a block diagram, programming model, and timing diagram.

18.1 Overview

The PWM module shown in Figure 18-1, generates a synchronous series of pulses having programmable duty cycle. With a suitable low-pass filter, the PWM can be used as a digital-to-analog converter.

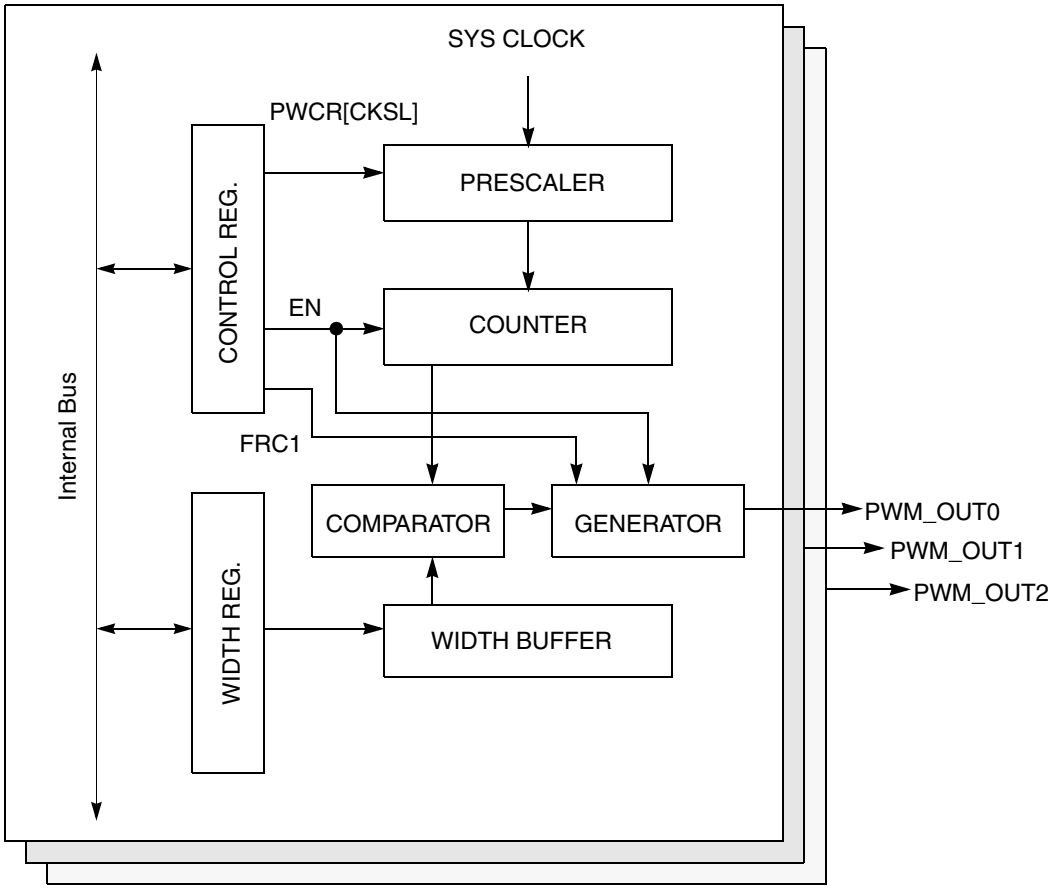

Figure 18-1. PWM Block Diagram (3 Identical Modules)

19.11.4 USB Transmit Data Negative (USB_TN/PA3)

USB mode: USB_TN is the inverted data transmit output.

Port A mode: This pin can also be configured as the PA3 I/O.

19.11.5 USB Suspend Driver (USB_SUSP/PA4)

USB mode: USB_SUSP is used to put the USB driver in suspend operation.

Port A mode: This pin can also be configured as the PA4 I/O.

19.11.6 USB Transmitter Output Enable ($\overline{\text{USB\_TxEN}}$/PA5)

USB mode: $\overline{\text{USB\_TxEN}}$ enables the transceiver to transmit data on the bus. It requires a 4.7-K¾ pullup resistor to ensure that the external USB Tx driver is off between the MCF5272 coming out of reset and initializing the port A pin configuration register.

Port A mode: This pin can also be configured as the PA5 I/O.

19.11.7 USB Rx Data Output (USB_RxD/PA6)

USB mode: USB_RxD is the receive data output from the differential receiver inputs USB_RN and USB_RP.

Port A mode: This pin can also be configured as the PA6 I/O.

19.11.8 USB_D+ and USB_D-

USB_D+ and USB_D- are the on-chip USB interface transceiver signals. When these signals are enabled, the USB module uses them to communicate to an external USB bus. When not used, each signal should be pulled to VDD using a 4.7-K¾ resistor.

19.11.9 USB_CLK

USB_CLK is used to connect an external 48-MHz oscillator to the USB module. When this pin is tied to GND or VDD, the USB module automatically uses the internal CPU clock. In this case the CLKIN must be 48 MHz if the system is to use the USB function.

19.11.10 $\overline{\text{INT1}}$/USB Wake-on-Ring (USB_WOR)

The USB module allows for $\overline{\text{INT1}}$ to generate the USB wake-on-ring signal to the USB host controller. This function is enabled by a control bit in the USB module. WOR is provided to allow the CPU and the USB interface to be woken up when in power down mode. This occurs when the USB controller detects a resume state at the USB inputs.

The interrupt output of an ISDN transceiver, such as the MC145574, can be connected to $\overline{\text{INT1}}$/USB_WOR. Before putting the device into sleep mode, the USB module's wake on ring function

Appendix A
List of Memory Maps

The MCF5272 memory map is given in this section.

The addresses for the system configuration registers are absolute addresses in the MCF5272 CPU space memory map. The on-chip peripheral modules are configured as a group by programming the module base address register, (MBAR).

In the title block for each module is shown the offset address of the given register from the base address programmed in MBAR.

The following formula allows calculation of the absolute address of a given register.

Absolute address = MBAR + register offset

A.1 List of Memory Map Tables

Table A-1. On-Chip Module Base Address Offsets from MBAR

Module	Module Base Address	Mnemonic
Configuration Registers	MBAR+0x0000	CFG_Base
Interrupt Registers	MBAR+0x0020	INT_Base
Chip Select Registers	MBAR+0x0040	CS_Base
Ports Registers	MBAR+0x0080	GPIO_Base
QSPI Module Registers	MBAR+0x00A0	QSPI_Base
PWM Module Registers	MBAR+0x00C0	PWM_Base
DMA Module Registers	MBAR+0x00E0	DMAC_Base
UART0 Module Registers	MBAR+0x0100	UART0_Base
UART1 Module Registers	MBAR+0x0140	UART1_Base
SDRAM Controller Registers	MBAR+0x0180	SDRAMC_Base
Timer Module Registers	MBAR+0x0200	Timer_Base
PLIC Module Registers	MBAR+0x0300	PLIC_Base
Ethernet Module Registers	MBAR+0x0800	ENET_Base
USB Module Registers	MBAR+0x1000	USB_Base

Table A-16. USB Module Memory Map (continued)

MBAR Offset	[31:24]	[23:16]	[15:8]	[7:0]
0x108C	USB Endpoint 0 Interrupt Mask Register (EP0IMR)			
0x1090	Reserved		USB Endpoint 1 Interrupt Mask Register (EP1IMR)	
0x1094	Reserved		USB Endpoint 2 Interrupt Mask Register (EP2IMR)	
0x1098	Reserved		USB Endpoint 3 Interrupt Mask Register (EP3IMR)	
0x109C	Reserved		USB Endpoint 4 Interrupt Mask Register (EP4IMR)	
0x10A0	Reserved		USB Endpoint 5 Interrupt Mask Register (EP5IMR)	
0x10A4	Reserved		USB Endpoint 6 Interrupt Mask Register (EP6IMR)	
0x10A8	Reserved		USB Endpoint 7 Interrupt Mask Register (EP7IMR)	
0x10AC	USB Endpoint 0 Data Register (EP0DR)			
0x10B0	USB Endpoint 1 Data Register (EP1DR)			
0x10B4	USB Endpoint 2 Data Register (EP2DR)			
0x10B8	USB Endpoint 3 Data Register (EP3DR)			
0x10BC	USB Endpoint 4 Data Register (EP4DR)			
0x10C0	USB Endpoint 5 Data Register (EP5DR)			
0x10C4	USB Endpoint 6 Data Register (EP6DR)			
0x10C8	USB Endpoint 7 Data Register (EP7DR)			
0x10CE	Reserved		USB Endpoint 0 Data Present Register (EP0DPR)	
0x10D2	Reserved		USB Endpoint 1 Data Present Register (EP1DPR)	
0x10D6	Reserved		USB Endpoint 2 Data Present Register (EP2DPR)	
0x10DA	Reserved		USB Endpoint 3 Data Present Register (EP3DPR)	
0x10DE	Reserved		USB Endpoint 4 Data Present Register (EP4DPR)	
0x10E2	Reserved		USB Endpoint 5 Data Present Register (EP5DPR)	
0x10E6	Reserved		USB Endpoint 6 Data Present Register (EP6DPR)	
0x10EA	Reserved		USB Endpoint 7 Data Present Register (EP7DPR)	
0x1400 – 0x17FF	USB Configuration RAM, 1 K Bytes			