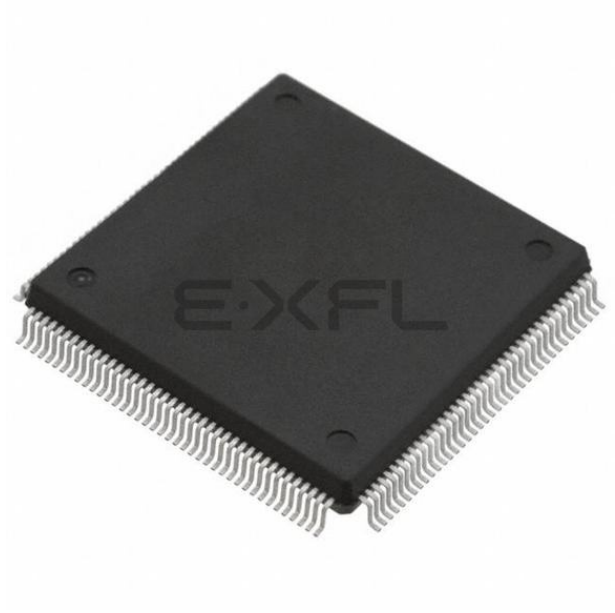


Welcome to E-XFL.COM

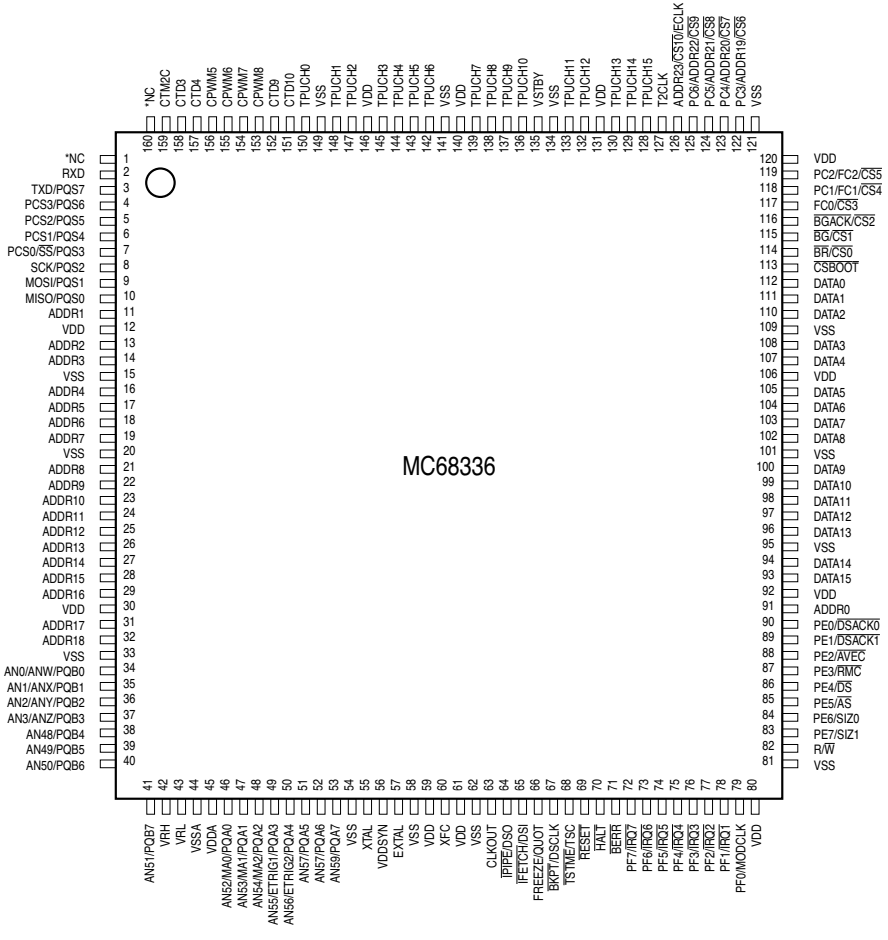
What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	CPU32
Core Size	32-Bit Single-Core
Speed	20MHz
Connectivity	CANbus, EBI/EMI, SCI, SPI
Peripherals	POR, PWM, WDT
Number of I/O	18
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	7.5K x 8
Voltage - Supply (Vcc/Vdd)	4.75V ~ 5.25V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	160-BQFP
Supplier Device Package	160-QFP (28x28)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68336amab20

TABLE OF CONTENTS
(Continued)



*NOTE: MC68336 REVISION D AND LATER (F60K AND LATER MASK SETS) HAVE ASSIGNED PINS 1 AND 160 AS "NO CONNECT", TO ALLOW PIN COMPATIBILITY WITH THE MC68376. FOR REVISION C (D65J MASK SET) DEVICES, PIN 1 IS V_{SS} AND PIN 160 IS V_{DD}.

336 160-PIN QFP

Figure 3-2 MC68336 Pin Assignments for 160-Pin Package

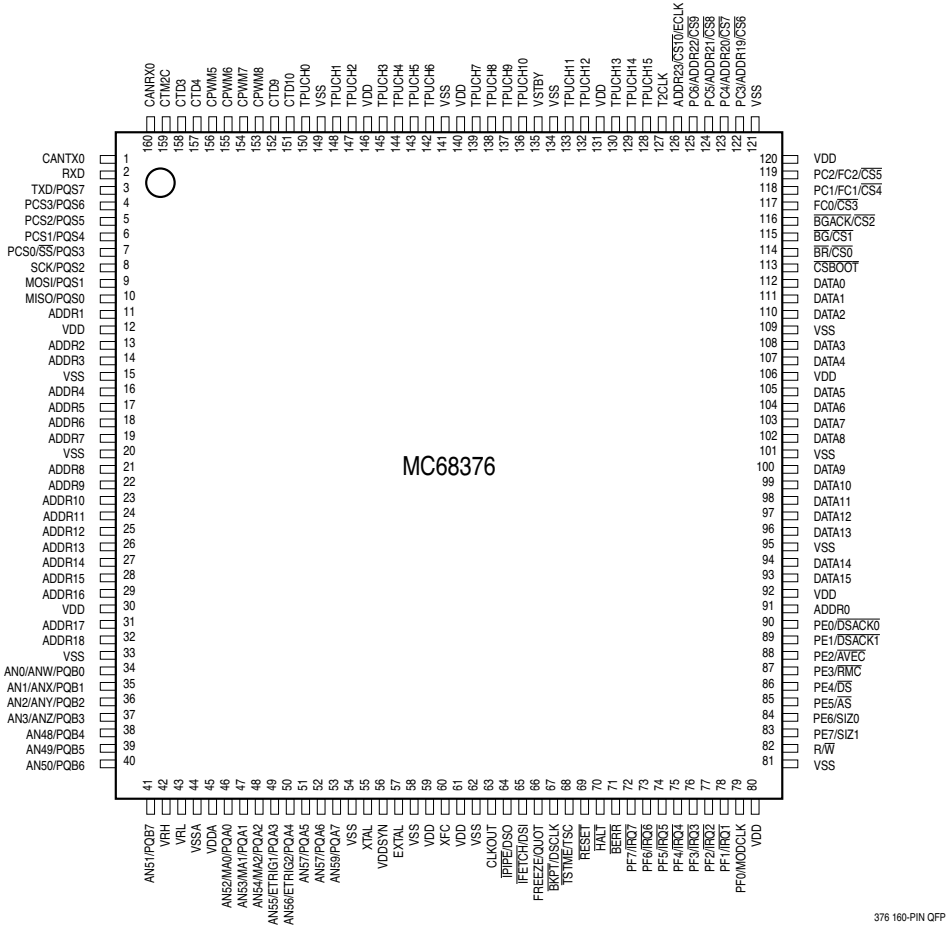


Figure 3-3 MC68376 Pin Assignments for 160-Pin Package

3.4 Pin Descriptions

The following tables summarize the functional characteristics of MC68336/376 pins. **Table 3-1** shows all inputs and outputs. Digital inputs and outputs use CMOS logic levels. An entry in the “Discrete I/O” column indicates that a pin can also be used for general-purpose input, output, or both. The I/O port designation is given when it applies. Refer to **Figure 3-1** for port organization. **Table 3-2** shows types of output drivers. **Table 3-3** shows the characteristics of power pins.

3.5 Signal Descriptions

The following tables define the MC68336/376 signals. **Table 3-4** shows signal origin, type, and active state. **Table 3-5** describes signal functions. Both tables are sorted alphabetically by mnemonic. MCU pins often have multiple functions. More than one description can apply to a pin.

Table 3-4 MC68336/376 Signal Characteristics

Signal Name	MCU Module	Signal Type	Active State
ADDR[23:0]	SIM	Bus	—
AN[59:48]/[3:0]	QADC	Input	—
AN[w, x, y, z]	QADC	Input	—
$\overline{\text{AS}}$	SIM	Output	0
$\overline{\text{AVEC}}$	SIM	Input	0
$\overline{\text{BERR}}$	SIM	Input	0
$\overline{\text{BG}}$	SIM	Output	0
$\overline{\text{BGACK}}$	SIM	Input	0
$\overline{\text{BKPT}}$	CPU32	Input	0
$\overline{\text{BR}}$	SIM	Input	0
CLKOUT	SIM	Output	—
CANRX0 (MC68376 Only)	TouCAN	Input	—
CANTX0 (MC68376 Only)	TouCAN	Output	—
$\overline{\text{CS}}$[10:0]	SIM	Output	0
$\overline{\text{CSBOOT}}$	SIM	Output	0
CPWM[8:5]	CTM4	Output	—
CTD[10:9]/[4:3]	CTM4	Input/Output	—
CTM2C	CTM4	Input	—
DATA[15:0]	SIM	Bus	—
DS	SIM	Output	0
$\overline{\text{DSACK}}$[1:0]	SIM	Input	0
DSCLK	CPU32	Input	Serial Clock
DSI	CPU32	Input	Serial Data
DSO	CPU32	Output	Serial Data
ECLK	SIM	Output	—
ETRIG[2:1]	QADC	Input	—
EXTAL	SIM	Input	—
FC[2:0]	SIM	Output	—
FREEZE	SIM	Output	1
$\overline{\text{HALT}}$	SIM	Input/Output	0
$\overline{\text{IFETCH}}$	CPU32	Output	0
$\overline{\text{IPIPE}}$	CPU32	Output	0
$\overline{\text{IRQ}}$[7:1]	SIM	Input	0
MA[2:0]	QADC	Output	1
MISO	QSM	Input/Output	—
MODCLK	SIM	Input	—
MOSI	QSM	Input/Output	—
PC[6:0]	SIM	Output	—
PCS[3:0]	QSM	Input/Output	—
PE[7:0]	SIM	Input/Output	—
PF[7:0]	SIM	Input/Output	—

Table 4-6 Background Mode Command Summary

Command	Mnemonic	Description
Read D/A Register	RDREG/RAREG	Read the selected address or data register and return the results via 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. An initial WRITE is executed to set up the starting address of the block and supply the first operand. Subsequent operands are written with the FILL command.
Resume Execution	GO	The pipe is flushed and re-filled 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 $\overline{\text{RESET}}$ 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 may be used as a null command.

4.10.7 Background Mode Registers

BDM processing uses three special purpose registers to keep track of program context during development. A description of each follows.

4.10.7.1 Fault Address Register (FAR)

The FAR contains the address of the faulting bus cycle immediately following a bus or address error. This address remains available until overwritten by a subsequent bus cycle. Following a double bus fault, the FAR contains the address of the last bus cycle. The address of the first fault (if there was one) is not visible to the user.

4.10.7.2 Return Program Counter (RPC)

The RPC points to the location where fetching will commence after transition from background mode to normal mode. This register should be accessed to change the flow of a program under development. Changing the RPC to an odd value will cause an address error when normal mode prefetching begins.

When an external device asserts $\overline{\text{RESET}}$ for the proper period, reset control logic clocks the signal into an internal latch. The control logic drives the $\overline{\text{RESET}}$ pin low for an additional 512 CLKOUT cycles after it detects that the $\overline{\text{RESET}}$ signal is no longer being externally driven to guarantee this length of reset to the entire system.

If an internal source asserts a reset signal, the reset control logic asserts the $\overline{\text{RESET}}$ pin for a minimum of 512 cycles. If the reset signal is still asserted at the end of 512 cycles, the control logic continues to assert the $\overline{\text{RESET}}$ pin until the internal reset signal is negated.

After 512 cycles have elapsed, the $\overline{\text{RESET}}$ pin goes to an inactive, high-impedance state for ten cycles. At the end of this 10-cycle period, the $\overline{\text{RESET}}$ input is tested. When the input is at logic level one, reset exception processing begins. If, however, the $\overline{\text{RESET}}$ input is at logic level zero, reset control logic drives the pin low for another 512 cycles. At the end of this period, the pin again goes to high-impedance state for ten cycles, then it is tested again. The process repeats until $\overline{\text{RESET}}$ is released.

5.7.7 Power-On Reset

When the SIM clock synthesizer is used to generate system clocks, power-on reset involves special circumstances related to application of system and clock synthesizer power. Regardless of clock source, voltage must be applied to the clock synthesizer power input pin V_{DDSYN} for the MCU to operate. The following discussion assumes that V_{DDSYN} is applied before and during reset, which minimizes crystal start-up time. When V_{DDSYN} is applied at power-on, start-up time is affected by specific crystal parameters and by oscillator circuit design. V_{DD} ramp-up time also affects pin state during reset. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for voltage and timing specifications.

During power-on reset, an internal circuit in the SIM drives the IMB internal (MSTRST) and external (EXTRST) reset lines. The power-on reset circuit releases the internal reset line as V_{DD} ramps up to the minimum operating voltage, and SIM pins are initialized to the values shown in **Table 5-17**. When V_{DD} reaches the minimum operating voltage, the clock synthesizer VCO begins operation. Clock frequency ramps up to specified limp mode frequency (f_{limp}). The external $\overline{\text{RESET}}$ line remains asserted until the clock synthesizer PLL locks and 512 CLKOUT cycles elapse.

The SIM clock synthesizer provides clock signals to the other MCU modules. After the clock is running and MSTRST is asserted for at least four clock cycles, these modules reset. V_{DD} ramp time and VCO frequency ramp time determine how long the 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 external pull-up resistors, external logic on input/output or output-only pins during this time must condition the lines. Active drivers require high-impedance buffers or isolation resistors to prevent conflict.

At the release of reset, the exception vector table is located beginning at address $000000. This value can be changed by programming the vector base register (VBR) with a new value. Multiple vector tables can be used. Refer to **4.9 Exception Processing** for more information.

5.8.2 Interrupt Priority and Recognition

The CPU32 provides seven levels of interrupt priority (1-7), seven automatic interrupt vectors, and 200 assignable interrupt vectors. All interrupts with priorities less than seven can be masked by the interrupt priority (IP) field in status register.

NOTE

Exceptions such as “address error” are not interrupts and have no “level” associated. Exceptions cannot ever be masked.

There are seven interrupt request signals ($\overline{\text{IRQ}}$[7:1]). These signals are used internally on the IMB, and have corresponding pins for external interrupt service requests. The CPU32 treats all interrupt requests as though they come from internal modules; external interrupt requests are treated as interrupt service requests from the SIM. Each of the interrupt request signals corresponds to an interrupt priority. $\overline{\text{IRQ1}}$ has the lowest priority and $\overline{\text{IRQ7}}$ the highest.

Interrupt recognition is determined by interrupt priority level and interrupt priority (IP) mask value. The interrupt priority mask consists of three bits in the CPU32 status register. 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 from being recognized and processed. $\overline{\text{IRQ7}}$, however, is always recognized, even if the mask value is %111.

$\overline{\text{IRQ}}$[7:1] are active-low level-sensitive inputs. The low on the pin must remain asserted until an interrupt acknowledge cycle corresponding to that level is detected.

$\overline{\text{IRQ7}}$ is transition-sensitive as well as level-sensitive: a level-7 interrupt is not detected unless a falling edge transition is detected on the $\overline{\text{IRQ7}}$ line. This prevents redundant servicing and stack overflow. A non-maskable interrupt is generated each time $\overline{\text{IRQ7}}$ is asserted as well as each time the priority mask is written while $\overline{\text{IRQ7}}$ is asserted. If $\overline{\text{IRQ7}}$ is asserted and the IP mask is written to any new value (including %111), $\overline{\text{IRQ7}}$ will be recognized as a new $\overline{\text{IRQ7}}$.

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 interrupts is complete.

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. Writing a value to DTL[7:0] in SPCR1 specifies a delay period. The DT bit in each command RAM byte determines whether the standard delay period (DT = 0) or the specified delay period (DT = 1) is used. The following expression is used to calculate the delay:

$$\text{Delay after Transfer} = \frac{32 \times \text{DTL[7:0]}}{\text{System Clock}}$$

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

A zero value for DTL[7:0] causes a delay-after-transfer value of 8192/System Clock.

$$\text{Standard Delay after Transfer} = \frac{17}{\text{System Clock}}$$

Adequate delay between transfers must be specified for long data streams because the QSPI requires time to load a transmit RAM entry for transfer. Receiving devices need at least the standard delay between successive transfers. If the system clock is operating at a slower rate, the delay between transfers must be increased proportionately.

Operation is initiated by setting the SPE bit in SPCR1. Shortly after SPE is set, the QSPI executes the command at the command RAM address pointed to by NEWQP. Data at the pointer address in transmit RAM is loaded into the data serializer and transmitted. Data that is simultaneously received is stored at the pointer address in receive RAM.

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

When the CONT bit in a command RAM byte is set, PCS pins are continuously driven in specified states during and between transfers. If the chip-select pattern changes during or between transfers, the original pattern is driven until execution of the following transfer begins. When CONT is cleared, the data in register PORTQS is driven between transfers. The data in PORTQS must match the inactive states of SCK and any peripheral chip-selects used.

When the QSPI reaches the end of the queue, it sets the SPIF flag. If the SPIFIE bit in SPCR2 is set, an interrupt request is generated when SPIF is asserted. At this point, the QSPI clears SPE and stops unless wrap-around mode is enabled.

8.13 Interrupts

The QADC supports both polled and interrupt driven operation. Status bits in QASR reflect the operating condition of each queue and can optionally generate interrupts when enabled by the appropriate bits in QACR1 and/or QACR2.

8.13.1 Interrupt Sources

The QADC has four interrupt service sources, each of which is separately enabled. Each time the result is written for the last CCW in a queue, the completion flag for the corresponding queue is set, and when enabled, an interrupt request is generated. In the same way, each time the result is written for a CCW with the pause bit set, the queue pause flag is set, and when enabled, an interrupt request is generated.

Table 8-5 displays the status flag and interrupt enable bits which correspond to queue 1 and queue 2 activity.

Table 8-5 QADC Status Flags and Interrupt Sources

Queue	Queue Activity	Status Flag	Interrupt Enable Bit
Queue 1	Result written for the last CCW in queue 1	CF1	CIE1
	Result written for a CCW with pause bit set in queue 1	PF1	PIE1
Queue 2	Result written for the last CCW in queue 2	CF2	CIE2
	Result written for a CCW with pause bit set in queue 2	PF2	PIE2

Both polled and interrupt-driven QADC operations require that status flags must be cleared after an event occurs. Flags are cleared by first reading QASR with the appropriate flag bits set to one, then writing zeros to the flags that are to be cleared. A flag can be cleared only if the flag was a logic one at the time the register was read by the CPU. If a new event occurs between the time that the register is read and the time that it is written, the associated flag is not cleared.

8.13.2 Interrupt Register

The QADC interrupt register QADCINT specifies the priority level of QADC interrupt requests and the upper six bits of the vector number provided during an interrupt acknowledge cycle.

The values contained in the IRLQ1 and IRLQ2 fields in QADCINT determine the priority of QADC interrupt service requests. A value of %000 in either field disables the interrupts associated with that field. The interrupt levels for queue 1 and queue 2 may be different.

The IVB[7:2] bits specify the upper six bits of each QADC interrupt vector number. IVB[1:0] have fixed assignments for each of the four QADC interrupt sources. Refer to **8.13.3 Interrupt Vectors** for more information.

In the CTM4, TBB2 is global and accessible to every submodule. TBB1 and TBB4 are split to form two local time base buses. **Table 10-1** shows which time base buses are available to each CTM4 submodule.

Table 10-1 CTM4 Time Base Bus Allocation

Submodule	Global/Local Time Base Bus Allocation		Submodule	Global/Local Time Base Bus Allocation	
	Global Bus A	Global Bus B		Global Bus A	Global Bus B
DASM9	TBB1	TBB2	MCSM 2	TBB4	TBB2
DASM10	TBB1	TBB2	DASM 3	TBB4	TBB2
MCSM 11	TBB1	TBB2	DASM 4	TBB4	TBB2
FCSM 12	TBB1	TBB2			

Each PWMSM has an independent 16-bit counter and 8-bit prescaler clocked by the PCLK1 signal, which is generated by the CPSM. The PWMSMs are not connected to any of the time base buses. Refer to **10.9 Pulse-Width Modulation Submodule (PWMSM)** for more information.

10.4 Bus Interface Unit Submodule (BIUSM)

The BIUSM connects the SMB to the IMB and allows the CTM4 submodules to communicate with the CPU32. The BIUSM also communicates CTM4 submodule interrupt requests to the IMB, and transfers the interrupt level, arbitration number and vector number to the CPU32 during the interrupt acknowledge cycle.

10.4.1 STOP Effect On the BIUSM

When the CPU32 STOP instruction is executed, only the CPU32 is stopped; the CTM4 continues to operate as normal.

10.4.2 Freeze Effect On the BIUSM

CTM4 response to assertion of the IMB FREEZE signal is controlled by the FRZ bit in the BIUSM configuration register (BIUMCR). Since the BIUSM propagates FREEZE to the CTM4 submodules via the SMB, the setting of FRZ affects all CTM4 submodules.

If the IMB FREEZE signal is asserted and FRZ = 1, all CTM4 submodules freeze. The following conditions apply when the CTM4 is frozen:

- All submodule registers can still be accessed.
- The CPSM, FCSM, MCSM, and PWMSM counters stop counting.
- The IN status bit still reflects the state of the FCSM external clock input pin.
- The IN2 status bit still reflects the state of the MCSM external clock input pin, and the IN1 status bit still reflects the state of the MCSM modulus load input pin.
- DASM capture and compare functions are disabled.
- The DASM IN status bit still reflects the state of its associated pin in the DIS, IPWM, IPM, and IC modes. In the OCB, OCAB, and OPWM modes, IN reflects the state of the DASM output flip flop.
- When configured for OCB, OCAB, or OPWM modes, the state of the DASM

10.5.1 CPSM Registers

The CPSM contains a control register (CPCR) and a test register (CPTR). All unused bits and reserved address locations return zero when read. Writes to unused bits and reserved address locations have no effect. Refer to **D.7.4 CPSM Control Register** and **D.7.5 CPSM Test Register** for information concerning CPSM register and bit descriptions.

10.6 Free-Running Counter Submodule (FCSM)

The free-running counter submodule (FCSM) has a 16-bit up counter with an associated clock source selector, selectable time-base bus drivers, control registers, status bits, and interrupt logic. When the 16-bit up counter overflows from $FFFF to $0000, an optional overflow interrupt request can be generated. The current state of the 16-bit counter is the primary output of the counter submodules. The user can select which, if any, time base bus is to be driven by the 16-bit counter. A software control register selects whether the clock input to the counter is one of the taps from the prescaler or an input pin. The polarity of the external input pin is also programmable.

In order to count, the FCSM requires the CPSM clock signals to be present. After reset, the FCSM does not count until the prescaler in the CPSM starts running (when the software sets the PRUN bit). This allows all counters in the CTM4 submodules to be synchronized.

The CTM4 has one FCSM. **Figure 10-3** shows a block diagram of the FCSM.

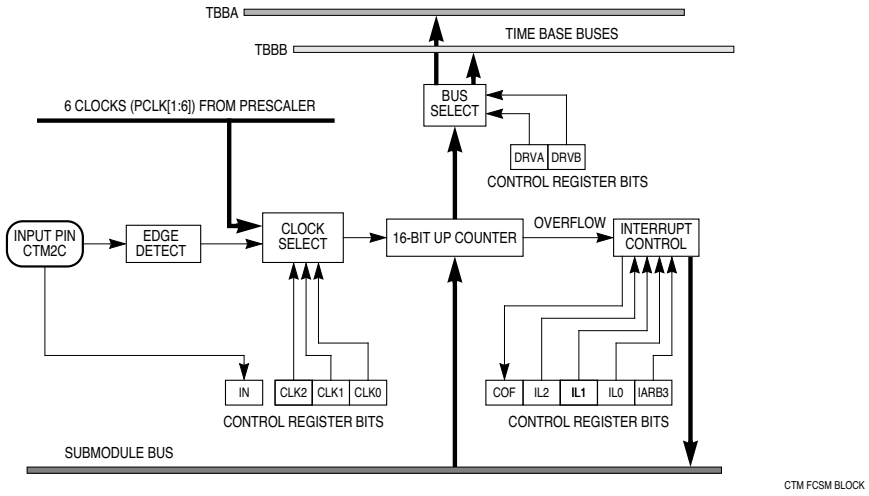


Figure 10-3 FCSM Block Diagram

12.7 Low-Power Stop Operation

Setting the STOP bit in TRAMMCR places the TPURAM in low-power stop mode. In low-power stop mode, the array retains its contents, but cannot be read or written by the CPU32. STOP can be written only when the processor is operating in supervisor mode. STOP is set during resets. Low-power stop mode is exited by clearing STOP.

The TPURAM module will switch to standby mode while it is in low-power mode, provided the operating constraints discussed above are met.

12.8 Reset

Reset places the TPURAM in low-power stop mode, enables supervisor mode access only, clears the base address register, and disables the array. These actions make it possible to write a new base address into the base address register.

When a synchronous reset occurs while a byte or word TPURAM access is in progress, the access is completed. If reset occurs during the first word access of a long-word operation, only the first word access is completed. If reset occurs during the second word access of a long-word operation, the entire access is completed. Data being read from or written to the TPURAM may be corrupted by asynchronous reset. Refer to **5.7 Reset** for more information concerning resets.

12.9 TPU Microcode Emulation

The TPURAM array can emulate the microcode ROM in the TPU module. This provides a means for developing custom TPU code. The TPU selects TPU emulation mode.

The TPU is connected to the TPURAM via a dedicated bus. While the TPURAM array is in TPU emulation mode, the access timing of the TPURAM module matches the timing of the TPU microcode ROM to ensure accurate emulation. Normal accesses through the IMB are inhibited and the control registers have no effect, allowing external RAM to emulate the TPURAM at the same addresses. Refer to **SECTION 11 TIME PROCESSOR UNIT** and to the *TPU Reference Manual* (TPURM/AD) for more information.

If a message transfer between the message buffer and a serial message buffer is in progress when the control/status word is read, the BUSY status will be indicated in the code field, and the lock will not be activated.

The user can release the lock on a message buffer in one of two ways. Reading the control/status word of another message buffer will lock that buffer, releasing the previously locked buffer. A global release can also be performed on any locked message buffer by reading the free-running timer.

Once a lock is released, any message transfers between an SMB and a message buffer which was delayed due to that buffer being locked will take place. For more details on the message buffer locking mechanism, and the effects on message buffer operation, refer to **13.5 TouCAN Operation**.

13.4.2 Receive Mask Registers

The receive mask registers are used as acceptance masks for received frame IDs. The following masks are defined:

- A global mask, used for receive buffers 0-13
- Two separate masks for buffers 14 and 15

The value of the mask registers should not be changed during normal operation. If the mask register data is changed after the masked identifier of a received message is matched to a locked message buffer, that message will be transferred into that message buffer once it is unlocked, regardless of whether that message's masked identifier still matches the receive buffer identifier. **Table 13-6** shows mask bit values.

Table 13-6 Receive Mask Register Bit Values

Mask Bit	Values
0	The corresponding incoming ID bit is “don’t care”.
1	The corresponding ID bit is checked against the incoming ID bit to see if a match exists.

Table 13-7 shows mask examples for normal and extended messages. Refer to **APPENDIX D REGISTER SUMMARY** for more information on RX mask registers.

Table A-6 AC Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
F1	Frequency of Operation[2]	f_{sys}	—	20.97	MHz
1	Clock Period	t_{cyc}	47.7	—	ns
1A	ECLK Period	t_{Ecyc}	381	—	ns
1B	External Clock Input Period[3]	t_{Xcyc}	47.7	—	ns
2, 3	Clock Pulse Width	t_{CW}	18.8	—	ns
2A, 3A	ECLK Pulse Width	t_{ECW}	183	—	ns
2B, 3B	External Clock Input High/Low Time[3]	t_{XCHL}	23.8	—	ns
3, 4	Clock Rise and Fall Time	t_{Crf}	—	5	ns
4A, 5A	Rise and Fall Time — All Outputs except CLKOUT	t_{rf}	—	8	ns
4B, 5B	External Clock Rise and Fall Time[4]	t_{XCrf}	—	5	ns
4	Clock High to Address, FC, SIZE, $\overline{RMC}$ Valid	t_{CHAV}	0	23	ns
5	Clock High to Address, Data, FC, SIZE, $\overline{RMC}$ High Impedance	t_{CHAZx}	0	47	ns
6	Clock High to Address, FC, SIZE, $\overline{RMC}$ Invalid[5]	t_{CHAZn}	0	—	ns
7	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Asserted	t_{CLSA}	0	23	ns
8A	$\overline{AS}$ to $\overline{DS}$ or $\overline{CS}$ Asserted (Read)[6]	t_{STSA}	–10	10	ns
8C	Clock Low to $\overline{IFETCH}$, $\overline{IPIPE}$ Asserted	t_{CLIA}	2	22	ns
11	Address, FC, SIZE, $\overline{RMC}$ Valid to $\overline{AS}$, $\overline{CS}$ Asserted	t_{AVSA}	10	—	ns
12	Clock Low to $\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated	t_{CLSN}	2	23	ns
12A	Clock Low to $\overline{IFETCH}$, $\overline{IPIPE}$ Negated	t_{CLIN}	2	22	ns
13	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to Address, FC, SIZE Invalid (Address Hold)	t_{SNAI}	10	—	ns
14	$\overline{AS}$, $\overline{CS}$ Width Asserted	t_{SWA}	80	—	ns
14A	$\overline{DS}$, $\overline{CS}$ Width Asserted (Write)	t_{SWAW}	36	—	ns
14B	$\overline{AS}$, $\overline{CS}$ Width Asserted (Fast Write Cycle)	t_{SWDW}	32	—	ns
15	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Width Negated[7]	t_{SN}	32	—	ns
16	Clock High to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ High Impedance	t_{CHSZ}	—	47	ns
17	$\overline{AS}$, $\overline{DS}$, $\overline{CS}$ Negated to R/$\overline{W}$ Negated	t_{SNRN}	10	—	ns
18	Clock High to R/$\overline{W}$ High	t_{CHRH}	0	23	ns
20	Clock High to R/$\overline{W}$ Low	t_{CHRL}	0	23	ns
21	R/$\overline{W}$ Asserted to $\overline{AS}$, $\overline{CS}$ Asserted	t_{RAAA}	10	—	ns
22	R/$\overline{W}$ Low to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{RASA}	54	—	ns
23	Clock High to Data Out Valid	t_{CHDO}	—	23	ns
24	Data Out Valid to Negating Edge of $\overline{AS}$, $\overline{CS}$	t_{DVASN}	10	—	ns
25	$\overline{DS}$, $\overline{CS}$ Negated to Data Out Invalid (Data Out Hold)	t_{SNDOI}	10	—	ns

Table A-6 AC Timing (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc ±5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
26	Data Out Valid to $\overline{DS}$, $\overline{CS}$ Asserted (Write)	t_{DVSA}	10	—	ns
27	Data In Valid to Clock Low (Data Setup)[5]	t_{DICL}	5	—	ns
27A	Late $\overline{BERR}$, $\overline{HALT}$ Asserted to Clock Low (Setup Time)	t_{BELCL}	15	—	ns
28	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACK[1:0]}$, $\overline{BERR}$, $\overline{HALT}$, $\overline{AVEC}$ Negated	t_{SNDN}	0	60	ns
29	$\overline{DS}$, $\overline{CS}$ Negated to Data In Invalid (Data In Hold)[8]	t_{SNDI}	0	—	ns
29A	$\overline{DS}$, $\overline{CS}$ Negated to Data In High Impedance[8, 9]	t_{SHDI}	—	48	ns
30	CLKOUT Low to Data In Invalid (Fast Cycle Hold)[8]	t_{CLDI}	10	—	ns
30A	CLKOUT Low to Data In High Impedance[8]	t_{CLDH}	—	72	ns
31	$\overline{DSACK[1:0]}$ Asserted to Data In Valid[10]	t_{DADI}	—	46	ns
33	Clock Low to $\overline{BG}$ Asserted/Negated	t_{CLBAN}	—	23	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted ($\overline{RMC}$ Not Asserted)[11]	t_{BRAGA}	1	—	t_{cyc}
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GAGN}	1	2	t_{cyc}
39	$\overline{BG}$ Width Negated	t_{GH}	2	—	t_{cyc}
39A	$\overline{BG}$ Width Asserted	t_{GA}	1	—	t_{cyc}
46	R/$\overline{W}$ Width Asserted (Write or Read)	t_{RWA}	115	—	ns
46A	R/$\overline{W}$ Width Asserted (Fast Write or Read Cycle)	t_{RWAS}	70	—	ns
47A	Asynchronous Input Setup Time $\overline{BR}$, $\overline{BGACK}$, $\overline{DSACK[1:0]}$, $\overline{BERR}$, $\overline{AVEC}$, $\overline{HALT}$	t_{AIST}	5	—	ns
47B	Asynchronous Input Hold Time	t_{AIHT}	12	—	ns
48	$\overline{DSACK[1:0]}$ Asserted to $\overline{BERR}$, $\overline{HALT}$ Asserted[12]	t_{DABA}	—	30	ns
53	Data Out Hold from Clock High	t_{DOCH}	0	—	ns
54	Clock High to Data Out High Impedance	t_{CHDH}	—	23	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RADC}	32	—	ns
56	$\overline{RESET}$ Pulse Width (Reset Instruction)	t_{HRPW}	512	—	t_{cyc}
57	$\overline{BERR}$ Negated to $\overline{HALT}$ Negated (Rerun)	t_{BNHN}	0	—	ns
70	Clock Low to Data Bus Driven (Show)	t_{SCLDD}	0	23	ns
71	Data Setup Time to Clock Low (Show)	t_{SCLDS}	10	—	ns
72	Data Hold from Clock Low (Show)	t_{SCLDH}	10	—	ns
73	$\overline{BKPT}$ Input Setup Time	t_{BKST}	10	—	ns
74	$\overline{BKPT}$ Input Hold Time	t_{BKHT}	10	—	ns
75	Mode Select Setup Time	t_{MSS}	20	—	t_{cyc}

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 background debug mode.

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.

SHEN[1:0] — Show Cycle Enable

The SHEN field determines how the external bus is driven during internal transfer operations. A show cycle allows internal transfers to be monitored externally.

Table D-4 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.

Table D-4 Show Cycle Enable Bits

SHEN[1:0]	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 in either supervisor or user mode.

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 ID

Each module that can generate interrupts, including the SIM, has an IARB field. Each IARB field can be assigned a value from $0 to $F. During an interrupt acknowledge cycle, IARB permits arbitration among simultaneous interrupts of the same priority level. The reset value of the SIM IARB field is $F. This prevents SIM interrupts from being discarded during system initialization.

D.2.2 System Integration Test Register

SIMTR — System Integration Test Register **$YFFA02**

Used for factory test only.

Bits in this register determine the function of port F pins. Setting a bit assigns the corresponding pin to a control signal; clearing a bit assigns the pin to port F. Refer to **Table D-6**.

Table D-6 Port F Pin Assignments

PFPAR Field	Port F Signal	Alternate Signal
PFPA7	PF7	$\overline{\text{IRQ7}}$
PFPA6	PF6	$\overline{\text{IRQ6}}$
PFPA5	PF5	$\overline{\text{IRQ5}}$
PFPA4	PF4	$\overline{\text{IRQ4}}$
PFPA3	PF3	$\overline{\text{IRQ3}}$
PFPA2	PF2	$\overline{\text{IRQ2}}$
PFPA1	PF1	$\overline{\text{IRQ1}}$
PFPA0	PF0	MODCLK

D.2.12 System Protection Control Register

SYPCR — System Protection Control Register **$YFFA21**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	SWE	SWP	SWT[1:0]		HME	BME	BMT[1:0]	
RESET:	1	$\overline{\text{MODCLK}}$	0	0	0	0	0	0

SYPCR controls system monitor functions, software watchdog clock prescaling, and bus monitor timing. This register can be written once following power-on or reset.

SWE — Software Watchdog Enable
- 0 = Software watchdog is disabled.
- 1 = Software watchdog is enabled.

SWP — Software Watchdog Prescale

This bit controls the value of the software watchdog prescaler.
- 0 = Software watchdog clock is not prescaled.
- 1 = Software watchdog clock is prescaled by 512.

The reset value of SWP is the complement of the state of the MODCLK pin during reset.

SWT[1:0] — Software Watchdog Timing

This field selects the divide ration used to establish software watchdog timeout period. Refer to **Table D-7**.

Table D-12 Reset Pin Function of $\overline{CS}$[10:6]

Data Bus Pins at Reset					Chip-Select/Address Bus Pin Function				
DATA7	DATA6	DATA5	DATA4	DATA3	$\overline{CS10}$/ ADDR23	$\overline{CS9}$/ ADDR22	$\overline{CS8}$/ ADDR21	$\overline{CS7}$/ ADDR20	$\overline{CS6}$/ ADDR19
1	1	1	1	1	$\overline{CS10}$	$\overline{CS9}$	$\overline{CS8}$	$\overline{CS7}$	$\overline{CS6}$
1	1	1	1	0	$\overline{CS10}$	$\overline{CS9}$	$\overline{CS8}$	$\overline{CS7}$	ADDR19
1	1	1	0	X	$\overline{CS10}$	$\overline{CS9}$	$\overline{CS8}$	ADDR20	ADDR19
1	1	0	X	X	$\overline{CS10}$	$\overline{CS9}$	ADDR21	ADDR20	ADDR19
1	0	X	X	X	$\overline{CS10}$	ADDR22	ADDR21	ADDR20	ADDR19
0	X	X	X	X	ADDR23	ADDR22	ADDR21	ADDR20	ADDR19

D.2.18 Chip-Select Base Address Register Boot ROM

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

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

D.2.19 Chip-Select Base Address Registers

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

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

Each chip-select pin has an associated base address register. A base address is the lowest address in the block of addresses enabled by a chip select. CSBARBT contains the base address for selection of a bootstrap memory device. Bit and field definitions for CSBARBT and CSBAR[0:10] are the same, but reset block sizes differ.

ADDR[23:11] — Base Address

This field sets the starting address of a particular chip-select's 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 an integer multiple of the block size. Base address register diagrams show how base register bits correspond to address lines.

BLKSZ[2:0] — Block Size Field

This field determines the size of the block that is enabled by the chip-select.

Table D-13 shows bit encoding for the base address registers block size field.

Table D-14 BYTE Field Bit Encoding

BYTE[1:0]	Description
00	Disable
01	Lower byte
10	Upper byte
11	Both bytes

R/$\overline{W}$[1:0]— Read/Write

This field causes a chip-select to be asserted only for a read, only for a write, or for both read and write. **Table D-15** shows the options.

Table D-15 Read/Write Field Bit Encoding

R/$\overline{W}$[1:0]	Description
00	Disable
01	Read only
10	Write only
11	Read/Write

STRB — Address Strobe/Data Strobe

This bit controls the timing for assertion of a chip-select in asynchronous mode. Selecting address strobe causes the chip-select to be asserted synchronized with address strobe. Selecting data strobe causes the chip-select to be asserted synchronized with data strobe.

0 = Address strobe

1 = Data strobe

$\overline{DSACK}$[3:0] — Data Strobe Acknowledge

This field specifies the source of $\overline{DSACK}$ in asynchronous mode. It also allows the user to adjust bus timing with internal $\overline{DSACK}$ generation by controlling the number of wait states that are inserted to optimize bus speed in a particular application. **Table D-16** shows the $\overline{DSACK}$[3:0] field encoding. The fast termination encoding (%1110) effectively corresponds to –1 wait states.

INDEX

–A–

–B–