



Welcome to [E-XFL.COM](https://www.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	Not For New Designs
Core Processor	CPU32
Core Size	32-Bit Single-Core
Speed	25MHz
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	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68331cag25

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/CS10/ECLK	A	Y	N	O	—
ADDR[22:19]/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	—	—
$\overline{BG/CS1}$	B	—	—	—	—
$\overline{BGACK/CS2}$	B	Y	N	—	—
$\overline{BKPT/DSCLK}$	—	Y	Y	—	—
$\overline{BR/CS0}$	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
$\overline{DSI/IFETCH}$	A	Y	Y	—	—
$\overline{DSO/IPIPE}$	A	—	—	—	—
EXTAL ²	—	—	Special	—	—
FC[2:0]/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	—
$\overline{PCS0/SS}$	Bo	Y	Y	I/O	PQS3
$\overline{PCS[3:1]}$	Bo	Y	Y	I/O	PQS[6:4]
PWMA, PWMB	A	—	—	O	—
R/W	A	Y	N	—	—
$\overline{RESET}$	Bo	Y	Y	—	—
$\overline{RMC}$	B	Y	N	I/O	PE3

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 –\$7FFFFFFF.

1 = Internal modules are addressed from \$FFF000 –\$FFFFFFF.

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.

3.2.2 System Protection Control Register

The system protection control register controls system monitor functions, software watchdog clock prescaling, and bus monitor timing. This register can be written only once following power-on or reset, but can be read at any time.

SYPCR —System Protection Control Register

\$YFFA21

15	8	7	6	5	4	3	2	1	0
NOT USED								SWE	BMT

RESET:

1 MODCLK 0 0 0 0 0 0

SWE —Software Watchdog Enable

0 = Software watchdog disabled

1 = Software watchdog enabled

SWP —Software Watchdog Prescale

This bit controls the value of the software watchdog prescaler.

- 0 = Software watchdog clock not prescaled
- 1 = Software watchdog clock prescaled by 512

SWT[1:0] —Software Watchdog Timing

This field selects the divide ratio used to establish software watchdog time-out period. The following table gives the ratio for each combination of SWP and SWT bits.

SWP	SWT	Ratio
0	00	2^9
0	01	2^{11}
0	10	2^{13}
0	11	2^{15}
1	00	2^{18}
1	01	2^{20}
1	10	2^{22}
1	11	2^{24}

HME —Halt Monitor Enable

- 0 = Disable halt monitor function
- 1 = Enable halt monitor function

BME —Bus Monitor External Enable

- 0 = Disable bus monitor function for an internal to external bus cycle.
- 1 = Enable bus monitor function for an internal to external bus cycle.

BMT[1:0] —Bus Monitor Timing

This field selects a bus monitor time-out period as shown in the following table.

BMT	Bus Monitor Time-out Period
00	64 System Clocks
01	32 System Clocks
10	16 System Clocks
11	8 System Clocks

3.2.3 Bus Monitor

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

$\overline{DSACK}$ and $\overline{AVEC}$ 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 $\overline{DSACK}$ 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 $\overline{HALT}$ 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 0 0

Register shown with read value

Perform a software watchdog service sequence as follows:

1. Write \$55 to SWSR.
2. 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

3.2.7 Periodic Interrupt Timer

The periodic interrupt timer (PIT) generates interrupts of specified priorities at specified intervals. Timing for the PIT is provided by a programmable prescaler driven by the system clock.

PICR — Periodic Interrupt Control Register

\$YFFA22

15	14	13	12	11	10	8	7	0
0	0	0	0	0	PIRQL	PIV		

RESET:

0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1

This register contains information concerning periodic interrupt priority and vectoring. Bits [10:0] can be read or written at any time. Bits [15:11] are unimplemented and always return zero.

PIRQL[2:0] — Periodic Interrupt Request Level

The following table shows what interrupt request level is asserted when a periodic interrupt is generated. If a PIT interrupt and an external $\overline{\text{IRQ}}$ signal of the same priority occur simultaneously, the PIT interrupt is serviced first. The periodic timer continues to run when the interrupt is disabled.

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 External Bus Interface

The external bus interface (EBI) transfers information between the internal MCU bus and external devices. The external bus has 24 address lines and 16 data lines.

The EBI provides dynamic sizing between 8-bit and 16-bit data accesses. It supports byte, word, and long-word transfers. Ports are accessed through the use of asynchronous cycles controlled by the data transfer (SIZ1 and SIZ0) and data size acknowledge pins (DSACK1 and DSACK0). Multiple bus cycles may be required for a transfer to or from an 8-bit port.

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 $\overline{DSACK}$ or autovector support.

CSORBT —Chip-Select Option Register Boot ROM

\$YFFA4A

15	14	13	12	11	10	9	6	5	4	3	1	0
MODE	BYTE	R/W	STRB	DSACK	SPACE	IPL	AVEC					
RESET:												
0	1	1	1	1	0	1	1	0	1	1	1	0

CSOR[10:0] —Chip-Select Option Registers

\$YFFA4E–\$YFFA76

15	14	13	12	11	10	9	6	5	4	3	1	0
MODE	BYTE	R/W	STRB	DSACK	SPACE	IPL	AVEC					
RESET:												
0	0	0	0	0	0	0	0	0	0	0	0	0

CSORBT, the option register for $\overline{CSBOOT}$, 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 $\overline{AS}$ or $\overline{DS}$.

The $\overline{DSACK}$ 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.

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 $\overline{IRQ}[7:1]$ and the IP mask value. Each of the signals corresponds to an interrupt priority. $\overline{IRQ}1$ has the lowest priority, and $\overline{IRQ}7$ 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 $\overline{IRQ}7$) 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 $\overline{IRQ}[6:1]$ are active-low level-sensitive inputs. External $\overline{IRQ}7$ is an active-low transition-sensitive input. $\overline{IRQ}7$ requires both an edge and a voltage level for validity.

$\overline{IRQ}[6:1]$ are maskable. $\overline{IRQ}7$ is nonmaskable. The $\overline{IRQ}7$ input is transition-sensitive in order to prevent redundant servicing and stack overflow. A nonmaskable interrupt is generated each time $\overline{IRQ}7$ is asserted, and each time the priority mask changes from %111 to a lower number while $\overline{IRQ}7$ is asserted.

Interrupt requests are sampled on consecutive falling edges of the system clock. Interrupt request input circuitry has hysteresis. To be valid, a request signal must be asserted for at least two consecutive clock periods. Valid requests do not cause immediate exception processing, but are left pending. Pending requests are processed at instruction boundaries or when exception processing of higher-priority exceptions is complete.

The CPU32 does not latch the priority of a pending interrupt request. If an interrupt source of higher priority makes a service request while a lower priority request is pending, the higher priority request is serviced. If an interrupt request of equal or lower priority than the current IP mask value is made, the CPU does not recognize the occurrence of the request in any way.

3.8.1 Interrupt Acknowledge and Arbitration

Interrupt acknowledge bus cycles are generated during exception processing. When the CPU detects one or more interrupt requests of a priority higher than the interrupt priority mask value, it performs a CPU space read from address \$FFFF : [IP] : 1.

The CPU space read cycle performs two functions: it places a mask value corresponding to the highest priority interrupt request on the address bus, and it acquires an exception vector number from the interrupt source. The mask value also serves two purposes: it is latched into the CCR IP field in order to mask lower-priority interrupts during exception processing, and it is decoded by modules that have requested interrupt service to determine whether the current interrupt acknowledge cycle pertains to them.

Modules that have requested interrupt service decode the IP value placed on the address bus at the beginning of the interrupt acknowledge cycle, and if their requests are at the specified IP level, respond to the cycle. Arbitration between simultaneous requests of the same priority is performed by means of serial contention between module interrupt arbitration (IARB) field bit values.

Each module that can make an interrupt service request, including the SIM, has an IARB field in its configuration register. An IARB field can be assigned a value from %0001 (lowest priority) to %1111 (highest priority). A value of %0000 in an IARB field causes the CPU to process a spurious interrupt exception when an interrupt from that module is recognized.

Because the EBI manages external interrupt requests, the SIM IARB value is used for arbitration between internal and external interrupt requests. The reset value of IARB for the SIM is %1111, and the reset IARB value for all other modules is %0000. Initialization software must assign different IARB values in order to implement an arbitration scheme.

Each module must have a unique IARB value. When two or more IARB fields have the same nonzero value, the CPU interprets multiple vector numbers simultaneously, with unpredictable consequences.

Arbitration must always take place, even when a single source requests service. This point is important for two reasons: the CPU interrupt acknowledge cycle is not driven on the external bus unless the SIM wins contention, and failure to contend causes an interrupt acknowledge bus cycle to be terminated by a bus error, which causes a spurious interrupt exception to be taken.

When arbitration is complete, the dominant module must place an interrupt vector number on the data bus and terminate the bus cycle. In the case of an external interrupt request, because the interrupt acknowledge cycle is transferred to the external bus, an external device must decode the mask value and respond with a vector number, then generate bus cycle termination signals. If the device does not respond in time, a spurious interrupt exception is taken.

The periodic interrupt timer (PIT) in the SIM can generate internal interrupt requests of specific priority at predetermined intervals. By hardware convention, PIT interrupts are serviced before external interrupt service requests of the same priority. Refer to **3.2.7 Periodic Interrupt Timer** for more information.

3.8.2 Interrupt Processing Summary

A summary of the interrupt processing sequence follows. When the sequence begins, a valid interrupt service request has been detected and is pending.

- A. The CPU finishes higher priority exception processing or reaches an instruction boundary.
- B. Processor state is stacked. The contents of the status register and program counter are saved.
- C. The interrupt acknowledge cycle begins:
 1. FC[2:0] are driven to %111 (CPU space) encoding.
 2. The address bus is driven as follows. ADDR[23:20] = %1111; ADDR[19:16] = %1111, which indicates that the cycle is an interrupt acknowledge CPU space cycle; ADDR[15:4] = %111111111111; ADDR[3:1] = the level of the interrupt request being acknowledged; and ADDR0 = %1.
 3. Request priority level is latched into the IP field in the status register from the address bus.
- D. Modules or external peripherals that have requested interrupt service decode the request level in ADDR[3:1]. If the request level of at least one interrupting module or device is the same as the value in ADDR[3:1], interrupt arbitration contention takes place. When there is no contention, the spurious interrupt monitor asserts $\overline{\text{BERR}}$, and a spurious interrupt exception is processed.
- E. After arbitration, the interrupt acknowledge cycle can be completed in one of three ways:
 1. The dominant interrupt source supplies a vector number and $\overline{\text{DSACK}}$ signals appropriate to the access. The CPU32 acquires the vector number.
 2. The $\overline{\text{AVEC}}$ signal is asserted (the signal can be asserted by the dominant interrupt source or the pin can be tied low), and the CPU32 generates an autovector number corresponding to interrupt priority.
 3. The bus monitor asserts $\overline{\text{BERR}}$ and the CPU32 generates the spurious interrupt vector number.
- F. The vector number is converted to a vector address.
- G. The content of the vector address is loaded into the PC, and the processor transfers control to the exception handler routine.

3.9 Factory Test Block

The test submodule supports scan-based testing of the various MCU modules. It is integrated into the SIM to support production testing.

Test submodule registers are intended for Motorola use. Register names and addresses are provided to indicate that these addresses are occupied.

SIMTR — System Integration Module Test Register	\$YFFA02
SIMTRE — System Integration Module Test Register (E Clock)	\$YFFA08
TSTMSRA — Master Shift Register A	\$YFFA30
TSTMSRB — Master Shift Register B	\$YFFA32
TSTSC — Test Module Shift Count	\$YFFA34
TSTRC — Test Module Repetition Count	\$YFFA36
CREG — Test Module Control Register	\$YFFA38
DREG — Test Module Distributed Register	\$YFFA3A

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	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

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.

4.6 Instruction Set Summary

Table 20 Instruction Set Summary

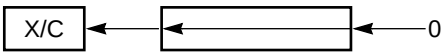
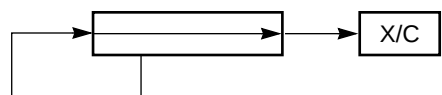
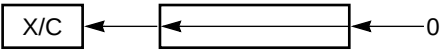
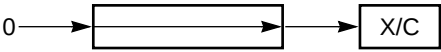
Instruction	Syntax	Operand Size	Operation
ABCD	Dn, Dn – (An), – (An)	8 8	Source ₁₀ + Destination ₁₀ + X ⇒ Destination
ADD	Dn, <ea> <ea>, Dn	8, 16, 32 8, 16, 32	Source + Destination ⇒ Destination
ADDA	<ea>, An	16, 32	Source + Destination ⇒ Destination
ADDI	#<data>, <ea>	8, 16, 32	Immediate data + Destination ⇒ Destination
ADDQ	# <data>, <ea>	8, 16, 32	Immediate data + Destination ⇒ Destination
ADDX	Dn, Dn – (An), – (An)	8, 16, 32 8, 16, 32	Source + Destination + X ⇒ Destination
AND	<ea>, Dn Dn, <ea>	8, 16, 32 8, 16, 32	Source • Destination ⇒ Destination
ANDI	# <data>, <ea>	8, 16, 32	Data • Destination ⇒ Destination
ANDI to CCR	# <data>, CCR	8	Source • CCR ⇒ CCR
ANDI to SR1 ¹	# <data>, SR	16	Source • SR ⇒ SR
ASL	Dn, Dn # <data>, Dn <ea>	8, 16, 32 8, 16, 32 16	
ASR	Dn, Dn # <data>, Dn <ea>	8, 16, 32 8, 16, 32 16	
Bcc	label	8, 16, 32	If condition true, then PC + d ⇒ PC
BCHG	Dn, <ea> # <data>, <ea>	8, 32 8, 32	$\overline{(\langle \text{bit number} \rangle \text{ of destination})} \Rightarrow Z \Rightarrow \text{bit of destination}$
BCLR	Dn, <ea> # <data>, <ea>	8, 32 8, 32	$\overline{(\langle \text{bit number} \rangle \text{ of destination})}$ 0 ⇒ bit of destination
BGND	none	none	If background mode enabled, then enter background mode, else format/vector ⇒ – (SSP); PC ⇒ – (SSP); SR ⇒ – (SSP); (vector) ⇒ PC
BKPT	# <data>	none	If breakpoint cycle acknowledged, then execute returned operation word, else trap as illegal instruction
BRA	label	8, 16, 32	PC + d ⇒ PC
BSET	Dn, <ea> # <data>, <ea>	8, 32 8, 32	$\overline{(\langle \text{bit number} \rangle \text{ of destination})} \Rightarrow Z$; 1 ⇒ bit of destination
BSR	label	8, 16, 32	SP – 4 ⇒ SP; PC ⇒ (SP); PC + d ⇒ PC
BTST	Dn, <ea> # <data>, <ea>	8, 32 8, 32	$\overline{(\langle \text{bit number} \rangle \text{ of destination})} \Rightarrow Z$
CHK	<ea>, Dn	16, 32	If Dn < 0 or Dn > (ea), then CHK exception
CHK2	<ea>, Rn	8, 16, 32	If Rn < lower bound or Rn > upper bound, then CHK exception
CLR	<ea>	8, 16, 32	0 ⇒ Destination
CMP	<ea>, Dn	8, 16, 32	(Destination – Source), CCR shows results
CMPA	<ea>, An	16, 32	(Destination – Source), CCR shows results
CMPI	# <data>, <ea>	8, 16, 32	(Destination – Data), CCR shows results

Table 20 Instruction Set Summary (Continued)

Instruction	Syntax	Operand Size	Operation
CMPM	(An) +, (An) +	8, 16, 32	(Destination – Source), CCR shows results
CMP2	<ea>, Rn	8, 16, 32	Lower bound ≤ Rn ≤ Upper bound, CCR shows result
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	<ea>	none	Destination ⇒ PC
JSR	<ea>	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 <ea>	8, 16, 32 8, 16, 32 16	
LSR	Dn, Dn # <data>, Dn <ea>	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

4.7 Background Debugging Mode

The background debugger on the CPU32 is implemented in CPU microcode. The background debugging commands are summarized below.

Table 21 Background Debugging Mode

Command	Mnemonic	Description
Read D/A Register	RDREG/RAREG	Read the selected address or data register and return the results through the serial interface.
Write D/A Register	WDREG/WAREG	The data operand is written to the specified address or data register.
Read System Register	RSREG	The specified system control register is read. All registers that can be read in supervisor mode can be read in background mode.
Write System Register	WSREG	The operand data is written into the specified system control register.
Read Memory Location	READ	Read the sized data at the memory location specified by the long-word address. The source function code register (SFC) determines the address space accessed.
Write Memory Location	WRITE	Write the operand data to the memory location specified by the long-word address. The destination function code (DFC) register determines the address space accessed.
Dump Memory Block	DUMP	Used in conjunction with the READ command to dump large blocks of memory. An initial READ is executed to set up the starting address of the block and retrieve the first result. Subsequent operands are retrieved with the DUMP command.
Fill Memory Block	FILL	Used in conjunction with the WRITE command to fill large blocks of memory. Initially, a WRITE is executed to set up the starting address of the block and supply the first operand. The FILL command writes subsequent operands.
Resume Execution	GO	The pipe is flushed and refilled before resuming instruction execution at the current PC.
Patch User Code	CALL	Current program counter is stacked at the location of the current stack pointer. Instruction execution begins at user patch code.
Reset Peripherals	RST	Asserts RESET for 512 clock cycles. The CPU is not reset by this command. Synonymous with the CPU RESET instruction.
No Operation	NOP	NOP performs no operation and can be used as a null command.

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:

$$\text{SCK Baud Rate} = \text{System Clock}/(2\text{SPBR})$$

or

$$\text{SPBR} = \text{System Clock}/(2\text{SCK})(\text{Baud Rate Desired})$$

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

15	14	8	7	0
SPE	DCKL	DTL		

RESET:

0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0

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.

DCKL — Delay before SCK

When the DCKL 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:

$$\text{PCS to SCK Delay} = [\text{DCKL}/\text{System Clock}]$$

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

When the DCKL value of a queue entry equals zero, then DCKL 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:

$$\text{Delay after Transfer} = [(32\text{DTL})/\text{System Clock}]$$

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.

$$\text{Standard Delay after Transfer} = [17/\text{System Clock}]$$

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.

5.5.1 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

5.5.2 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		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}

Writing a value of zero to SCBR disables the baud rate generator.

The following table lists the SCBR settings for standard and maximum baud rates using 16.78-MHz and 20.97-MHz system clocks.

RAF — Receiver Active Flag

0 = SCI receiver is idle

1 = SCI receiver is busy

RAF indicates whether the SCI receiver is busy. It is set when the receiver detects a possible start bit and is cleared when the chosen type of idle line is detected. RAF can be used to reduce collisions in systems with multiple masters.

IDLE — Idle-Line Detected Flag

0 = SCI receiver did not detect an idle-line condition.

1 = SCI receiver detected an idle-line condition.

IDLE is disabled when RWU in SCCR1 is set. IDLE is set when the SCI receiver detects the idle-line condition specified by ILT in SCCR1. If cleared, IDLE will not set again until after RDRF is set. RDRF is set when a break is received, so that a subsequent idle line can be detected.

OR — Overrun Error Flag

0 = RDRF is cleared before new data arrives.

1 = RDRF is not cleared before new data arrives.

OR is set when a new byte is ready to be transferred from the receive serial shifter to the RDR, and RDRF is still set. Data transfer is inhibited until OR is cleared. Previous data in RDR remains valid, but data received during overrun condition (including the byte that set OR) is lost.

NF — Noise Error Flag

0 = No noise detected on the received data

1 = Noise occurred on the received data

NF is set when the SCI receiver detects noise on a valid start bit, on any data bit, or on a stop bit. It is not set by noise on the idle line or on invalid start bits. Each bit is sampled three times. If none of the three samples are the same logic level, the majority value is used for the received data value, and NF is set. NF is not set until an entire frame is received and RDRF is set.

FE — Framing Error Flag

0 = No framing error on the received data.

1 = Framing error or break occurred on the received data.

FE is set when the SCI receiver detects a zero where a stop bit was to have occurred. FE is not set until the entire frame is received and RDRF is set. A break can also cause FE to be set. It is possible to miss a framing error if RXD happens to be at logic level one at the time the stop bit is expected.

PF — Parity Error Flag

0 = No parity error on the received data

1 = Parity error occurred on the received data

PF is set when the SCI receiver detects a parity error. PF is not set until the entire frame is received and RDRF is set.

SCDR — SCI Data Register

\$YFFC0E

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	R8/T8	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0

RESET:

0 0 0 0 0 0 0 U U U U U U U U U

SCDR contains two data registers at the same address. Receive data register (RDR) is a read-only register that contains data received by the SCI. The data comes into the receive serial shifter and is transferred to RDR. Transmit data register (TDR) is a write-only register that contains data to be transmitted. The data is first written to TDR, then transferred to the transmit serial shifter, where additional format bits are added before transmission. R[7:0]/T[7:0] contain either the first eight data bits received when SCDR is read, or the first eight data bits to be transmitted when SCDR is written. R8/T8 are used when the SCI is configured for 9-bit operation. When it is configured for 8-bit operation, they have no meaning or effect.

Table 26 GPT Address Map

Access	Address	15	8	7	0
S	\$YFF900	GPT MODULE CONFIGURATION (GPTMCR)			
S	\$YFF902	(RESERVED FOR TEST)			
S	\$YFF904	INTERRUPT CONFIGURATION (ICR)			
U	\$YFF906	PGP DATA DIRECTION (DDRGP)		PGP DATA (PORTGP)	
U	\$YFF908	OC1 ACTION MASK (OC1M)		OC1 ACTION DATA (OC1D)	
U	\$YFF90A	TIMER COUNTER (TCNT)			
U	\$YFF90C	PA CONTROL (PACTL)		PA COUNTER (PACNT)	
U	\$YFF90E	INPUT CAPTURE 1 (TIC1)			
U	\$YFF910	INPUT CAPTURE 2 (TIC2)			
U	\$YFF912	INPUT CAPTURE 3 (TIC3)			
U	\$YFF914	OUTPUT COMPARE 1 (TOC1)			
U	\$YFF916	OUTPUT COMPARE 2 (TOC2)			
U	\$YFF918	OUTPUT COMPARE 3 (TOC3)			
U	\$YFF91A	OUTPUT COMPARE 4 (TOC4)			
U	\$YFF91C	INPUT CAPTURE 4/OUTPUT COMPARE 5 (TI4/O5)			
U	\$YFF91E	TIMER CONTROL 1 (TCTL1)		TIMER CONTROL 2 (TCTL2)	
U	\$YFF920	TIMER MASK 1 (TMSK1)		TIMER MASK 2 (TMSK2)	
U	\$YFF922	TIMER FLAG 1 (TFLG1)		TIMER FLAG 2 (TFLG2)	
U	\$YFF924	FORCE COMPARE (CFORC)		PWM CONTROL C (PWMC)	
U	\$YFF926	PWM CONTROL A (PWMA)		PWM CONTROL B (PWMB)	
U	\$YFF928	PWM COUNT (PWMCNT)			
U	\$YFF92A	PWMA BUFFER (PWMBUFA)		PWMB BUFFER (PWMBUFB)	
U	\$YFF92C	GPT PRESCALER (PRESCL)			
	\$YFF92E– \$YFF93F	NOT USED			

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

6.2 Capture/Compare Unit

The capture/compare unit features three input capture channels, four output compare channels, and one input capture/output compare channel (function selected by control register). These channels share a 16-bit free-running counter (TCNT), which derives its clock from seven stages of a 9-stage prescaler or from external clock input PCLK. This section, which is similar to the timer found on the MC68HC11F1, also contains one pulse accumulator channel. The pulse accumulator logic includes its own 8-bit counter and can operate in either event counting mode or gated time accumulation mode. Refer to the following block diagrams of the GPT timer and prescaler.

CPR[2:0] — Timer Prescaler/PCLK Select Field

This field selects one of seven prescaler taps or PCLK to be TCNT input.

CPR[2:0]	System Clock Divide-By Factor
000	4
001	8
010	16
011	32
100	64
101	128
110	256
111	PCLK

TFLG1/TFLG2 — Timer Interrupt Flag Registers 1–2

\$YFF922

15	14	11	10	8	7	6	5	4	3	2	1	0
I4/O5F	OCF	ICF	TOF	0	PAOVF	PAIF	0	0	0	0	0	0

RESET:

0 0 0 0 0 0 0 0 0 0 0 0 0

These registers show condition flags that correspond to various GPT events. If the corresponding interrupt enable bit in TMSK1/TMSK2 is set, an interrupt occurs.

I4/O5F — Input Capture 4/Output Compare 5 Flag

When I4/O5 in PACTL is zero, this flag is set each time TCNT matches the value in TOC5. When I4/O5 in PACTL is one, the flag is set each time a selected edge is detected at the I4/O5 pin.

OCF[4:1] — Output Compare Flags

An output compare flag is set each time TCNT matches the corresponding TOC register. OCF[4:1] correspond to OC[4:1].

ICF[3:1] — Input Capture Flags

A flag is set each time a selected edge is detected at the corresponding input capture pin. ICF[3:1] correspond to IC[3:1].

TOF — Timer Overflow Flag

This flag is set each time TCNT advances from a value of \$FFFF to \$0000.

PAOVF — Pulse Accumulator Overflow Flag

This flag is set each time the pulse accumulator counter advances from a value of \$FF to \$00.

PAIF — Pulse Accumulator Flag

In event counting mode, this flag is set when an active edge is detected on the PAI pin. In gated time accumulation mode, PAIF is set at the end of the timed period.

CFORC/PWMC — Compare Force Register/PWM Control Register C

\$YFF924

15	11	10	9	8	7	6	4	3	2	1	0
FOC	0	FPWMA	FPWMB	PPROUT	PPR	SFA	SFB	F1A	F1B		

RESET:

0 0 0 0 0 0 0 0 0 0 0 0

Setting a bit in CFORC causes a specific output on OC or PWM pins. PWMC sets PWM operating conditions.

FOC[5:1] — Force Output Compare

0 = Has no meaning

1 = Causes pin action programmed for corresponding OC pin, but the OC flag is not set.

FOC[5:1] correspond to OC[5:1].

FPWMA — Force PWMA Value

0 = Normal PWMA operation

1 = The value of F1A is driven out on the PWMA pin, regardless of the state of PPROUT.

FPWMB — Force PWMB Value

0 = Normal PWMB operation

1 = The value of F1B is driven out on the PWMB pin.

PPROUT — PWM Clock Output Enable

0 = Normal PWM operation on PWMA

1 = TCNT clock driven out PWMA pin

PPR[2:0] — PWM Prescaler/PCLK Select

This field selects one of seven prescaler taps, or PCLK, to be PWMCNT input.

PPR[2:0]	System Clock Divide-By Factor
000	2
001	4
010	8
011	16
100	32
101	64
110	128
111	PCLK

SFA — PWMA Slow/Fast Select

0 = PWMA period is 256 PWMCNT increments long.

1 = PWMA period is 32768 PWMCNT increments long.

SFB — PWMB Slow/Fast Select

0 = PWMB period is 256 PWMCNT increments long.

1 = PWMB period is 32768 PWMCNT increments long.

The following table shows the effects of SF settings on PWM frequency for a 16.78-MHz system clock and a 20.97-MHz system clock.

PPR[2:0]	Prescaler Tap		SFA/B = 0		SFA/B = 1	
	16.78 MHz	20.97 MHz	16.78 MHz	20.97 MHz	16.78 MHz	20.97 MHz
000	Div 2 = 8.39 MHz	Div 2 = 10.5 MHz	32.8 kHz	41 kHz	256 Hz	320 Hz
001	Div 4 = 4.19 MHz	Div 4 = 5.25 MHz	16.4 kHz	20.5 kHz	128 Hz	160 Hz
010	Div 8 = 2.10 MHz	Div 8 = 2.62 MHz	8.19 kHz	10.2 kHz	64.0 Hz	80.0 Hz
011	Div 16 = 1.05 MHz	Div 16 = 1.31 MHz	4.09 kHz	5.15 kHz	32.0 Hz	40.0 Hz
100	Div 32 = 524 kHz	Div 32 = 655 kHz	2.05 kHz	2.56 kHz	16.0 Hz	20.0 Hz
101	Div 64 = 262 kHz	Div 64 = 328 kHz	1.02 kHz	1.28 kHz	8.0 Hz	10.0 Hz
110	Div 128 = 131 kHz	Div 128 = 164 kHz	512 Hz	641 Hz	4.0 Hz	5.0 Hz
111	PCLK	PCLK	PCLK/256	PCLK/256	PCLK/32768	PCLK/32768

F1A — Force Logic Level One on PWMA

0 = Force logic level zero output on PWMA pin

1 = Force logic level one output on PWMA pin

Home Page:

www.freescal.com

email:

support@freescal.com

USA/Europe or Locations Not Listed:

Freescal Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescal.com

Europe, Middle East, and Africa:

Freescal Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescal.com

Japan:

Freescal Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescal.com

Asia/Pacific:

Freescal Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescal.com

For Literature Requests Only:

Freescal Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescalSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb- free versions of Freescal products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb- free counterparts. For further information, see <http://www.freescal.com> or contact your Freescal sales representative.

For information on Freescal.s Environmental Products program, go to <http://www.freescal.com/epp>.

Information in this document is provided solely to enable system and software implementers to use Freescal Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document. Freescal Semiconductor reserves the right to make changes without further notice to any products herein. Freescal Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescal Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescal Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescal Semiconductor does not convey any license under its patent rights nor the rights of others. Freescal Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescal Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescal Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescal Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescal Semiconductor was negligent regarding the design or manufacture of the part.