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	Active
Core Processor	CPU32
Core Size	32-Bit Single-Core
Speed	20MHz
Connectivity	EBI/EMI, SCI, SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	15
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°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/pro/item?MUrl=&PartUrl=mc68332aveh20

Table 1 Ordering Information

Package Type	TPU Type	Temperature	Frequency (MHz)	Package Order Quantity	Order Number
132-Pin PQFP	Motion Control	–40 to +85 °C	16 MHz	2 pc tray	SPAKMC332GCFC16
				36 pc tray	MC68332GCFC16
			20 MHz	2 pc tray	SPAKMC332GCFC20
				36 pc tray	MC68332GCFC20
		–40 to +105 °C	16 MHz	2 pc tray	SPAKMC332GVFC16
				36 pc tray	MC68332GVFC16
			20 MHz	2 pc tray	SPAKMC332GVFC20
				36 pc tray	MC68332GVFC20
		–40 to +125 °C	16 MHz	2 pc tray	SPAKMC332GMFC16
				36 pc tray	MC68332GMFC16
			20 MHz	2 pc tray	SPAKMC332GMFC20
				36 pc tray	MC68332GMFC20
	Standard	–40 to +85 °C	16 MHz	2 pc tray	SPAKMC332CFC16
				36 pc tray	MC68332CFC16
			20 MHz	2 pc tray	SPAKMC332CFC20
				36 pc tray	MC68332CFC20
		–40 to +105 °C	16 MHz	2 pc tray	SPAKMC332VFC16
				36 pc tray	MC68332VFC16
			20 MHz	2 pc tray	SPAKMC332VFC20
				36 pc tray	MC68332VFC20
		–40 to +125 °C	16 MHz	2 pc tray	SPAKMC332MFC16
				36 pc tray	MC68332MFC16
			20 MHz	2 pc tray	SPAKMC332MFC20
				36 pc tray	MC68332MFC20
	Std w/enhanced PPWA	–40 to +85 °C	16 MHz	2 pc tray	SPAKMC332ACFC16
				36 pc tray	MC68332ACFC16
			20 MHz	2 pc tray	SPAKMC332ACFC20
				36 pc tray	MC68332ACFC20
		–40 to +105 °C	16 MHz	2 pc tray	SPAKMC332AVFC16
				36 pc tray	MC68332AVFC16
			20 MHz	2 pc tray	SPAKMC332AVFC20
				36 pc tray	MC68332AVFC20
		–40 to +125 °C	16 MHz	2 pc tray	SPAKMC332AMFC16
				36 pc tray	MC68332AMFC16
			20 MHz	2 pc tray	SPAKMC332AMFC20
				36 pc tray	MC68332AMFC20

1.1 Features

- Central Processing Unit (CPU32)
  - — 32-Bit Architecture
  - — Virtual Memory Implementation
  - — Table Lookup and Interpolate Instruction
  - — Improved Exception Handling for Controller Applications
  - — High-Level Language Support
  - — Background Debugging Mode
  - — Fully Static Operation
- System Integration Module (SIM)
  - — External Bus Support
  - — Programmable Chip-Select Outputs
  - — System Protection Logic
  - — Watchdog Timer, Clock Monitor, and Bus Monitor
  - — Two 8-Bit Dual Function Input/Output Ports
  - — One 7-Bit Dual Function Output Port
  - — Phase-Locked Loop (PLL) Clock System
- Time Processor Unit (TPU)
  - — Dedicated Microengine Operating Independently of CPU32
  - — 16 Independent, Programmable Channels and Pins
  - — Any Channel can Perform any Time Function
  - — Two Timer Count Registers with Programmable Prescalers
  - — Selectable Channel Priority Levels
- Queued Serial Module (QSM)
  - — Enhanced Serial Communication Interface
  - — Queued Serial Peripheral Interface
  - — One 8-Bit Dual Function Port
- Static RAM Module with TPU Emulation Capability (TPURAM)
  - — 2-Kbytes of Static RAM
  - — May be Used as Normal RAM or TPU Microcode Emulation RAM

1.2 Block Diagram

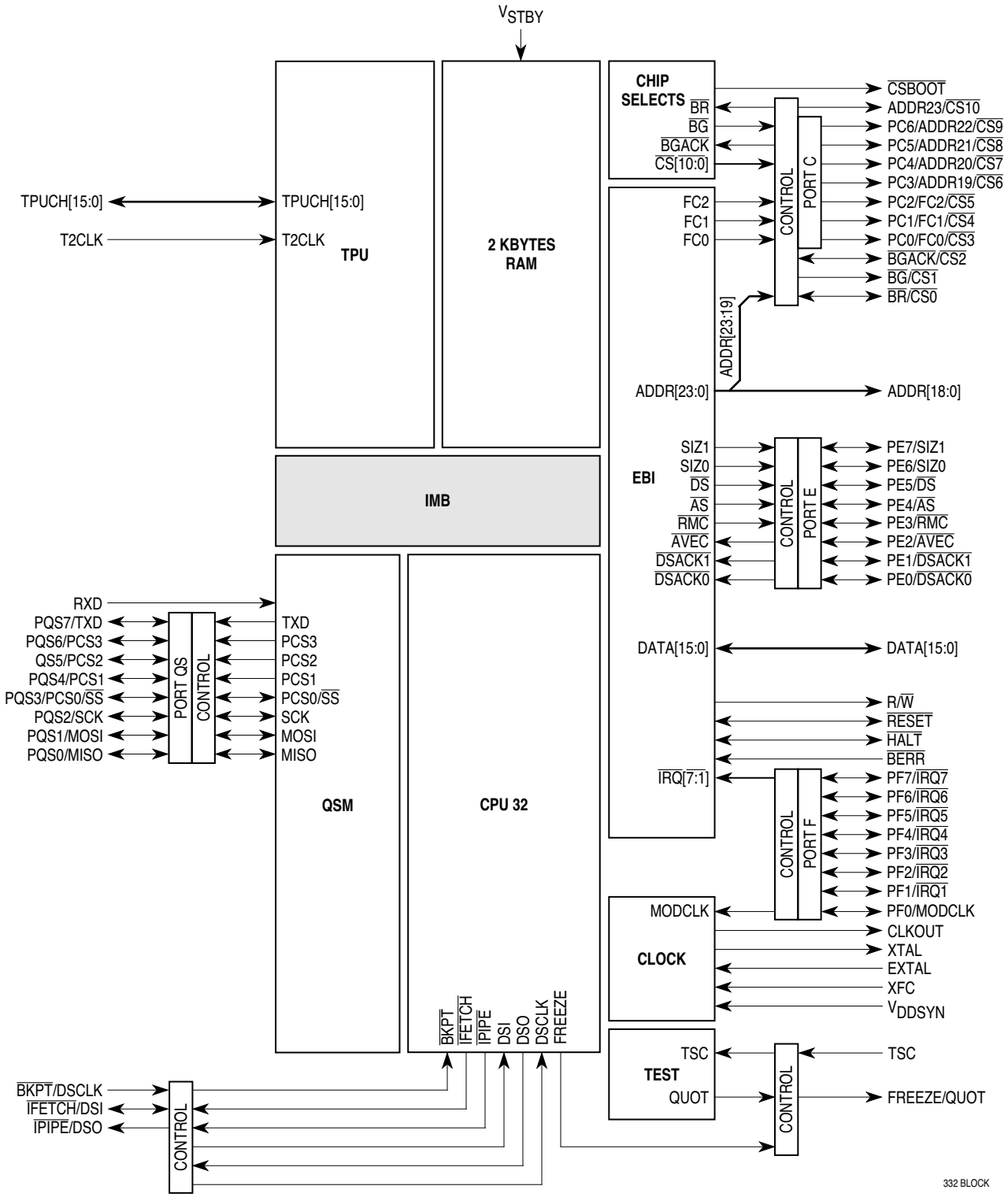


Figure 1 MCU Block Diagram

Table 6 MCU Signal Function (Continued)

Signal Name	Mnemonic	Function
Quotient Out	QUOT	Provides the quotient bit of the polynomial divider
Reset	$\overline{\text{RESET}}$	System reset
Read-Modify-Write Cycle	$\overline{\text{RMC}}$	Indicates an indivisible read-modify-write instruction
Read/Write	R/$\overline{\text{W}}$	Indicates the direction of data transfer on the bus
SCI Receive Data	RXD	Serial input to the SCI
QSPI Serial Clock	SCK	Clock output from QSPI in master mode; clock input to QSPI in slave mode
Size	SIZ[1:0]	Indicates the number of bytes to be transferred during a bus cycle
Slave Select	$\overline{\text{SS}}$	Causes serial transmission when QSPI is in slave mode; causes mode fault in master mode
TCR2 Clock	T2CLK	External clock source for TCR2 counter
TPU Channel Pins	TPUCH[15:0]	Bidirectional pins associated with TPU channels
Three-State Control	TSC	Places all output drivers in a high-impedance state
SCI Transmit Data	TXD	Serial output from the SCI
External Filter Capacitor	XFC	Connection for external phase-locked loop filter capacitor

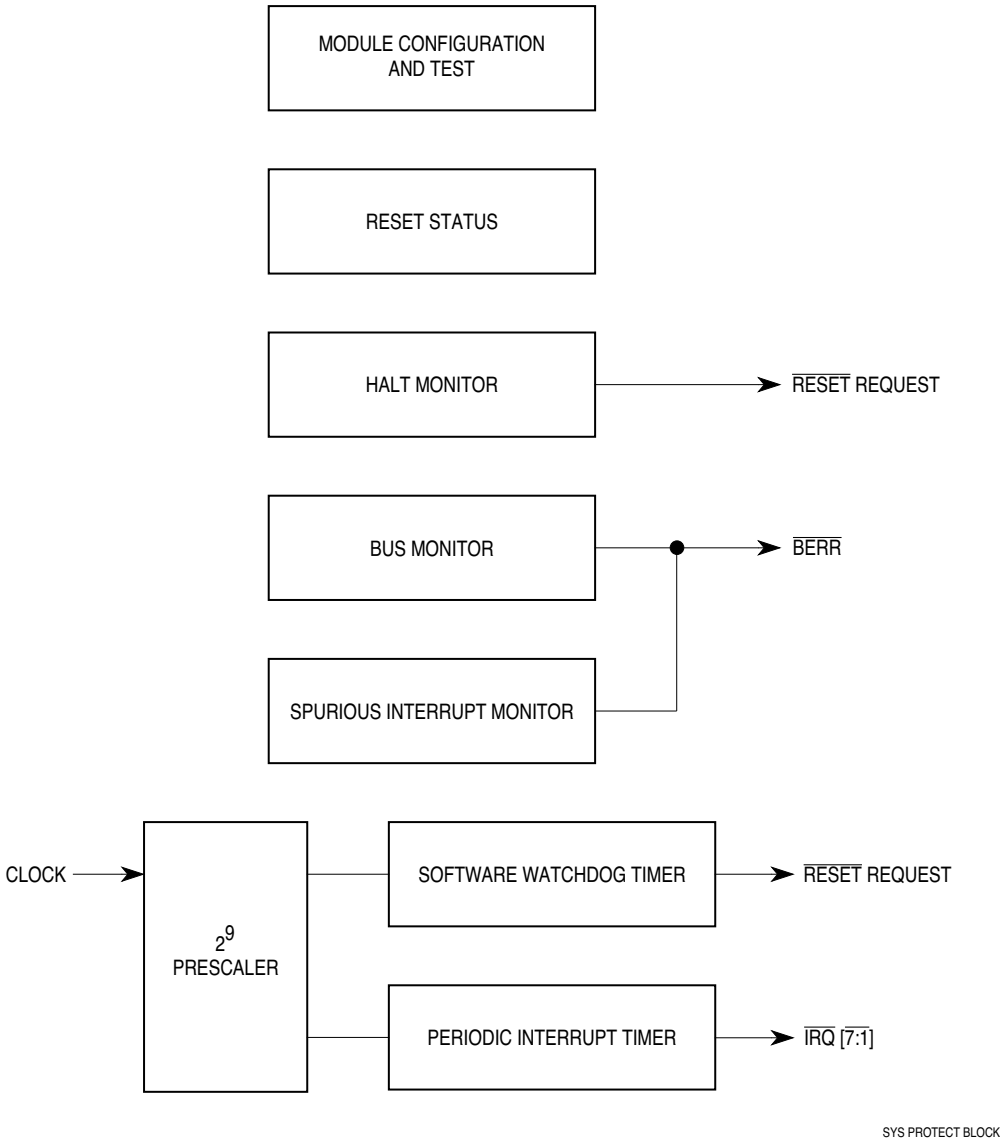


Figure 6 System Configuration and Protection Block

3.2.1 System Configuration

The SIM controls MCU configuration during normal operation and during internal testing.

SIMCR —SIM Configuration Register **$YFFA00**

15	14	13	12	11	10	9	8	7	6	5	4	3	0
EXOFF	FRZSW	FRZBM	0	SLVEN	0	SHEN		SUPV	MM	0	0	IARB	
RESET:													
0	0	0	0	DATA11	0	0	0	1	1	0	0	1	1	1	1

The SIM configuration register controls system configuration. It can be read or written at any time, except for the module mapping (MM) bit, which can be written only once.

EXOFF — External Clock Off

0 = The CLKOUT pin is driven from an internal clock source.
1 = The CLKOUT pin is placed in a high-impedance state.

FRZSW — Freeze Software Enable

0 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters continue to run.
1 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters are disabled, preventing interrupts during software debug.

FRZBM — Freeze Bus Monitor Enable

0 = When FREEZE is asserted, the bus monitor continues to operate.
1 = When FREEZE is asserted, the bus monitor is disabled.

SLVEN — Factory Test Mode Enabled

This bit is a read-only status bit that reflects the state of DATA11 during reset.

0 = IMB is not available to an external master.
1 = An external bus master has direct access to the IMB.

SHEN[1:0] — Show Cycle Enable

This field determines what the EBI does with the external bus during internal transfer operations. A show cycle allows internal transfers to be externally monitored. The table below shows whether show cycle data is driven externally, and whether external bus arbitration can occur. To prevent bus conflict, external peripherals must not be enabled during show cycles.

SHEN	Action
00	Show cycles disabled, external arbitration enabled
01	Show cycles enabled, external arbitration disabled
10	Show cycles enabled, external arbitration enabled
11	Show cycles enabled, external arbitration enabled, internal activity is halted by a bus grant

SUPV — Supervisor/Unrestricted Data Space

The SUPV bit places the SIM global registers in either supervisor or user data space.

0 = Registers with access controlled by the SUPV bit are accessible from either the user or supervisor privilege level.
1 = Registers with access controlled by the SUPV bit are restricted to supervisor access only.

MM — Module Mapping

0 = Internal modules are addressed from $7FF000 –$7FFFFF.
1 = Internal modules are addressed from $FFF000 –$FFFFFF.

IARB[3:0] — Interrupt Arbitration Field

Each module that can generate interrupt requests has an interrupt arbitration (IARB) field. Arbitration between interrupt requests of the same priority is performed by serial contention between IARB field bit values. Contention must take place whenever an interrupt request is acknowledged, even when there is only a single pending request. An IARB field must have a non-zero value for contention to take place. If an interrupt request from a module with an IARB field value of %0000 is recognized, the CPU processes a spurious interrupt exception. Because the SIM routes external interrupt requests to the CPU, the SIM IARB field value is used for arbitration between internal and external interrupts of the same priority. The reset value of IARB for the SIM is %1111, and the reset IARB value for all other modules is %0000, which prevents SIM interrupts from being discarded during initialization.

3.2.3 Bus Monitor

The internal bus monitor checks for excessively long $\overline{\text{DSACK}}$ response times during normal bus cycles and for excessively long $\overline{\text{DSACK}}$ or $\overline{\text{AVEC}}$ response times during interrupt acknowledge cycles. The monitor asserts $\overline{\text{BERR}}$ if response time is excessive.

$\overline{\text{DSACK}}$ and $\overline{\text{AVEC}}$ response times are measured in clock cycles. The maximum allowable response time can be selected by setting the BMT field.

The monitor does not check $\overline{\text{DSACK}}$ response on the external bus unless the CPU initiates the bus cycle. The BME bit in the SYPCR enables the internal bus monitor for internal to external bus cycles. If a system contains external bus masters, an external bus monitor must be implemented and the internal to external bus monitor option must be disabled.

3.2.4 Halt Monitor

The halt monitor responds to an assertion of $\overline{\text{HALT}}$ on the internal bus. A flag in the reset status register (RSR) indicates that the last reset was caused by the halt monitor. The halt monitor reset can be inhibited by the HME bit in the SYPCR.

3.2.5 Spurious Interrupt Monitor

The spurious interrupt monitor issues $\overline{\text{BERR}}$ if no interrupt arbitration occurs during an interrupt-acknowledge cycle.

3.2.6 Software Watchdog

The software watchdog is controlled by SWE in the SYPCR. Once enabled, the watchdog requires that a service sequence be written to SWSR on a periodic basis. If servicing does not take place, the watchdog times out and issues a reset. This register can be written at any time, but returns zeros when read.

SWSR —Software Service Register **$YFFA27**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	0	0	0	0	0	0	0	0
RESET:	0	0	0	0	0	0	0	0

Register shown with read value

Perform a software watchdog service sequence as follows:

a. Write $55 to SWSR.
b. Write $AA to SWSR.

Both writes must occur before time-out in the order listed, but any number of instructions can be executed between the two writes.

The watchdog clock rate is affected by SWP and SWT in SYPCR. When SWT[1:0] are modified, a watchdog service sequence must be performed before the new time-out period takes effect.

The reset value of SWP is affected by the state of the MODCLK pin on the rising edge of reset, as shown in the following table.

MODCLK	SWP
0	1
1	0

3.3.3 Clock Control

The clock control circuits determine system clock frequency and clock operation under special circumstances, such as following loss of synthesizer reference or during low-power operation. Clock source is determined by the logic state of the MODCLK pin during reset.

SYNCR —Clock Synthesizer Control Register **$YFFA04**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W	X	Y						EDIV	0	0	SLIMP	SLOCK	RSTEN	STSIM	STEXT
RESET:															
0	0	1	1	1	1	1	1	0	0	0	U	U	0	0	0

When the on-chip clock synthesizer is used, system clock frequency is controlled by the bits in the upper byte of SYNCR. Bits in the lower byte show status of or control operation of internal and external clocks. The SYNCR can be read or written only when the CPU is operating at the supervisor privilege level.

W — Frequency Control (VCO)
This bit controls a prescaler tap in the synthesizer feedback loop. Setting the bit increases the VCO speed by a factor of four. VCO relock delay is required.

X — Frequency Control Bit (Prescale)
This bit controls a divide by two prescaler that is not in the synthesizer feedback loop. Setting the bit doubles clock speed without changing the VCO speed. There is no VCO relock delay.

Y[5:0] — Frequency Control (Counter)
The Y field controls the modulus down counter in the synthesizer feedback loop, causing it to divide by a value of Y + 1. Values range from 0 to 63. VCO relock delay is required.

EDIV — E Clock Divide Rate
0 = ECLK frequency is system clock divided by 8.
1 = ECLK frequency is system clock divided by 16.
ECLK is an external M6800 bus clock available on pin ADDR23. Refer to **3.5 Chip Selects** for more information.

SLIMP — Limp Mode Flag
0 = External crystal is VCO reference.
1 = Loss of crystal reference.
When the on-chip synthesizer is used, loss of reference frequency causes SLIMP to be set. The VCO continues to run using the base control voltage. Maximum limp frequency is maximum specified system clock frequency. X-bit state affects limp frequency.

SLOCK — Synthesizer Lock Flag
0 = VCO is enabled, but has not locked.
1 = VCO has locked on the desired frequency (or system clock is external).
The MCU maintains reset state until the synthesizer locks, but SLOCK does not indicate synthesizer lock status until after the user writes to SYNCR.

RSTEN — Reset Enable
0 = Loss of crystal causes the MCU to operate in limp mode.
1 = Loss of crystal causes system reset.

STSIM — Stop Mode SIM Clock
0 = When LPSTOP is executed, the SIM clock is driven from the crystal oscillator and the VCO is turned off to conserve power.
1 = When LPSTOP is executed, the SIM clock is driven from the VCO.

STEXT — Stop Mode External Clock
0 = When LPSTOP is executed, the CLKOUT signal is held negated to conserve power.
1 = When LPSTOP is executed, the CLKOUT signal is driven from the SIM clock, as determined by the state of the STSIM bit.

3.4.8 Data Transfer Mechanism

The MCU architecture supports byte, word, and long-word operands, allowing access to 8- and 16-bit data ports through the use of asynchronous cycles controlled by the data transfer and size acknowledge inputs ($\overline{\text{DSACK1}}$ and $\overline{\text{DSACK0}}$).

3.4.9 Dynamic Bus Sizing

The MCU dynamically interprets the port size of the addressed device during each bus cycle, allowing operand transfers to or from 8- and 16-bit ports. During an operand transfer cycle, the slave device signals its port size and indicates completion of the bus cycle to the MCU through the use of the $\overline{\text{DSACK0}}$ and $\overline{\text{DSACK1}}$ inputs, as shown in the following table.

Table 10 Effect of $\overline{\text{DSACK}}$ Signals

$\overline{\text{DSACK1}}$	$\overline{\text{DSACK0}}$	Result
1	1	Insert Wait States in Current Bus Cycle
1	0	Complete Cycle —Data Bus Port Size is 8 Bits
0	1	Complete Cycle —Data Bus Port Size is 16 Bits
0	0	Reserved

For example, if the MCU is executing an instruction that reads a long-word operand from a 16-bit port, the MCU latches the 16 bits of valid data and then runs another bus cycle to obtain the other 16 bits. The operation for an 8-bit port is similar, but requires four read cycles. The addressed device uses the $\overline{\text{DSACK0}}$ and $\overline{\text{DSACK1}}$ signals to indicate the port width. For instance, a 16-bit device always returns $\overline{\text{DSACK0}}$ = 1 and $\overline{\text{DSACK1}}$ = 0 for a 16-bit port, regardless of whether the bus cycle is a byte or word operation.

Dynamic bus sizing requires that the portion of the data bus used for a transfer to or from a particular port size be fixed. A 16-bit port must reside on data bus bits [15:0] and an 8-bit port must reside on data bus bits [15:8]. This minimizes the number of bus cycles needed to transfer data and ensures that the MCU transfers valid data.

The MCU always attempts to transfer the maximum amount of data on all bus cycles. For a word operation, it is assumed that the port is 16 bits wide when the bus cycle begins. Operand bytes are designated as shown in the following figure. OP0 is the most significant byte of a long-word operand, and OP3 is the least significant byte. The two bytes of a word-length operand are OP0 (most significant) and OP1. The single byte of a byte-length operand is OP0.

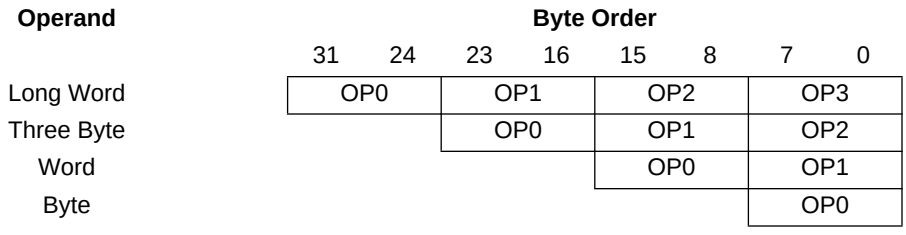


Figure 8 Operand Byte Order

3.4.10 Operand Alignment

The data multiplexer establishes the necessary connections for different combinations of address and data sizes. The multiplexer takes the two bytes of the 16-bit bus and routes them to their required positions. Positioning of bytes is determined by the size and address outputs. SIZ1 and SIZ0 indicate the remaining number of bytes to be transferred during the current bus cycle. The number of bytes transferred is equal to or less than the size indicated by SIZ1 and SIZ0, depending on port width.

3.5.3 Base Address Registers

A base address is the starting address for the block enabled by a given chip select. Block size determines the extent of the block above the base address. Each chip select has an associated base register so that an efficient address map can be constructed for each application. If a chip-select base address register is programmed with the same address as a microcontroller module or memory array, an access to that address goes to the module or array and the chip-select signal is not asserted.

CSBARBT — Chip-Select Base Address Register Boot ROM **$YFFA48**

15	14	13	12	11	10	9	8	7	6	5	4	3	2		0
ADDR 23	ADDR 22	ADDR 21	ADDR 20	ADDR 19	ADDR 18	ADDR 17	ADDR 16	ADDR 15	ADDR 14	ADDR 13	ADDR 12	ADDR 11	BLKSZ		
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

CSBAR[10:0] —Chip-Select Base Address Registers **$YFFA4C–$YFFA74**

15	14	13	12	11	10	9	8	7	6	5	4	3	2		0
ADDR 23	ADDR 22	ADDR 21	ADDR 20	ADDR 19	ADDR 18	ADDR 17	ADDR 16	ADDR 15	ADDR 14	ADDR 13	ADDR 12	ADDR 11	BLKSZ		
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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.

BLKSZ — Block Size Field

This field determines the size of the block that must be enabled by the chip select. The following table shows bit encoding for the base address registers block size field.

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]

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.

Table 18 Reset Mode Selection

DATA0	$\overline{\text{CSBOOT}}$ 16-Bit	$\overline{\text{CSBOOT}}$ 8-Bit
DATA1	$\overline{\text{CS0}}$ $\overline{\text{CS1}}$ $\overline{\text{CS2}}$	$\overline{\text{BR}}$ $\overline{\text{BG}}$ $\overline{\text{BGACK}}$
DATA2	$\overline{\text{CS3}}$ $\overline{\text{CS4}}$ $\overline{\text{CS5}}$	FC0 FC1 FC2
DATA3 DATA4 DATA5 DATA6 DATA7	$\overline{\text{CS6}}$ $\overline{\text{CS}}[7:6]$ $\overline{\text{CS}}[8:6]$ $\overline{\text{CS}}[9:6]$ $\overline{\text{CS}}[10:6]$	ADDR19 ADDR[20:19] ADDR[21:19] ADDR[22:19] ADDR[23:19]
DATA8	$\overline{\text{DSACK0}}$, $\overline{\text{DSACK1}}$, $\overline{\text{AVEC}}$, $\overline{\text{DS}}$, $\overline{\text{AS}}$, $\overline{\text{SIZ}}[1:0]$	PORTE
DATA9	$\overline{\text{IRQ}}[7:1]$ MODCLK	PORTF
DATA11	Test Mode Disabled	Test Mode Enabled
MODCLK	VCO = System Clock	EXTAL = System Clock
$\overline{\text{BKPT}}$	Background Mode Disabled	Background Mode Enabled

3.7.2 Functions of Pins for Other Modules During Reset

Generally, pins associated with modules other than the SIM default to port functions, and input/output ports are set to input state. This is accomplished by disabling pin functions in the appropriate control registers, and by clearing the appropriate port data direction registers. Refer to individual module sections in this manual for more information. The following table is a summary of module pin function out of reset.

Table 19 Module Pin Functions

Module	Pin Mnemonic	Function
CPU32	DSI/$\overline{\text{IFETCH}}$	DSI/$\overline{\text{IFETCH}}$
	DSO/$\overline{\text{IPIPE}}$	DSO/$\overline{\text{IPIPE}}$
	$\overline{\text{BKPT}}$/DSCLK	$\overline{\text{BKPT}}$/DSCLK
GPT	PGP7/IC4/OC5	Discrete Input
	PGP[6:3]/OC[4:1]	Discrete Input
	PGP[2:0]/IC[3:1]	Discrete Input
	PAI	Discrete Input
	PCLK	Discrete Input
	PWMA, PWMB	Discrete Output
QSM	PQS7/TXD	Discrete Input
	PQS[6:4]/PCS[3:1]	Discrete Input
	PQS3/PCS0/$\overline{\text{SS}}$	Discrete Input
	PQS2/SCK	Discrete Input
	PQS1/MOSI	Discrete Input
	PQS0/MISO	Discrete Input
	RXD	RXD

4.3 Status Register

The status register contains the condition codes that reflect the results of a previous operation and can be used for conditional instruction execution in a program. The lower byte containing the condition codes is the only portion of the register available at the user privilege level; it is referenced as the condition code register (CCR) in user programs. At the supervisor privilege level, software can access the full status register, including the interrupt priority mask and additional control bits.

SR —Status Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T1	T0	S	0	0	IP			0	0	0	X	N	Z	V	C
RESET:															
0	0	1	0	0	1	1	1	0	0	0	U	U	U	U	U

System Byte
- T[1:0] —Trace Enable
- S —Supervisor/User State
- Bits [12:11] —Unimplemented
- IP[2:0] —Interrupt Priority Mask

User Byte (Condition Code Register)
- Bits [7:5] — Unimplemented
- X — Extend
- N — Negative
- Z — Zero
- V — Overflow
- C — Carry

4.4 Data Types

Six basic data types are supported:

- Bits
- Packed Binary Coded Decimal Digits
- Byte Integers (8 bits)
- Word Integers (16 bits)
- Long-Word Integers (32 bits)
- Quad-Word Integers (64 bits)

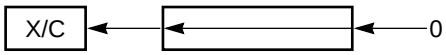
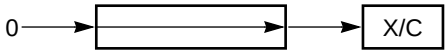
4.5 Addressing Modes

Addressing in the CPU32 is register-oriented. Most instructions allow the results of the specified operation to be placed either in a register or directly in memory. This flexibility eliminates the need for extra instructions to store register contents in memory. The CPU32 supports seven basic addressing modes:

- Register direct
- Register indirect
- Register indirect with index
- Program counter indirect with displacement
- Program counter indirect with index
- Absolute
- Immediate

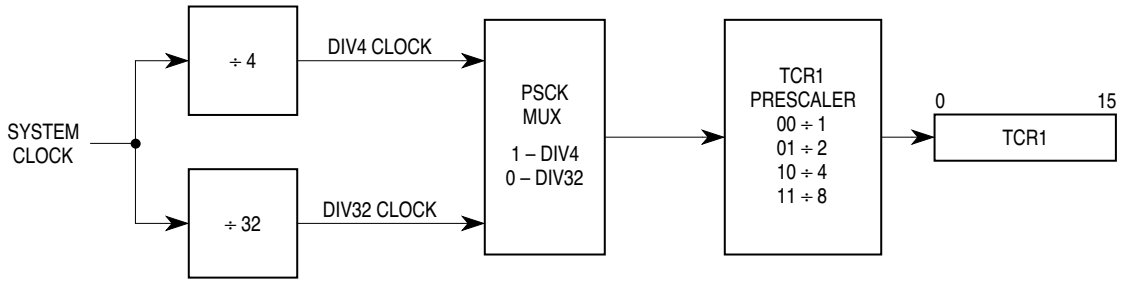
Included in the register indirect addressing modes are the capabilities to post-increment, predecrement, and offset. The program counter relative mode also has index and offset capabilities. In addition to these addressing modes, many instructions implicitly specify the use of the status register, stack pointer, or program counter.

Table 20 Instruction Set Summary(Continued)

Instruction	Syntax	Operand Size	Operation
DBcc	Dn, label	16	If condition false, then Dn – 1 ⇒ PC; if Dn ≠ (– 1), then PC + d ⇒ PC
DIVS/DIVU	<ea>, Dn	32/16 ⇒ 16 : 16	Destination / Source ⇒ Destination (signed or unsigned)
DIVSL/DIVUL	<ea>, Dr : Dq <ea>, Dq <ea>, Dr : Dq	64/32 ⇒ 32 : 32 32/32 ⇒ 32 32/32 ⇒ 32 : 32	Destination / Source ⇒ Destination (signed or unsigned)
EOR	Dn, <ea>	8, 16, 32	Source ⊕ Destination ⇒ Destination
EORI	# <data>, <ea>	8, 16, 32	Data ⊕ Destination ⇒ Destination
EORI to CCR	# <data>, CCR	8	Source ⊕ CCR ⇒ CCR
EORI to SR[1]	# <data>, SR	16	Source ⊕ SR ⇒ SR
EXG	Rn, Rn	32	Rn ⇒ Rn
EXT	Dn Dn	8 ⇒ 16 16 ⇒ 32	Sign extended Destination ⇒ Destination
EXTB	Dn	8 ⇒ 32	Sign extended Destination ⇒ Destination
ILLEGAL	none	none	SSP – 2 ⇒ SSP; vector offset ⇒ (SSP); SSP – 4 ⇒ SSP; PC ⇒ (SSP); SSP – 2 ⇒ SSP; SR ⇒ (SSP); Illegal instruction vector address ⇒ PC
JMP	Í	none	Destination ⇒ PC
JSR	Í	none	SP – 4 ⇒ SP; PC ⇒ (SP); destination ⇒ PC
LEA	<ea>, An	32	<ea> ⇒ An
LINK	An, # d	16, 32	SP – 4 ⇒ SP, An ⇒ (SP); SP ⇒ An, SP + d ⇒ SP
LPSTOP[1]	# <data>	16	Data ⇒ SR; interrupt mask ⇒ EBI; STOP
LSL	Dn, Dn # <data>, Dn Í	8, 16, 32 8, 16, 32 16	
LSR	Dn, Dn #<data>, Dn Í	8, 16, 32 8, 16, 32 16	
MOVE	<ea>, <ea>	8, 16, 32	Source ⇒ Destination
MOVEA	<ea>, An	16, 32 ⇒ 32	Source ⇒ Destination
MOVEA[1]	USP, An An, USP	32 32	USP ⇒ An An ⇒ USP
MOVE from CCR	CCR, <ea>	16	CCR ⇒ Destination
MOVE to CCR	<ea>, CCR	16	Source ⇒ CCR
MOVE from SR[1]	SR, <ea>	16	SR ⇒ Destination
MOVE to SR[1]	<ea>, SR	16	Source ⇒ SR
MOVE USP[1]	USP, An An, USP	32 32	USP ⇒ An An ⇒ USP
MOVEC[1]	Rc, Rn Rn, Rc	32 32	Rc ⇒ Rn Rn ⇒ Rc
MOVEM	list, <ea> <ea>, list	16, 32 16, 32 ⇒ 32	Listed registers ⇒ Destination Source ⇒ Listed registers
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

TCR1P — Timer Count Register 1 Prescaler Control

TCR1 is clocked from the output of a prescaler. The prescaler's input is the internal TPU system clock divided by either 4 or 32, depending on the value of the PSCK bit. The prescaler divides this input by 1, 2, 4, or 8. Channels using TCR1 have the capability to resolve down to the TPU system clock divided by 4.

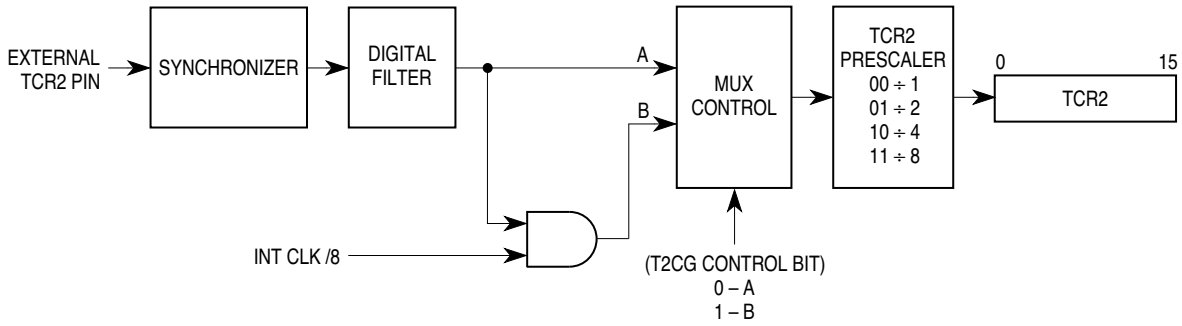


PRESCALER CTL BLOCK 1

TCR1 Prescaler	Divide By	PSCK = 0		PSCK = 1	
		Number of Clocks	Rate at 16 MHz	Number of Clocks	Rate at 16 MHz
00	1	32	2 ms	4	250 ns
01	2	64	4 ms	8	500 ns
10	4	128	8 ms	16	1 ms
11	8	256	16 ms	32	2 ms

TCR2P — Timer Count Register 2 Prescaler Control

TCR2 is clocked from the output of a prescaler. If T2CG = 0, the input to the TCR2 prescaler is the external TCR2 clock source. If T2CG = 1, the input is the TPU system clock divided by eight. The TCR2P field specifies the value of the prescaler: 1, 2, 4, or 8. Channels using TCR2 have the capability to resolve down to the TPU system clock divided by 8. The following table is a summary of prescaler output.



PRESCALER CTL BLOCK 2

TCR2 Prescaler	Divide By	Internal Clock Divided By	External Clock Divided By
00	1	8	1
01	2	16	2
10	4	32	4
11	8	64	8

5.5.2 Channel Control Registers

CIER — Channel Interrupt Enable Register **$YFFE0A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 15	CH 14	CH 13	CH 12	CH 11	CH 10	CH 9	CH 8	CH 7	CH 6	CH 5	CH 4	CH 3	CH 2	CH 1	CH 0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CH[15:0] — Channel Interrupt Enable/Disable
0 = Channel interrupts disabled
1 = Channel interrupts enabled

CISR — Channel Interrupt Status Register **$YFFE20**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 15	CH 14	CH 13	CH 12	CH 11	CH 10	CH 9	CH 8	CH 7	CH 6	CH 5	CH 4	CH 3	CH 2	CH 1	CH 0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CH[15:0] — Channel Interrupt Status Bit
0 = Channel interrupt not asserted
1 = Channel interrupt asserted

CFSR0 — Channel Function Select Register 0 **$YFFE0C**

15	12	11	8	7	4	3	0
CHANNEL15		CHANNEL14		CHANNEL13		CHANNEL12	
RESET:							
0	0	0	0	0	0	0	0

CFSR1 — Channel Function Select Register 1 **$YFFE0E**

15	12	11	8	7	4	3	0
CHANNEL11		CHANNEL10		CHANNEL9		CHANNEL8	
RESET:							
0	0	0	0	0	0	0	0

CFSR2 — Channel Function Select Register 2 **$YFFE10**

15	12	11	8	7	4	3	0
CHANNEL7		CHANNEL6		CHANNEL5		CHANNEL4	
RESET:							
0	0	0	0	0	0	0	0

CFSR3 — Channel Function Select Register 3 **$YFFE12**

15	12	11	8	7	4	3	0
CHANNEL3		CHANNEL2		CHANNEL1		CHANNEL0	
RESET:							
0	0	0	0	0	0	0	0

CHANNEL[15:0] — Encoded Time Function for each Channel
Encoded 4-bit fields in the channel function select registers specify one of 16 time functions to be executed on the corresponding channel.

HSQR0 — Host Sequence Register 0 **$YFFE14**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 15		CH 14		CH 13		CH 12		CH 11		CH 10		CH 9		CH 8	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

HSQR1 — Host Sequence Register 1 **$YFFE16**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 7		CH 6		CH 5		CH 4		CH 3		CH 2		CH 1		CH 0	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CH[15:0] — Encoded Host Sequence

The host sequence field selects the mode of operation for the time function selected on a given channel. The meaning of the host sequence bits depends on the time function specified.

HSRR0 — Host Service Request Register 0 **$YFFE18**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 15		CH 14		CH 13		CH 12		CH 11		CH 10		CH 9		CH 8	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

HSRR1 — Host Service Request Register 1 **$YFFE1A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 7		CH 6		CH 5		CH 4		CH 3		CH 2		CH 1		CH 0	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CH[15:0] — Encoded Type of Host Service

The host service request field selects the type of host service request for the time function selected on a given channel. The meaning of the host service request bits depends on the time function specified. A host service request field cleared to %00 signals the host that service is completed by the microengine on that channel. The host can request service on a channel by writing the corresponding host service request field to one of three nonzero states. The CPU should monitor the host service request register until the TPU clears the service request to %00 before the CPU changes any parameters or issues a new service request to the channel.

CPR0 — Channel Priority Register 0 **$YFFE1C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 15		CH 14		CH13		CH 12		CH 11		CH 10		CH 9		CH 8	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CPR1 — Channel Priority Register 1 **$YFFE1E**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH 7		CH 6		CH 5		CH 4		CH 3		CH 2		CH 1		CH 0	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CH[15:0] — Encoded One of Three Channel Priority Levels

MSTR — Master/Slave Mode Select

0 = QSPI is a slave device and only responds to externally generated serial data.

1 = QSPI is system master and can initiate transmission to external SPI devices.

MSTR configures the QSPI for either master or slave mode operation. This bit is cleared on reset and may only be written by the CPU.

WOMQ — Wired-OR Mode for QSPI Pins

0 = Outputs have normal MOS drivers.

1 = Pins designated for output by DDRQS have open-drain drivers.

WOMQ allows the wired-OR function to be used on QSPI pins, regardless of whether they are used as general-purpose outputs or as QSPI outputs. WOMQ affects the QSPI pins regardless of whether the QSPI is enabled or disabled.

BITS — Bits Per Transfer

In master mode, when BITSE in a command is set, the BITS field determines the number of data bits transferred. When BITSE is cleared, eight bits are transferred. Reserved values default to eight bits. BITSE is not used in slave mode.

The following table shows the number of bits per transfer.

BITS	Bits per Transfer
0000	16
0001	Reserved
0010	Reserved
0011	Reserved
0100	Reserved
0101	Reserved
0110	Reserved
0111	Reserved
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

CPOL — Clock Polarity

0 = The inactive state value of SCK is logic level zero.

1 = The inactive state value of SCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SCK). It is used with CPHA to produce a desired clock/data relationship between master and slave devices.

CPHA — Clock Phase

0 = Data is captured on the leading edge of SCK and changed on the following edge of SCK.

1 = Data is changed on the leading edge of SCK and captured on the following edge of SCK.

CPHA determines which edge of SCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce a desired clock/data relationship between master and slave devices. CPHA is set at reset.

SPBR — Serial Clock Baud Rate

The QSPI uses a modulus counter to derive SCK baud rate from the MCU system clock. Baud rate is selected by writing a value from 2 to 255 into the SPBR field. The following equation determines the

SCK baud rate:

SCK Baud Rate = System Clock/(2SPBR)

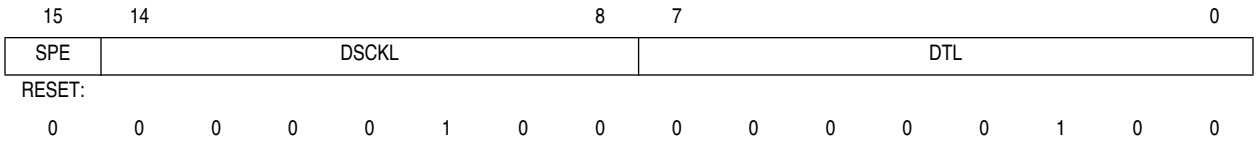
or

SPBR = System Clock/(2SCK)(Baud Rate Desired)

where SPBR equals {2, 3, 4,..., 255}

Giving SPBR a value of zero or one disables the baud rate generator. SCK is disabled and assumes its inactive state value. No serial transfers occur. At reset, baud rate is initialized to one eighth of the system clock frequency.

SPCR1 — QSPI Control Register 1 **$YFFC1A**



SPCR1 contains parameters for configuring the QSPI before it is enabled. The CPU can read and write this register, but the QSM has read access only, except for SPE, which is automatically cleared by the QSPI after completing all serial transfers, or when a mode fault occurs.

SPE — QSPI Enable

0 = QSPI is disabled. QSPI pins can be used for general-purpose I/O.
1 = QSPI is enabled. Pins allocated by PQSPAR are controlled by the QSPI.

DSCKL — Delay before SCK

When the DSCK bit in command RAM is set, this field determines the length of delay from PCS valid to SCK transition. PCS can be any of the four peripheral chip-select pins. The following equation determines the actual delay before SCK:

PCS to SCK Delay = [DSCKL/System Clock]

where DSCKL equals {1, 2, 3,..., 127}.

When the DSCK value of a queue entry equals zero, then DSCKL is not used. Instead, the PCS valid-to-SCK transition is one-half SCK period.

DTL — Length of Delay after Transfer

When the DT bit in command RAM is set, this field determines the length of delay after serial transfer. The following equation is used to calculate the delay:

Delay after Transfer = [(32DTL)/System Clock]

where DTL equals {1, 2, 3,..., 255}.

A zero value for DTL causes a delay-after-transfer value of 8192/System Clock.

If DT equals zero, a standard delay is inserted.

Standard Delay after Transfer = [17/System Clock]

Delay after transfer can be used to provide a peripheral deselect interval. A delay can also be inserted between consecutive transfers to allow serial A/D converters to complete conversion.

Command RAM is used by the QSPI when in master mode. The CPU writes one byte of control information to this segment for each QSPI command to be executed. The QSPI cannot modify information in command RAM.

Command RAM consists of 16 bytes. Each byte is divided into two fields. The peripheral chip-select field enables peripherals for transfer. The command control field provides transfer options.

A maximum of 16 commands can be in the queue. Queue execution by the QSPI proceeds from the address in NEWQP through the address in ENDQP. (Both of these fields are in SPCR2.)

CONT — Continue

0 = Control of chip selects returned to PORTQS after transfer is complete.
1 = Peripheral chip selects remain asserted after transfer is complete.

BITSE — Bits per Transfer Enable

0 = 8 bits
1 = Number of bits set in BITS field of SPCR0

DT — Delay after Transfer

The QSPI provides a variable delay at the end of serial transfer to facilitate the interface with peripherals that have a latency requirement. The delay between transfers is determined by the SPCR1 DTL field.

DSCK — PCS to SCK Delay

0 = PCS valid to SCK transition is one-half SCK.
1 = SPCR1 DSCKL field specifies delay from PCS valid to SCK.

PCS[3:0] — Peripheral Chip Select

Use peripheral chip-select bits to select an external device for serial data transfer. More than one peripheral chip select can be activated at a time, and more than one peripheral chip can be connected to each PCS pin, provided that proper fanout is observed.

$\overline{SS}$ — Slave Mode Select

Initiates slave mode serial transfer. If $\overline{SS}$ is taken low when the QSPI is in master mode, a mode fault will be generated.

6.5.4 Operating Modes

The QSPI operates in either master or slave mode. Master mode is used when the MCU originates data transfers. Slave mode is used when an external device initiates serial transfers to the MCU through the QSPI. Switching between the modes is controlled by MSTR in SPCR0. Before entering either mode, appropriate QSM and QSPI registers must be properly initialized.

In master mode, the QSPI executes a queue of commands defined by control bits in each command RAM queue entry. Chip-select pins are activated, data is transmitted from transmit RAM and received into receive RAM.

In slave mode, operation proceeds in response to $\overline{SS}$ pin activation by an external bus master. Operation is similar to master mode, but no peripheral chip selects are generated, and the number of bits transferred is controlled in a different manner. When the QSPI is selected, it automatically executes the next queue transfer to exchange data with the external device correctly.

Although the QSPI inherently supports multimaster operation, no special arbitration mechanism is provided. A mode fault flag (MODF) indicates a request for SPI master arbitration. System software must provide arbitration. Note that unlike previous SPI systems, MSTR is not cleared by a mode fault being set, nor are the QSPI pin output drivers disabled. The QSPI and associated output drivers must be disabled by clearing SPE in SPCR1.

6.6 SCI Submodule

The SCI submodule is used to communicate with external devices through an asynchronous serial bus. The SCI is fully compatible with the SCI systems found on other Motorola MCUs, such as the M68HC11 and M68HC05 Families.

6.6.1 SCI Pins

There are two unidirectional pins associated with the SCI. The SCI controls the transmit data (TXD) pin when enabled, whereas the receive data (RXD) pin remains a dedicated input pin to the SCI. TXD is available as a general-purpose I/O pin when the SCI transmitter is disabled. When used for I/O, TXD can be configured either as input or output, as determined by QSM register DDRQS.

The following table shows SCI pins and their functions.

Pin Names	Mnemonics	Mode	Function
Receive Data	RXD	Receiver Disabled Receiver Enabled	Not Used Serial Data Input to SCI
Transmit Data	TXD	Transmitter Disabled Transmitter Enabled	General-Purpose I/O Serial Data Output from SCI

6.6.2 SCI Registers

The SCI programming model includes QSM global and pin control registers, and four SCI registers. There are two SCI control registers, one status register, and one data register. All registers can be read or written at any time by the CPU.

Changing the value of SCI control bits during a transfer operation may disrupt operation. Before changing register values, allow the transmitter to complete the current transfer, then disable the receiver and transmitter. Status flags in the SCSR may be cleared at any time.

SCCR0 — SCI Control Register 0 **$YFFC08**

15	14	13	12												0
0	0	0	SCBR												
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

SCCR0 contains a baud rate selection parameter. Baud rate must be set before the SCI is enabled. The CPU can read and write this register at any time.

Bits [15:13] — Not Implemented

SCBR — Baud Rate

SCI baud rate is programmed by writing a 13-bit value to BR. The baud rate is derived from the MCU system clock by a modulus counter.

The SCI receiver operates asynchronously. An internal clock is necessary to synchronize with an incoming data stream. The SCI baud rate generator produces a receiver sampling clock with a frequency 16 times that of the expected baud rate of the incoming data. The SCI determines the position of bit boundaries from transitions within the received waveform, and adjusts sampling points to the proper positions within the bit period. Receiver sampling rate is always 16 times the frequency of the SCI baud rate, which is calculated as follows:

$$\text{SCI Baud Rate} = \text{System Clock}/(32\text{SCBR})$$

or

$$\text{SCBR} = \text{System Clock}(32\text{SCK})(\text{Baud Rate desired})$$

where SCBR is in the range {1, 2, 3, ..., 8191}