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	CPU32
Core Size	32-Bit Single-Core
Speed	16MHz
Connectivity	EBI/EMI, SCI, SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	18
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	-
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	132-BQFP Bumpered
Supplier Device Package	132-PQFP (24.13x24.13)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68331cfc16b1

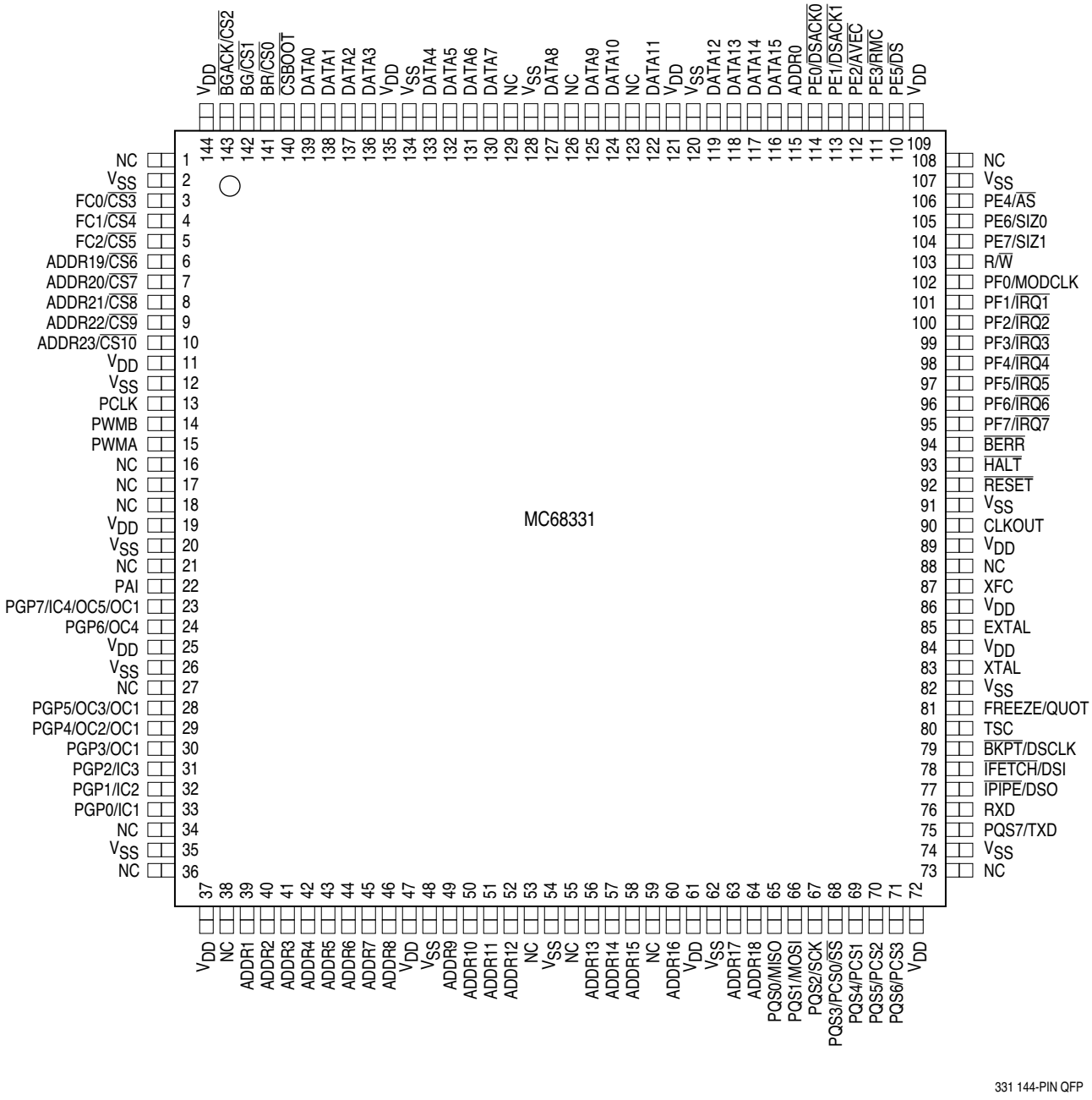


Figure 3 MC68331 144-Pin QFP Pin Assignments

1.4 Address Map

The following figure is a map of the MCU internal addresses. Unimplemented blocks are mapped externally.

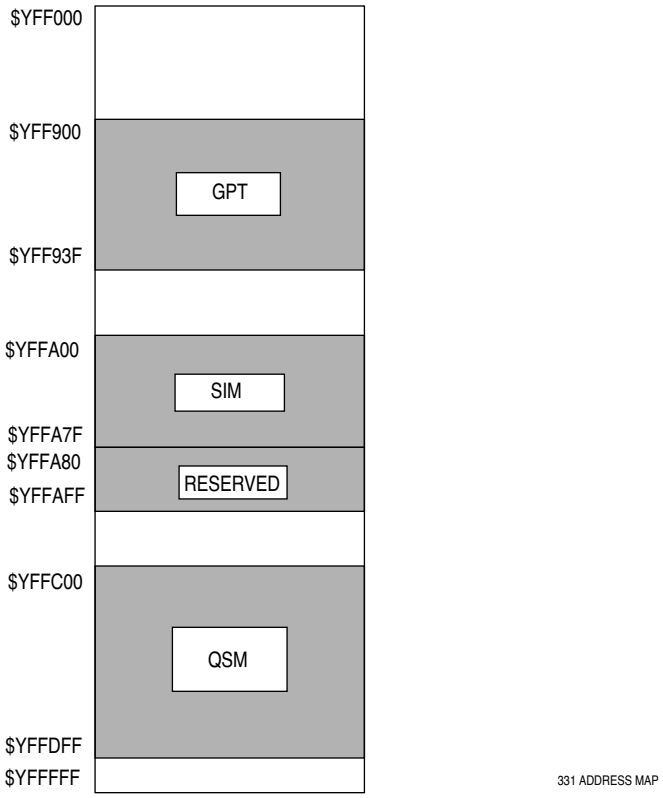


Figure 4 MCU Address Map

1.5 Intermodule Bus

The intermodule bus (IMB) is a standardized bus developed to facilitate both design and operation of modular microcontrollers. It contains circuitry to support exception processing, address space partitioning, multiple interrupt levels, and vectored interrupts. The standardized modules in the MCU communicate with one another and with external components through the IMB. The IMB in the MCU uses 24 address and 16 data lines.

2 Signal Descriptions

2.1 Pin Characteristics

The following table shows MCU pins and their characteristics. All inputs detect CMOS logic levels. All inputs can be put in a high-impedance state, but the method of doing this differs depending upon pin function. Refer to **Table 4**, for a description of output drivers. An entry in the discrete I/O column of **Table 2** indicates that a pin has an alternate I/O function. The port designation is given when it applies. Refer to the MCU Block Diagram for information about port organization.

Table 2 MCU Pin Characteristics

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
ADDR23/$\overline{CS10}$/ECLK	A	Y	N	O	—
ADDR[22:19]/$\overline{CS[9:6]}$	A	Y	N	O	PC[6:3]
ADDR[18:0]	A	Y	N	—	—
$\overline{AS}$	B	Y	N	I/O	PE5
$\overline{AVEC}$	B	Y	N	I/O	PE2
$\overline{BERR}$	B	Y	N	—	—
$\overline{BG}$/$\overline{CS1}$	B	—	—	—	—
$\overline{BGACK}$/$\overline{CS2}$	B	Y	N	—	—
$\overline{BKPT}$/DSCLK	—	Y	Y	—	—
$\overline{BR}$/$\overline{CS0}$	B	Y	N	—	—
CLKOUT	A	—	—	—	—
$\overline{CSBOOT}$	B	—	—	—	—
DATA[15:0][1]	Aw	Y	N	—	—
$\overline{DS}$	B	Y	N	I/O	PE4
$\overline{DSACK1}$	B	Y	N	I/O	PE1
$\overline{DSACK0}$	B	Y	N	I/O	PE0
DSI/$\overline{IFETCH}$	A	Y	Y	—	—
DSO/$\overline{IPIPE}$	A	—	—	—	—
EXTAL[2]	—	—	Special	—	—
FC[2:0]/$\overline{CS[5:3]}$	A	Y	N	O	PC[2:0]
FREEZE/QUOT	A	—	—	—	—
IC4/OC5	A	Y	Y	I/O	GP4
IC[3:1]	A	Y	Y	I/O	GP[7:5]
$\overline{HALT}$	Bo	Y	N	—	—
$\overline{IRQ[7:1]}$	B	Y	Y	I/O	PF[7:1]
MISO	Bo	Y	Y	I/O	PQS0
MODCLK[1]	B	Y	N	I/O	PF0
MOSI	Bo	Y	Y	I/O	PQS1
OC[4:1]	A	Y	Y	I/O	GP[3:0]
PAI[3]	—	Y	Y	I	—
PCLK[3]	—	Y	Y	I	—
$\overline{PCS0}$/$\overline{SS}$	Bo	Y	Y	I/O	PQS3
$\overline{PCS[3:1]}$	Bo	Y	Y	I/O	PQS[6:4]
PWMA, PWMB	A	—	—	O	—
R/$\overline{W}$	A	Y	N	—	—
$\overline{RESET}$	Bo	Y	Y	—	—
$\overline{RMC}$	B	Y	N	I/O	PE3

3 System Integration Module

The system integration module (SIM) consists of five functional blocks that control system start-up, initialization, configuration, and external bus.

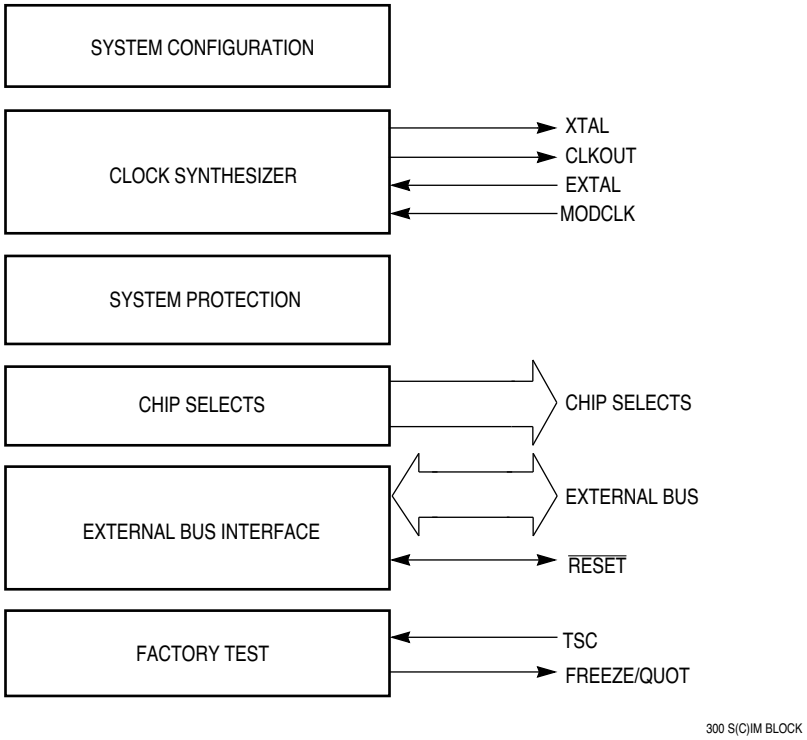


Figure 5 SIM Block Diagram

3.1 Overview

The system configuration and protection block controls MCU configuration and operating mode. The block also provides bus and software watchdog monitors.

The system clock generates clock signals used by the SIM, other IMB modules, and external devices. In addition, a periodic interrupt generator supports execution of time-critical control routines.

The external bus interface handles the transfer of information between IMB modules and external address space.

The chip-select block provides eleven general-purpose chip-select signals and a boot ROM chip-select signal. Both general-purpose and boot ROM chip-select signals have associated base address registers and option registers.

The system test block incorporates hardware necessary for testing the MCU. It is used to perform factory tests, and its use in normal applications is not supported.

The SIM control register address map occupies 128 bytes. Unused registers within the 128-byte address space return zeros when read. The “Access” column in the SIM address map below indicates which registers are accessible only at the supervisor privilege level and which can be assigned to either the supervisor or user privilege level, according to the value of the SUPV bit in the SIMCR.

PIRQL	Interrupt Request Level
000	Periodic Interrupt Disabled
001	Interrupt Request Level 1
010	Interrupt Request Level 2
011	Interrupt Request Level 3
100	Interrupt Request Level 4
101	Interrupt Request Level 5
110	Interrupt Request Level 6
111	Interrupt Request Level 7

PIV[7:0] —Periodic Interrupt Vector

The bits of this field contain the vector generated in response to an interrupt from the periodic timer. When the SIM responds, the periodic interrupt vector is placed on the bus.

PITR —Periodic Interrupt Timer Register **$YFFA24**

15	14	13	12	11	10	9	8	7							0
0	0	0	0	0	0	0	PTP	PITM							
RESET:															
0	0	0	0	0	0	0	$\overline{\text{MODCLK}}$	0	0	0	0	0	0	0	0

The PITR contains the count value for the periodic timer. A zero value turns off the periodic timer. This register can be read or written at any time.

PTP —Periodic Timer Prescaler Control

0 = Periodic timer clock not prescaled

1 = Periodic timer clock prescaled by a value of 512

The reset state of PTP is the complement of the state of the MODCLK signal during reset.

PITM[7:0] —Periodic Interrupt Timing Modulus Field

This is an 8-bit timing modulus. The period of the timer can be calculated as follows:

$$\text{PIT Period} = [(\text{PITM})(\text{Prescaler})(4)]/\text{EXTAL}$$

where

PIT Period = Periodic interrupt timer period

PITM = Periodic interrupt timer register modulus (PITR[7:0])

EXTAL Frequency = Crystal frequency

Prescale = 512 or 1 depending on the state of the PTP bit in the PITR

3.3 System Clock

The system clock in the SIM provides timing signals for the IMB modules and for an external peripheral bus. Because MCU operation is fully static, register and memory contents are not affected when the clock rate changes. System hardware and software support changes in the clock rate during operation.

The system clock signal can be generated in three ways. An internal phase-locked loop can synthesize the clock from an internal or external frequency source, or the clock signal can be input from an external source.

Following is a block diagram of the clock submodule.

Port width is the maximum number of bits accepted or provided during a bus transfer. External devices must follow the handshake protocol described below. Control signals indicate the beginning of the cycle, the address space, the size of the transfer, and the type of cycle. The selected device controls the length of the cycle. Strobe signals, one for the address bus and another for the data bus, indicate the validity of an address and provide timing information for data. The EBI operates in an asynchronous mode for any port width.

To add flexibility and minimize the necessity for external logic, MCU chip-select logic can be synchronized with EBI transfers. Chip-select logic can also provide internally-generated bus control signals for these accesses. Refer to **3.5 Chip Selects** for more information.

3.4.1 Bus Control Signals

The CPU initiates a bus cycle by driving the address, size, function code, and read/write outputs. At the beginning of the cycle, size signals SIZ0 and SIZ1 are driven along with the function code signals. The size signals indicate the number of bytes remaining to be transferred during an operand cycle. They are valid while the address strobe ($\overline{AS}$) is asserted. The following table shows SIZ0 and SIZ1 encoding. The read/write (R/$\overline{W}$) signal determines the direction of the transfer during a bus cycle. This signal changes state, when required, at the beginning of a bus cycle, and is valid while $\overline{AS}$ is asserted. R/$\overline{W}$ only changes state when a write cycle is preceded by a read cycle or vice versa. The signal can remain low for two consecutive write cycles.

Table 8 Size Signal Encoding

SIZ1	SIZ0	Transfer Size
0	1	Byte
1	0	Word
1	1	Three Byte
0	0	Long Word

3.4.2 Function Codes

The CPU32 automatically generates function code signals FC[2:0]. The function codes can be considered address extensions that automatically select one of eight address spaces to which an address applies. These spaces are designated as either user or supervisor, and program or data spaces. Address space 7 is designated CPU space. CPU space is used for control information not normally associated with read or write bus cycles. Function codes are valid while $\overline{AS}$ is asserted.

Table 9 CPU32 Address Space Encoding

FC2	FC1	FC0	Address Space
0	0	0	Reserved
0	0	1	User Data Space
0	1	0	User Program Space
0	1	1	Reserved
1	0	0	Reserved
1	0	1	Supervisor Data Space
1	1	0	Supervisor Program Space
1	1	1	CPU Space

3.4.3 Address Bus

Address bus signals ADDR[23:0] define the address of the most significant byte to be transferred during a bus cycle. The MCU places the address on the bus at the beginning of a bus cycle. The address is valid while $\overline{AS}$ is asserted.

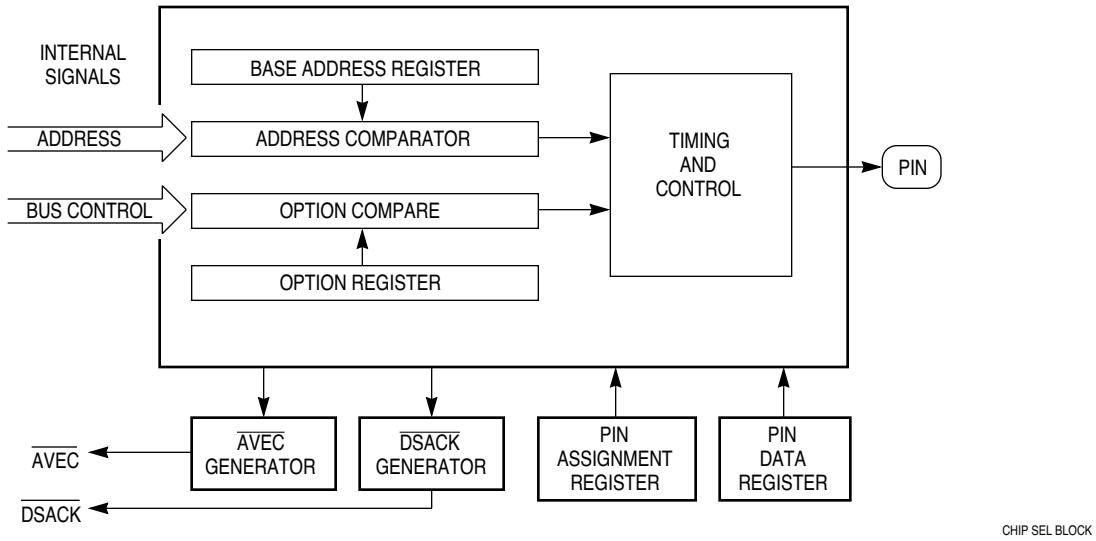


Figure 9 Chip-Select Circuit Block Diagram

The following table lists allocation of chip-selects and discrete outputs on the pins of the MCU.

Pin	Chip Select	Discrete Outputs
$\overline{\text{CSBOOT}}$	$\overline{\text{CSBOOT}}$	—
$\overline{\text{BR}}$	$\overline{\text{CS0}}$	—
$\overline{\text{BG}}$	$\overline{\text{CS1}}$	—
$\overline{\text{BGACK}}$	$\overline{\text{CS2}}$	—
FC0	$\overline{\text{CS3}}$	PC0
FC1	$\overline{\text{CS4}}$	PC1
FC2	$\overline{\text{CS5}}$	PC2
ADDR19	$\overline{\text{CS6}}$	PC3
ADDR20	$\overline{\text{CS7}}$	PC4
ADDR21	$\overline{\text{CS8}}$	PC5
ADDR22	$\overline{\text{CS9}}$	PC6
ADDR23	$\overline{\text{CS10}}$	ECLK

3.5.1 Chip-Select Registers

Pin assignment registers CSPAR0 and CSPAR1 determine functions of chip-select pins. These registers also determine port size (8- or 16-bit) for dynamic bus allocation.

A pin data register (PORTC) latches discrete output data.

Blocks of addresses are assigned to each chip-select function. Block sizes of 2 Kbytes to 1 Mbyte can be selected by writing values to the appropriate base address register (CSBAR). Address blocks for separate chip-select functions can overlap.

Chip-select option registers (CSORBT and CSOR[10:0]) determine timing of and conditions for assertion of chip-select signals. Eight parameters, including operating mode, access size, synchronization, and wait state insertion can be specified.

Initialization code often resides in a peripheral memory device controlled by the chip-select circuits. A set of special chip-select functions and registers (CSORBT, CSBARBT) is provided to support bootstrap operation.

Block Size Field	Block Size	Address Lines Compared
000	2 K	ADDR[23:11]
001	8 K	ADDR[23:13]
010	16 K	ADDR[23:14]
011	64 K	ADDR[23:16]
100	128 K	ADDR[23:17]
101	256 K	ADDR[23:18]
110	512 K	ADDR[23:19]
111	1 M	ADDR[23:20]

ADDR[23:11] —Base Address Field

This field sets the starting address of a particular address space. The address compare logic uses only the most significant bits to match an address within a block. The value of the base address must be a multiple of block size. Base address register diagrams show how base register bits correspond to address lines.

3.5.4 Option Registers

The option registers contain eight fields that determine timing of and conditions for assertion of chip-select signals. For a chip-select signal to be asserted, all bits in the base address register must match the corresponding internal upper address lines, and all conditions specified in the option register must be satisfied. These conditions also apply to providing $\overline{\text{DSACK}}$ or autovector support.

CSORBT —Chip-Select Option Register Boot ROM **$YFFA4A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE	BYTE		R/$\overline{\text{W}}$		STRB	DSACK				SPACE		IPL			$\overline{\text{AVEC}}$
RESET:															
0	1	1	1	1	0	1	1	0	1	1	1	0	0	0	0

CSOR[10:0] —Chip-Select Option Registers **$YFFA4E–$YFFA76**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE	BYTE		R/$\overline{\text{W}}$		STRB	DSACK				SPACE		IPL			$\overline{\text{AVEC}}$
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSORBT, the option register for $\overline{\text{CSBOOT}}$, contains special reset values that support bootstrap operations from peripheral memory devices.

The following bit descriptions apply to both CSORBT and CSOR[10:0] option registers.

MODE —Asynchronous/Synchronous Mode

0 = Asynchronous mode selected (chip-select assertion determined by internal or external bus control signals)

1 = Synchronous mode selected (chip-select assertion synchronized with ECLK signal)

In asynchronous mode, the chip select is asserted synchronized with $\overline{\text{AS}}$ or $\overline{\text{DS}}$.

The $\overline{\text{DSACK}}$ field is not used in synchronous mode because a bus cycle is only performed as a synchronous operation. When a match condition occurs on a chip select programmed for synchronous operation, the chip select signals the EBI that an ECLK cycle is pending.

BYTE —Upper/Lower Byte Option

This field is used only when the chip-select 16-bit port option is selected in the pin assignment register. The following table lists upper/lower byte options.

SPACE —Address Space

Use this option field to select an address space for the chip-select logic. The CPU32 normally operates in supervisor or user space, but interrupt acknowledge cycles must take place in CPU space.

Space Field	Address Space
00	CPU Space
01	User Space
10	Supervisor Space
11	Supervisor/User Space

IPL —Interrupt Priority Level

If the space field is set for CPU space (00), chip-select logic can be used for interrupt acknowledge. During an interrupt acknowledge cycle, the priority level on address lines ADDR[3:1] is compared to the value in the IPL field. If the values are the same, a chip select is asserted, provided that other option register conditions are met. The following table shows IPL field encoding.

IPL	Description
000	Any Level
001	IPL1
010	IPL2
011	IPL3
100	IPL4
101	IPL5
110	IPL6
111	IPL7

This field only affects the response of chip selects and does not affect interrupt recognition by the CPU. Any level means that chip select is asserted regardless of the level of the interrupt acknowledge cycle.

$\overline{\text{AVEC}}$ —Autovector Enable

0 = External interrupt vector enabled

1 = Autovector enabled

This field selects one of two methods of acquiring the interrupt vector during the interrupt acknowledge cycle. It is not usually used in conjunction with a chip-select pin.

If the chip select is configured to trigger on an interrupt acknowledge cycle (SPACE = 00) and the $\overline{\text{AVEC}}$ field is set to one, the chip select automatically generates an $\overline{\text{AVEC}}$ in response to the interrupt cycle. Otherwise, the vector must be supplied by the requesting device.

The $\overline{\text{AVEC}}$ bit must not be used in synchronous mode, as autovector response timing can vary because of ECLK synchronization.

3.5.5 Port C Data Register

Bit values in port C determine the state of chip-select pins used for discrete output. When a pin is assigned as a discrete output, the value in this register appears at the output. This is a read/write register. Bit 7 is not used. Writing to this bit has no effect, and it always returns zero when read.

PORTC — Port C Data Register **$YFFA41**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	0	PC6	PC5	PC4	PC3	PC2	PC1	PC0
RESET:	0	1	1	1	1	1	1	1

The SIM clock synthesizer provides clock signals to the other MCU modules. After the clock is running and the internal reset signal is asserted for four clock cycles, these modules reset. V_{DD} ramp time and VCO frequency ramp time determine how long these four cycles take. Worst case is approximately 15 milliseconds. During this period, module port pins may be in an indeterminate state. While input-only pins can be put in a known state by means of external pull-up resistors, external logic on input/output or output-only pins must condition the lines during this time. Active drivers require high-impedance buffers or isolation resistors to prevent conflict.

3.7.5 Use of Three State Control Pin

Asserting the three-state control (TSC) input causes the MCU to put all output drivers in an inactive, high-impedance state. The signal must remain asserted for 10 clock cycles in order for drivers to change state. There are certain constraints on use of TSC during power-on reset:

When the internal clock synthesizer is used (MODCLK held high during reset), synthesizer ramp-up time affects how long the 10 cycles take. Worst case is approximately 20 milliseconds from TSC assertion.

When an external clock signal is applied (MODCLK held low during reset), pins go to high-impedance state as soon after TSC assertion as 10 clock pulses have been applied to the EXTAL pin.

When TSC assertion takes effect, internal signals are forced to values that can cause inadvertent mode selection. Once the output drivers change state, the MCU must be powered down and restarted before normal operation can resume.

3.8 Interrupts

Interrupt recognition and servicing involve complex interaction between the central processing unit, the system integration module, and a device or module requesting interrupt service.

The CPU32 provides for eight levels of interrupt priority (0–7), seven automatic interrupt vectors and 200 assignable interrupt vector. All interrupts with priorities less than 7 can be masked by the interrupt priority (IP) field in the status register. The CPU32 handles interrupts as a type of asynchronous exception.

Interrupt recognition is based on the states of interrupt request signals $\overline{\text{IRQ}}$[7:1] and the IP mask value. Each of the signals corresponds to an interrupt priority. $\overline{\text{IRQ1}}$ has the lowest priority, and $\overline{\text{IRQ7}}$ has the highest priority.

The IP field consists of three bits. Binary values %000 to %111 provide eight priority masks. Masks prevent an interrupt request of a priority less than or equal to the mask value (except for $\overline{\text{IRQ7}}$) from being recognized and processed. When IP contains %000, no interrupt is masked. During exception processing, the IP field is set to the priority of the interrupt being serviced.

Interrupt request signals can be asserted by external devices or by microcontroller modules. Request lines are connected internally by means of a wired NOR — simultaneous requests of differing priority can be made. Internal assertion of an interrupt request signal does not affect the logic state of the corresponding MCU pin.

External interrupt requests are routed to the CPU via the external bus interface and SIM interrupt control logic. The CPU treats external interrupt requests as though they come from the SIM.

External $\overline{\text{IRQ}}$[6:1] are active-low level-sensitive inputs. External $\overline{\text{IRQ7}}$ is an active-low transition-sensitive input. $\overline{\text{IRQ7}}$ requires both an edge and a voltage level for validity.

$\overline{\text{IRQ}}$[6:1] are maskable. $\overline{\text{IRQ7}}$ is nonmaskable. The $\overline{\text{IRQ7}}$ input is transition-sensitive in order to prevent redundant servicing and stack overflow. A nonmaskable interrupt is generated each time $\overline{\text{IRQ7}}$ is asserted, and each time the priority mask changes from %111 to a lower number while $\overline{\text{IRQ7}}$ is asserted.

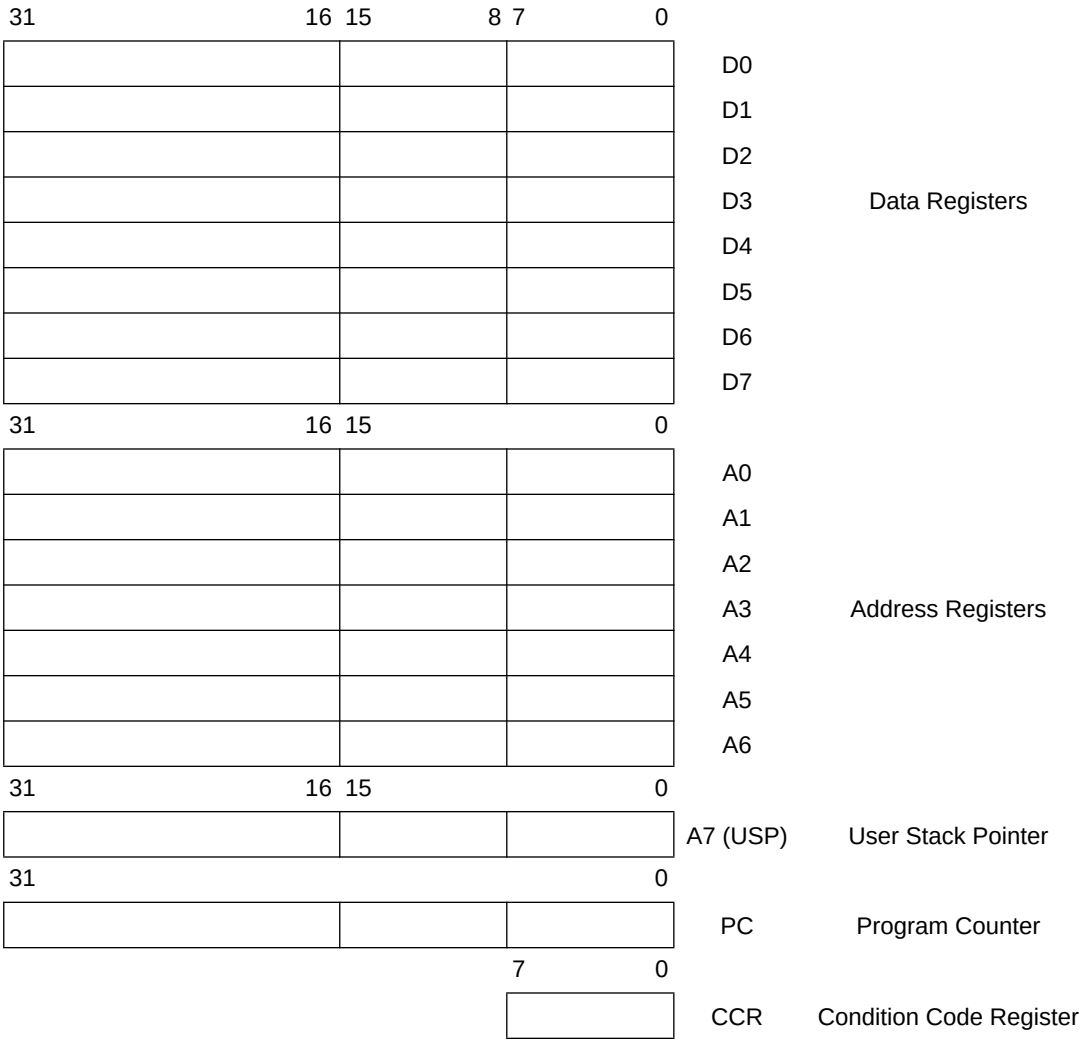


Figure 10 User Programming Model

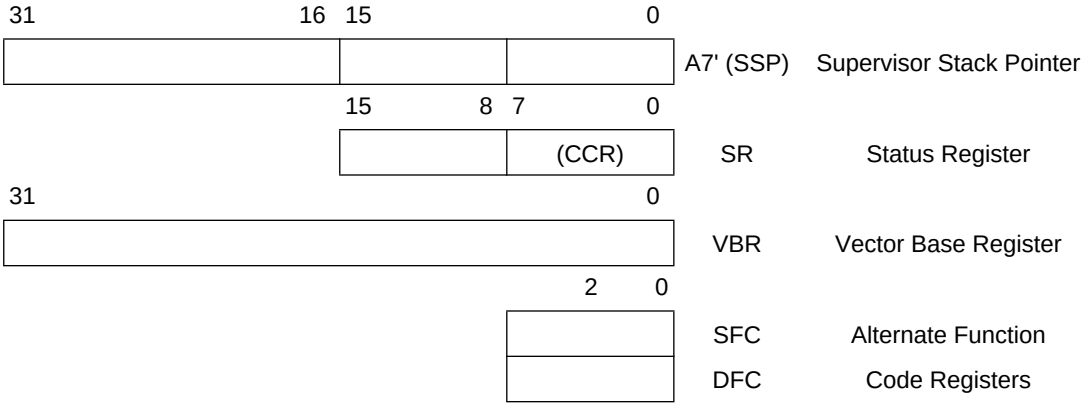
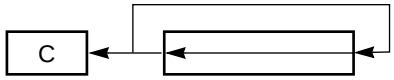
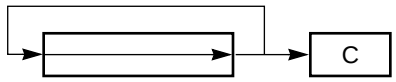
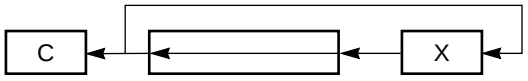
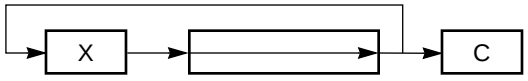


Figure 11 Supervisor Programming Model Supplement

Table 20 Instruction Set Summary (Continued)

Instruction	Syntax	Operand Size	Operation
MOVEP	Dn, (d16, An) (d16, An), Dn	16, 32	Dn [31 : 24] ⇒ (An + d); Dn [23 : 16] ⇒ (An + d + 2); Dn [15 : 8] ⇒ (An + d + 4); Dn [7 : 0] ⇒ (An + d + 6) (An + d) ⇒ Dn [31 : 24]; (An + d + 2) ⇒ Dn [23 : 16]; (An + d + 4) ⇒ Dn [15 : 8]; (An + d + 6) ⇒ Dn [7 : 0]
MOVEQ	#<data>, Dn	8 ⇒ 32	Immediate data ⇒ Destination
MOVES[1]	Rn, <ea> <ea>, Rn	8, 16, 32	Rn ⇒ Destination using DFC Source using SFC ⇒ Rn
MULS/MULU	<ea>, Dn <ea>, Dl <ea>, Dh : Dl	16 ∗ 16 ⇒ 32 32 ∗ 32 ⇒ 32 32 ∗ 32 ⇒ 64	Source ∗ Destination ⇒ Destination (signed or unsigned)
NBCD	<ea>	8 8	0 – Destination$_{10}$ – X ⇒ Destination
NEG	<ea>	8, 16, 32	0 – Destination ⇒ Destination
NEGX	<ea>	8, 16, 32	0 – Destination – X ⇒ Destination
NOP	none	none	PC + 2 ⇒ PC
NOT	<ea>	8, 16, 32	$\overline{\text{Destination}}$ ⇒ Destination
OR	<ea>, Dn Dn, <ea>	8, 16, 32 8, 16, 32	Source + Destination ⇒ Destination
ORI	#<data>, <ea>	8, 16, 32	Data + Destination ⇒ Destination
ORI to CCR	#<data>, CCR	16	Source + CCR ⇒ SR
ORI to SR[1]	#<data>, SR	16	Source ; SR ⇒ SR
PEA	<ea>	32	SP – 4 ⇒ SP; <ea> ⇒ SP
RESET[1]	none	none	Assert $\overline{\text{RESET}}$ line
ROL	Dn, Dn #<data>, Dn <ea>	8, 16, 32 8, 16, 32 16	
ROR	Dn, Dn #<data>, Dn <ea>	8, 16, 32 8, 16, 32 16	
ROXL	Dn, Dn #<data>, Dn <ea>	8, 16, 32 8, 16, 32 16	
ROXR	Dn, Dn #<data>, Dn <ea>	8, 16, 32 8, 16, 32 16	
RTD	#d	16	(SP) ⇒ PC; SP + 4 + d ⇒ SP
RTE[1]	none	none	(SP) ⇒ SR; SP + 2 ⇒ SP; (SP) ⇒ PC; SP + 4 ⇒ SP; Restore stack according to format
RTR	none	none	(SP) ⇒ CCR; SP + 2 ⇒ SP; (SP) ⇒ PC; SP + 4 ⇒ SP
RTS	none	none	(SP) ⇒ PC; SP + 4 ⇒ SP
SBCD	Dn, Dn – (An), – (An)	8 8	Destination10 – Source10 – X ⇒ Destination
Scc	<ea>	8	If condition true, then destination bits are set to 1; else, destination bits are cleared to 0
STOP[1]	#<data>	16	Data ⇒ SR; STOP

5 Queued Serial Module

The QSM contains two serial interfaces, the queued serial peripheral interface (QSPI) and the serial communication interface (SCI).

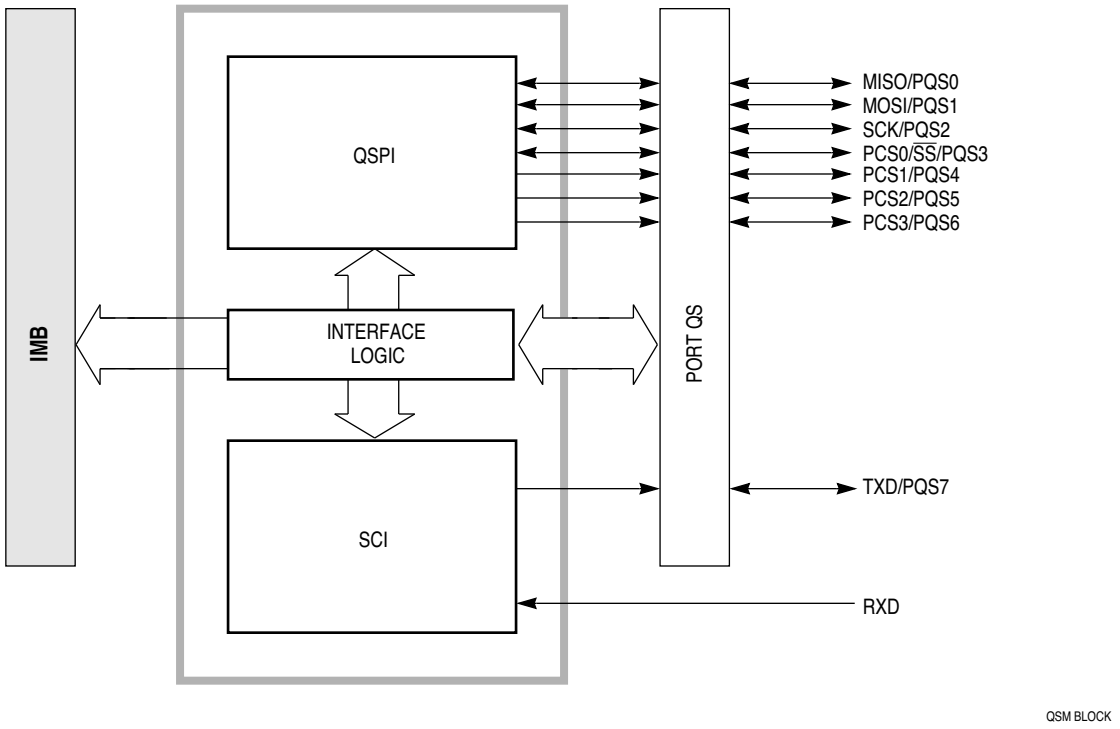


Figure 12 QSM Block Diagram

5.1 Overview

The QSPI provides easy peripheral expansion or interprocessor communication through a full-duplex, synchronous, three-line bus: data in, data out, and a serial clock. Four programmable peripheral chip-select pins provide addressability for up to 16 peripheral devices. A self-contained RAM queue allows up to 16 serial transfers of 8 to 16 bits each, or transmission of a 256-bit data stream without CPU intervention. A special wraparound mode supports continuous sampling of a serial peripheral, with automatic QSPI RAM updating, which makes the interface to A/D converters more efficient.

The SCI provides a standard nonreturn to zero (NRZ) mark/space format. It operates in either full- or half-duplex mode. There are separate transmitter and receiver enable bits and dual data buffers. A modulus-type baud rate generator provides rates from 64 to 524 kbaud with a 16.78-MHz system clock, or 110 to 655 kbaud with a 20.97 MHz system clock. Word length of either 8 or 9 bits is software selectable. Optional parity generation and detection provide either even or odd parity check capability. Advanced error detection circuitry catches glitches of up to 1/16 of a bit time in duration. Wakeup functions allow the CPU to run uninterrupted until meaningful data is available.

An address map of the QSM is shown below. The “Access” column indicates which registers are accessible only at the supervisor privilege level and which can be assigned to either the supervisor or user privilege level, according to the value of the SUPV bit in the QSMCR.

Table 22 QSM Address Map

Access	Address	15 8	7 0
S	$YFFC00	QSM MODULE CONFIGURATION (QSMCR)	
S	$YFFC02	QSM TEST (QTEST)	
S	$YFFC04	QSM INTERRUPT LEVEL (QILR)	QSM INTERRUPT VECTOR (QIVR)
S/U	$YFFC06	NOT USED	
S/U	$YFFC08	SCI CONTROL 0 (SCCR0)	
S/U	$YFFC0A	SCI CONTROL 1 (SCCR1)	
S/U	$YFFC0C	SCI STATUS (SCSR)	
S/U	$YFFC0E	SCI DATA (SCDR)	
S/U	$YFFC10	NOT USED	
S/U	$YFFC12	NOT USED	
S/U	$YFFC14	NOT USED	PQS DATA (PORTQS)
S/U	$YFFC16	PQS PIN ASSIGNMENT (PQSPAR)	PQS DATA DIRECTION (DDRQS)
S/U	$YFFC18	SPI CONTROL 0 (SPCR0)	
S/U	$YFFC1A	SPI CONTROL 1 (SPCR1)	
S/U	$YFFC1C	SPI CONTROL 2 (SPCR2)	
S/U	$YFFC1E	SPI CONTROL 3 (SPCR3)	SPI STATUS (SPSR)
S/U	$YFFC20–$YFFCFF	NOT USED	
S/U	$YFFD00–$YFFD1F	RECEIVE RAM (RR[0:F])	
S/U	$YFFD20–$YFFD3F	TRANSMIT RAM (TR[0:F])	
S/U	$YFFD40–$YFFD4F	COMMAND RAM (CR[0:F])	

Y = M111, where M is the logic state of the module mapping (MM) bit in the SIMCR

5.2 Pin Function

The following table is a summary of the functions of the QSM pins when they are not configured for general-purpose I/O. The QSM data direction register (DDRQS) designates each pin except RXD as an input or output.

	Pin	Mode	Pin Function
QSPI Pins	MISO	Master	Serial Data Input to QSPI
		Slave	Serial Data Output from QSPI
	MOSI	Master	Serial Data Output from QSPI
		Slave	Serial Data Input to QSPI
	SCK	Master	Clock Output from QSPI
		Slave	Clock Input to QSPI
	PCS0/$\overline{SS}$	Master	Input: Assertion Causes Mode Fault Output: Selects Peripherals
		Slave	Input: Selects the QSPI
	PCS[3:1]	Master	Output: Selects Peripherals
		Slave	None
SCI Pins	TXD	Transmit	Serial Data Output from SCI
	RXD	Receive	Serial Data Input to SCI

PORTQS — Port QS Data Register **$YFFC14**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	PQS7	PQS6	PQS5	PQS4	PQS3	PQS2	PQS1	PQS0
	RESET:							
	0	0	0	0	0	0	0	0

PORTQS latches I/O data. Writes drive pins defined as outputs. Reads return data present on the pins. To avoid driving undefined data, first write a byte to PORTQS, then configure DDRQS.

PQSPAR — PORT QS Pin Assignment Register **$YFFC16**
DDRQS — PORT QS Data Direction Register **$YFFC17**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	PQSPA6	PQSPA5	PQSPA4	PQSPA3	0	PQSPA1	PQSPA0	DDQS7	DDQS6	DDQS5	DDQS4	DDQS3	DDQS2	DDQS1	DDQS0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Clearing a bit in the PQSPAR assigns the corresponding pin to general-purpose I/O; setting a bit assigns the pin to the QSPI. The PQSPAR does not affect operation of the SCI.

Table 23 QSPAR Pin Assignments

PQSPAR Field	PQSPAR Bit	Pin Function
PQSPA0	0 1	PQS0 MISO
PQSPA1	0 1	PQS1 MOSI
PQSPA2	0 1	PQS2[1] SCK
PQSPA3	0 1	PQS3 PCS0/$\overline{SS}$
PQSPA4	0 1	PQS4 PCS1
PQSPA5	0 1	PQS5 PCS2
PQSPA6	0 1	PQS6 PCS3
PQSPA7	0 1	PQS7[2] TXD

NOTES:
1. PQS2 is a digital I/O pin unless the SPI is enabled (SPE in SPCR1 set), in which case it becomes SPI serial clock SCK.
2. PQS7 is a digital I/O pin unless the SCI transmitter is enabled (TE in SCCR1 = 1), in which case it becomes SCI serial output TXD.

6 General-Purpose Timer Module

The 11-channel general-purpose timer (GPT) is used in systems where a moderate level of CPU control is required. The GPT consists of a capture/compare unit, a pulse accumulator, and two pulse-width modulators. A bus interface unit connects the GPT to the intermodule bus.

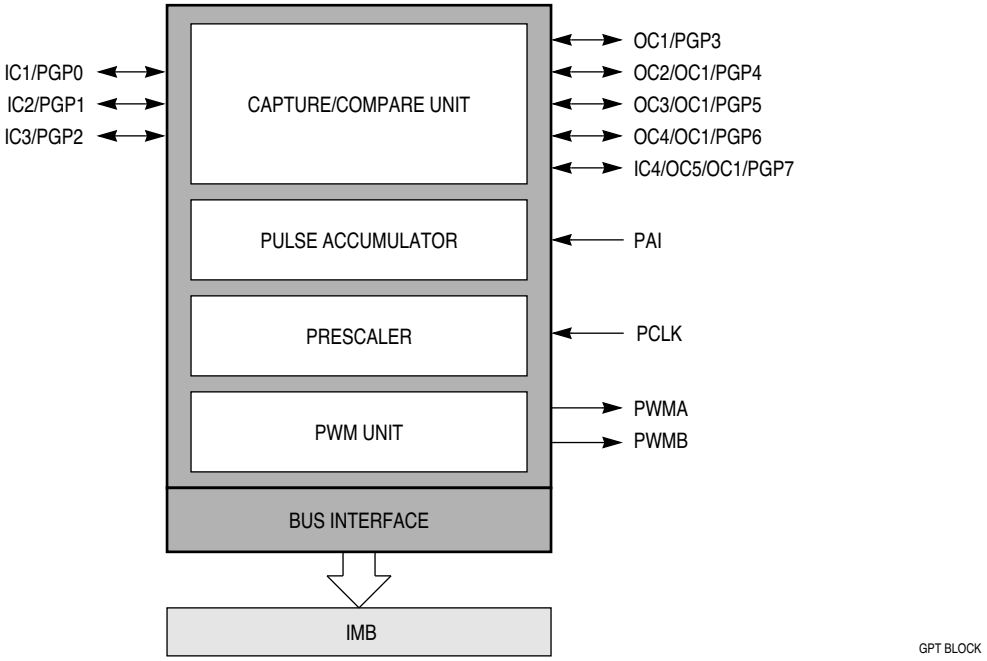


Figure 15 GPT Block Diagram

6.1 Overview

The capture/compare unit features three input capture channels, four output compare channels, and one channel that can be selected as an input capture or output compare channel. These channels share a 16-bit free-running counter (TCNT) which derives its clock from a nine-stage prescaler or from the external clock input signal, PCLK.

Pulse accumulator channel logic includes an 8-bit counter; the pulse accumulator can operate in either event counting mode or gated time accumulation mode.

Pulse-width modulator outputs are periodic waveforms whose duty cycles can be independently selected and modified by user software. The PWM circuits share a 16-bit free-running counter that can be clocked by the same nine-stage prescaler used by the capture/compare unit or by the PCLK input.

All GPT pins can also be used for general-purpose input/output. The input capture and output compare pins form a bidirectional 8-bit parallel port (PORTGP). PWM pins are outputs only. PAI and PCLK pins are inputs only.

GPT input capture/output compare pins are bidirectional and can be used to form an 8-bit parallel port. The pulse-width modulator outputs can be used as general-purpose outputs. The PAI and PCLK inputs can be used as general-purpose inputs.

The GPT control register address map is shown below. The “Access” column in the GPT address map indicates which registers are accessible only at the supervisor privilege level and which can be assigned to either the supervisor or user privilege level, according to the value of the SUPV bit in the GPTMCR.

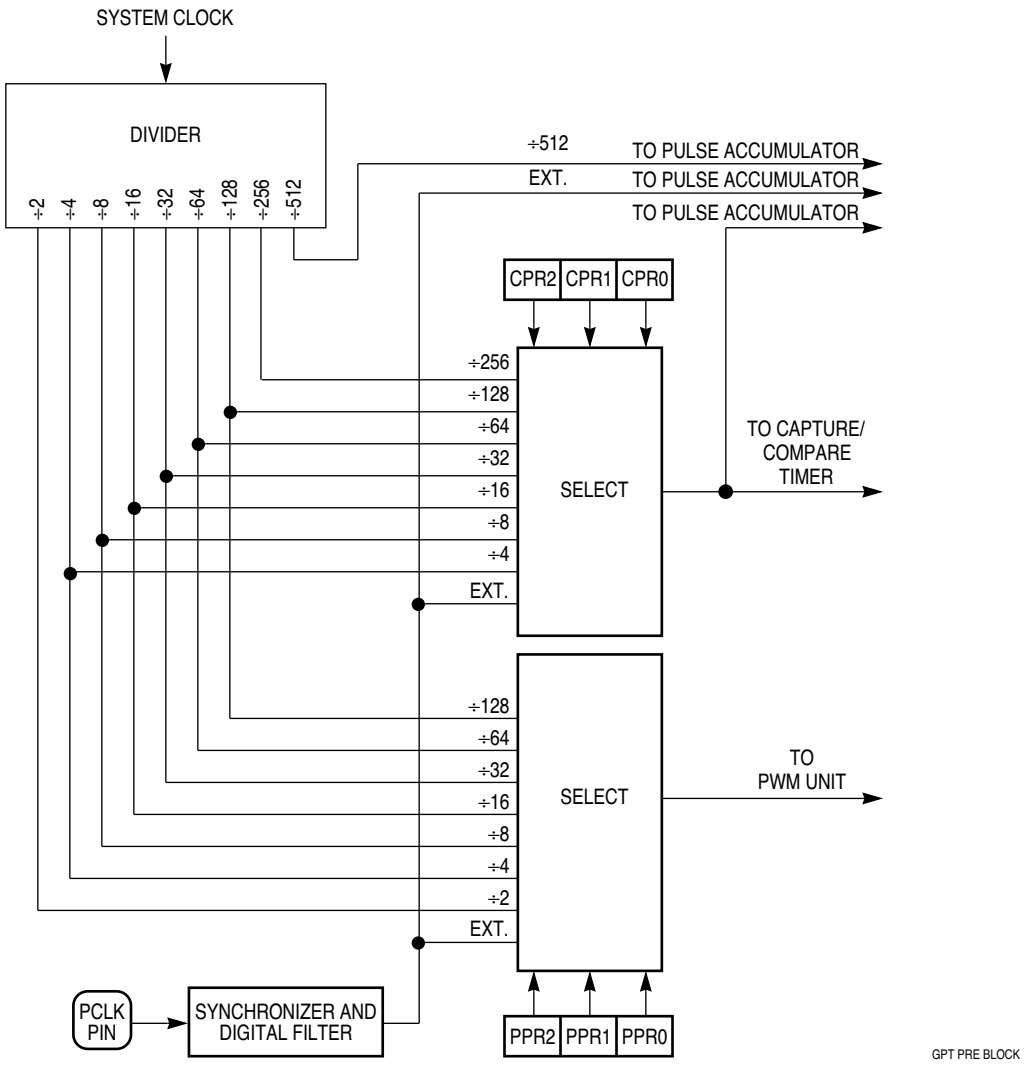


Figure 17 Prescaler Block Diagram

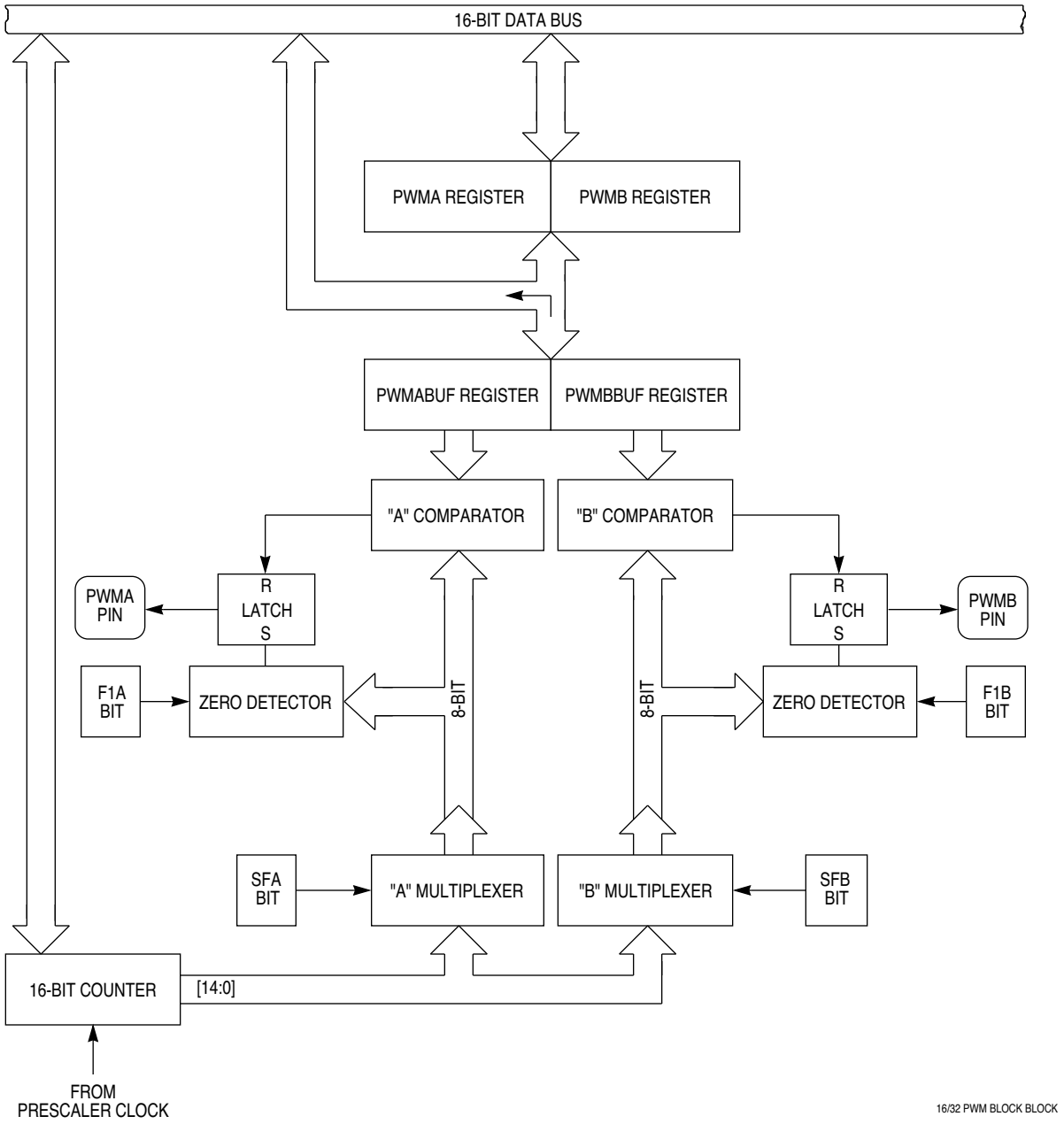


Figure 18 PWM Unit Block Diagram

6.3 Pulse-Width Modulator

The pulse-width modulation submodule has two output pins. The outputs are periodic waveforms controlled by a single frequency whose duty cycles can be independently selected and modified by user software. Each PWM can be independently programmed to run in fast or slow mode. The PWM unit has its own 16-bit free-running counter, which is clocked by an output of the nine-stage prescaler (the same prescaler used by the capture/compare unit) or by the clock input pin, PCLK.

F1B — Force Logic Level One on PWMB
0 = Force logic level zero output on PWMB pin
1 = Force logic level one output on PWMB pin

PWMA/PWMB — PWM Control Registers A/B **$YFF926, $YFF927**

These registers are associated with the pulse-width value of the PWM output on the corresponding PWM pin. A value of $00 loaded into one of these registers results in a continuously low output on the corresponding pin. A value of $80 results in a 50% duty cycle output. Maximum value ($FF) selects an output that is high for 255/256 of the period.

PWMCNT — PWM Count Register **$YFF928**

PWMCNT is the 16-bit free-running counter associated with the PWM functions of the GPT module.

PWMBUFA/B — PWM Buffer Registers A/B **$YFF92A, $YFF92B**

These read-only registers contain values associated with the duty cycles of the corresponding PWM. Reset state is $0000.

PRESCL — GPT Prescaler **$YFF92C**

The 9-bit prescaler value can be read from bits [8:0] at this address. Bits [15:9] always read as zeros. Reset state is $0000.