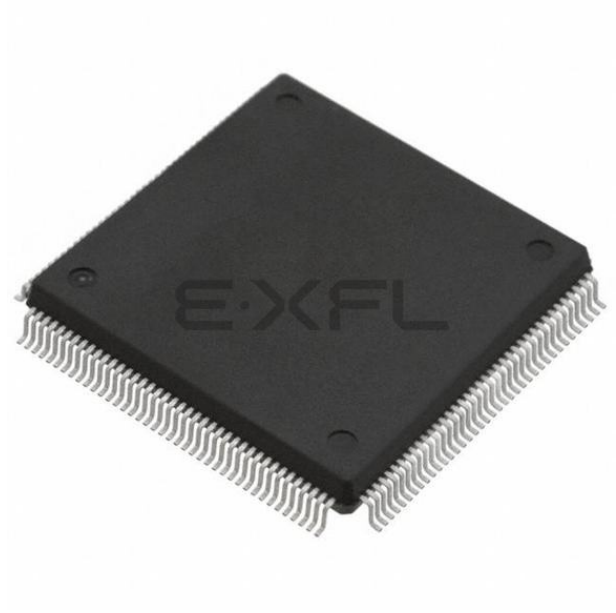


Welcome to E-XFL.COM

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	Not For New Designs
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/sc68376bgmab20

TABLE OF CONTENTS

Paragraph **Title** **Page**

Table 4-2 Instruction Set Summary (Continued)

TBLSN/TBLUN	<ea>, Dn Dym : Dyn, Dn	8, 16, 32	Dyn – Dym ⇒ Temp (Temp ∗ Dn [7 : 0]) / 256 ⇒ Temp Dym + Temp ⇒ Dn
TRAP	#<data>	none	SSP – 2 ⇒ SSP; format/vector offset ⇒ (SSP); SSP – 4 ⇒ SSP; PC ⇒ (SSP); SR ⇒ (SSP); vector address ⇒ PC
TRAPcc	none #<data>	none 16, 32	If cc true, then TRAP exception
TRAPV	none	none	If V set, then overflow TRAP exception
TST	<ea>	8, 16, 32	Source – 0, to set condition codes
UNLK	An	32	An ⇒ SP; (SP) ⇒ An, SP + 4 ⇒ SP

NOTES:
1. Privileged instruction.

4.8.1 M68000 Family Compatibility

It is the philosophy of the M68000 family that all user-mode programs can execute unchanged on future derivatives of the M68000 family, and supervisor-mode programs and exception handlers should require only minimal alteration.

The CPU32 can be thought of as an intermediate member of the M68000 Family. Object code from an MC68000 or MC68010 may be executed on the CPU32. Many of the instruction and addressing mode extensions of the MC68020 are also supported. Refer to the *CPU32 Reference Manual* (CPU32RM/AD) for a detailed comparison of the CPU32 and MC68020 instruction set.

4.8.2 Special Control Instructions

Low-power stop (LPSTOP) and table lookup and interpolate (TBL) instructions have been added to the MC68000 instruction set for use in controller applications.

4.8.2.1 Low-Power Stop (LPSTOP)

In applications where power consumption is a consideration, the CPU32 forces the device into a low-power standby mode when immediate processing is not required. The low-power stop mode is entered by executing the LPSTOP instruction. The processor remains in this mode until a user-specified (or higher) interrupt level or reset occurs.

4.8.2.2 Table Lookup and Interpolate (TBL)

To maximize throughput for real-time applications, reference data is often precalculated and stored in memory for quick access. Storage of many data points can require an inordinate amount of memory. The table lookup instruction requires that only a sample of data points be stored, reducing memory requirements. The TBL instruction recovers intermediate values using linear interpolation. Results can be rounded with a round-to-nearest algorithm.

4.10.11 Deterministic Opcode Tracking

CPU32 function code outputs are augmented by two supplementary signals to monitor the instruction pipeline. The instruction pipe ($\overline{\text{IPIPE}}$) output indicates the start of each new instruction and each mid-instruction pipeline advance. The instruction fetch ($\overline{\text{IFETCH}}$) output identifies the bus cycles in which the operand is loaded into the instruction pipeline. Pipeline flushes are also signaled with $\overline{\text{IFETCH}}$. Monitoring these two signals allows a bus state analyzer to synchronize itself to the instruction stream and monitor its activity.

4.10.12 On-Chip Breakpoint Hardware

An external breakpoint input and on-chip breakpoint hardware allow a breakpoint trap on any memory access. Off-chip address comparators preclude breakpoints unless show cycles are enabled. Breakpoints on instruction prefetches that are ultimately flushed from the instruction pipeline are not acknowledged; operand breakpoints are always acknowledged. Acknowledged breakpoints initiate exception processing at the address in exception vector number 12, or alternately enter background mode.

5.3 System Clock

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

The system clock signal can be generated from one of two sources. An internal phase-locked loop (PLL) can synthesize the clock from a fast reference, or the clock signal can be directly input from an external frequency source. The fast reference is typically a 4.194 MHz crystal, but may be generated by sources other than a crystal. Keep these sources in mind while reading the rest of this section. Refer to **Table A-4** in the **APPENDIX A ELECTRICAL CHARACTERISTICS** for clock specifications.

Figure 5-2 is a block diagram of the clock submodule.

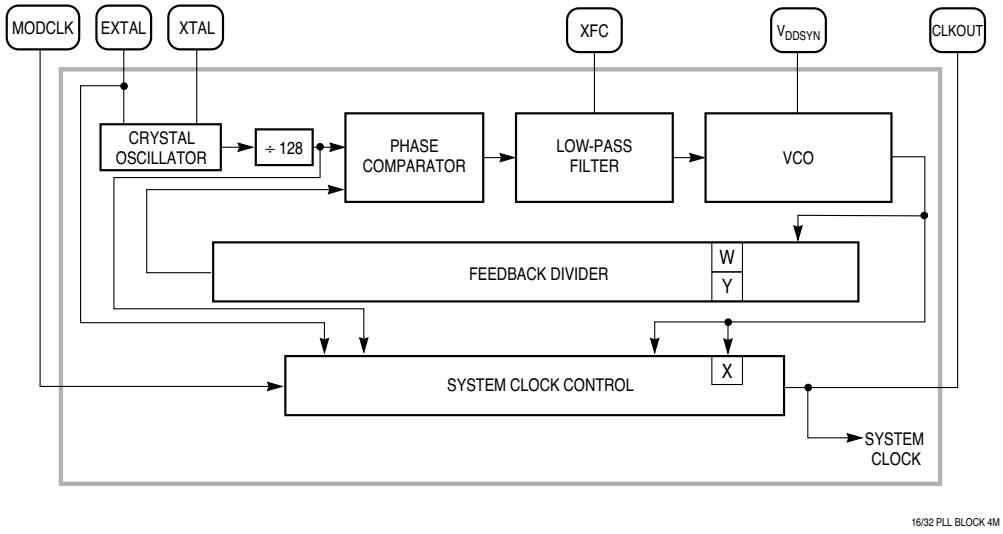


Figure 5-2 System Clock Block Diagram

5.3.1 Clock Sources

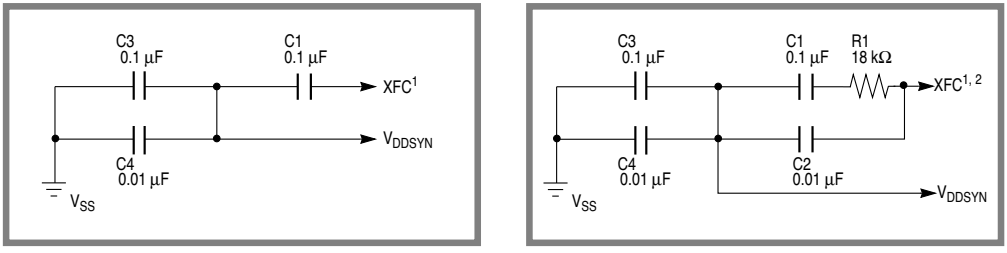
The state of the clock mode (MODCLK) pin during reset determines the system clock source. When MODCLK is held high during reset, the clock synthesizer generates a clock signal from an external reference frequency. The clock synthesizer control register (SYNCR) determines operating frequency and mode of operation. When MODCLK is held low during reset, the clock synthesizer is disabled and an external system clock signal must be driven onto the EXTAL pin.

The input clock is referred to as f_{ref}, and can be either a crystal or an external clock source. The output of the clock system is referred to as f_{sys}. Ensure that f_{ref} and f_{sys} are within normal operating limits.

A voltage controlled oscillator (VCO) in the PLL generates the system clock signal. To maintain a 50% clock duty cycle, the VCO frequency (f_{VCO}) is either two or four times the system clock frequency, depending on the state of the X bit in SYNCR. The clock signal is fed back to a divider/counter. The divider controls the frequency of one input to a phase comparator. The other phase comparator input is a reference signal, either from the crystal oscillator or from an external source. The comparator generates a control signal proportional to the difference in phase between the two inputs. This signal is low-pass filtered and used to correct the VCO output frequency.

Filter circuit implementation can vary, depending upon the external environment and required clock stability. **Figure 5-4** shows two recommended system clock filter networks. XFC pin leakage must be kept as low as possible to maintain optimum stability and PLL performance.

An external filter network connected to the XFC pin is not required when an external system clock signal is applied and the PLL is disabled (MODCLK = 0 at reset). The XFC pin must be left floating in this case.



1. MAINTAIN LOW LEAKAGE ON THE XFC NODE. REFER TO **APPENDIX A ELECTRICAL CHARACTERISTICS** FOR MORE INFORMATION.
2. RECOMMENDED LOOP FILTER FOR REDUCED SENSITIVITY TO LOW FREQUENCY NOISE.

NORMAL/HIGH-STABILITY XFC CONN

Figure 5-4 System Clock Filter Networks

The synthesizer locks when the VCO frequency is equal to f_{ref}. Lock time is affected by the filter time constant and by the amount of difference between the two comparator inputs. Whenever a comparator input changes, the synthesizer must relock. Lock status is shown by the SLOCK bit in SYNCR. During power-up, the MCU does not come out of reset until the synthesizer locks. Crystal type, characteristic frequency, and layout of external oscillator circuitry affect lock time.

6.6 Reset

Reset places the SRAM in low-power stop mode, enables program space access, and clears the base address registers and the register lock bit. These actions make it possible to write a new base address into the registers.

When a synchronous reset occurs while a byte or word SRAM 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 RAM may be corrupted by asynchronous reset. Refer to **5.7 Reset** for more information about resets.

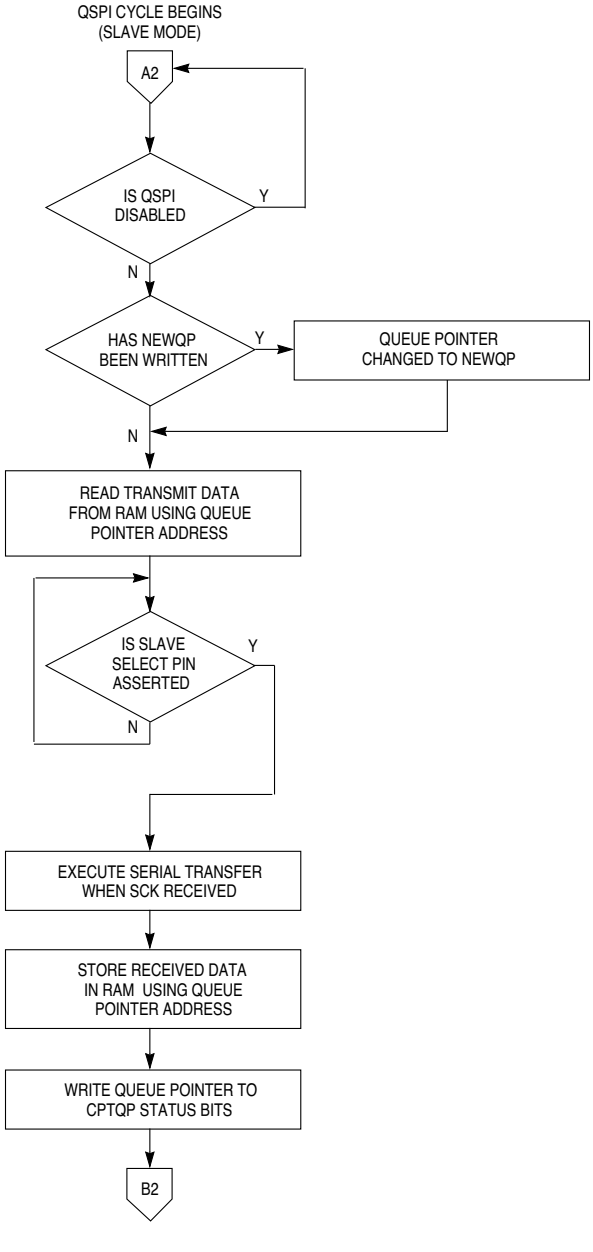


Figure 9-8 Flowchart of QSPI Slave Operation (Part 1)

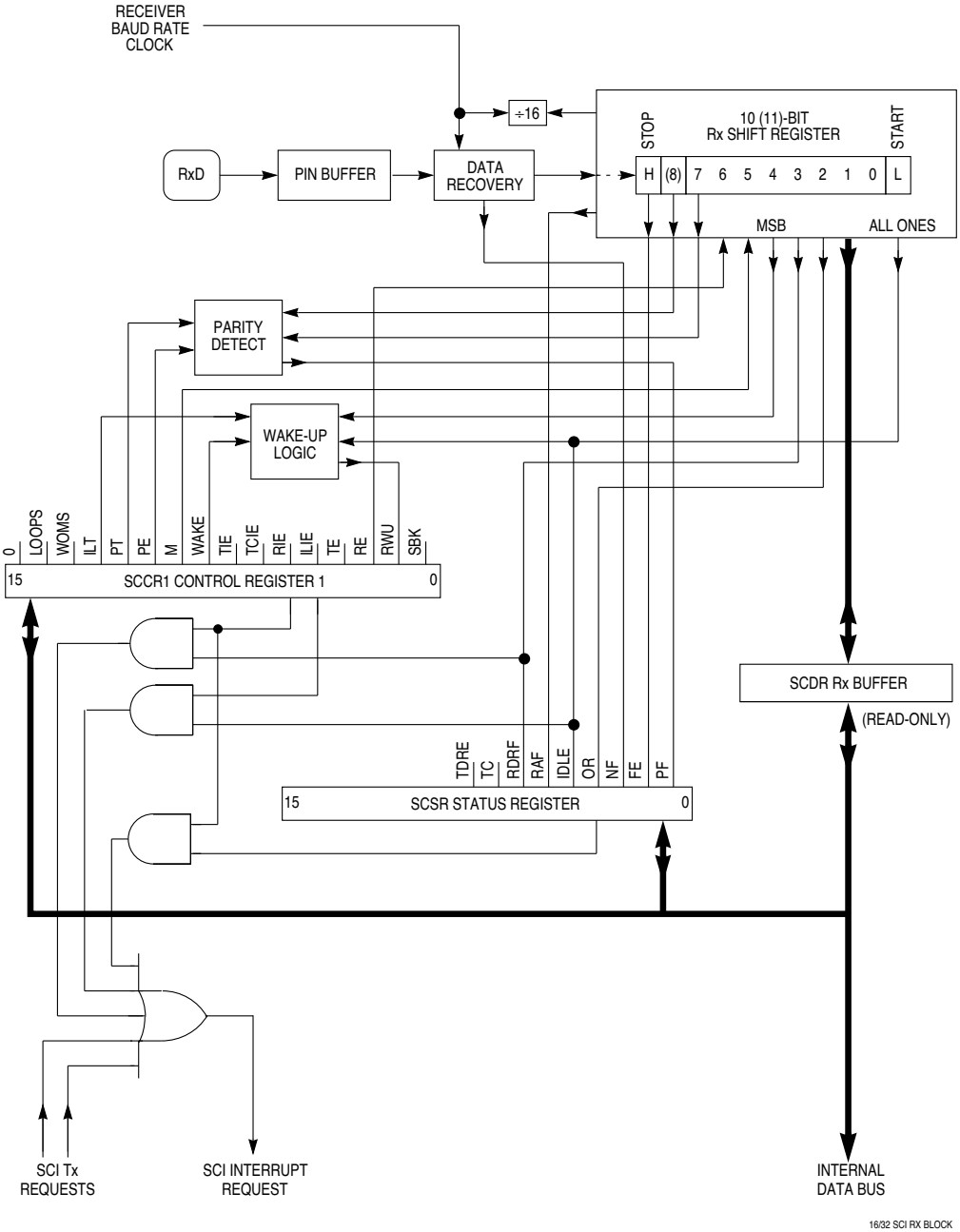


Figure 9-11 SCI Receiver Block Diagram

To accommodate wide variations of the main MCU clock frequency f_{sys}, QCLK is generated by a programmable prescaler which divides the MCU system clock to a frequency within the specified QCLK tolerance range. The prescaler also allows the duty cycle of the QCLK waveform to be programmable.

The basic high phase of the QCLK waveform is selected with the PSH (prescaler clock high time) field in QACR0, and the basic low phase of QCLK with the PSL (prescaler clock low time) field. The duty cycle of QCLK can be further modified with the PSA (prescaler add a clock tick) bit in QACR0. The combination of the PSH and PSL parameters establishes the frequency of QCLK.

Figure 8-8 shows that the prescaler is essentially a variable pulse width signal generator. A 5-bit down counter, clocked at the system clock rate, is used to create both the high phase and the low phase of the QCLK signal. At the beginning of the high phase, the 5-bit counter is loaded with the 5-bit PSH value. When the zero detector finds that the high phase is finished, QCLK is reset. A 3-bit comparator looks for a one's complement match with the 3-bit PSL value, which is the end of the low phase of QCLK. The PSA bit allows the QCLK high-to-low transition to be delayed by a half cycle of the input clock.

The following sequence summarizes the process of determining what values are to be put into the prescaler fields in QACR0:

1. Choose the system clock frequency f_{sys}.
2. Choose first-try values for PSH, PSL, and PSA, then skip to step 4.
3. Choose different values for PSH, PSL, and PSA.
4. If the QCLK high time is less than t_{PSH} (QADC clock duty cycle – Minimum high phase time), return to step 3. Refer to **Table A-13** for more information on t_{PSH}. QCLK high time is determined by the following equation:

$$\text{QCLK high time (in ns)} = \frac{1000\ (1 + \text{PSH} + 0.5\ \text{PSA})}{f_{sys}\text{(in MHz)}}$$

   where PSH = 0 to 31 and PSA = 0 or 1.
5. If QCLK low time is less than t_{PSL} (QADC clock duty cycle – Minimum low phase time), return to step 3. Refer to **Table A-13** for more information on t_{PSL}. QCLK low time is determined by the following equation:

$$\text{QCLK low time (in ns)} = \frac{1000\ (1 + \text{PSL} - 0.5\ \text{PSA})}{f_{sys}\text{(in MHz)}}$$

   where PSL = 0 to 7 and PSA = 0 or 1.

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

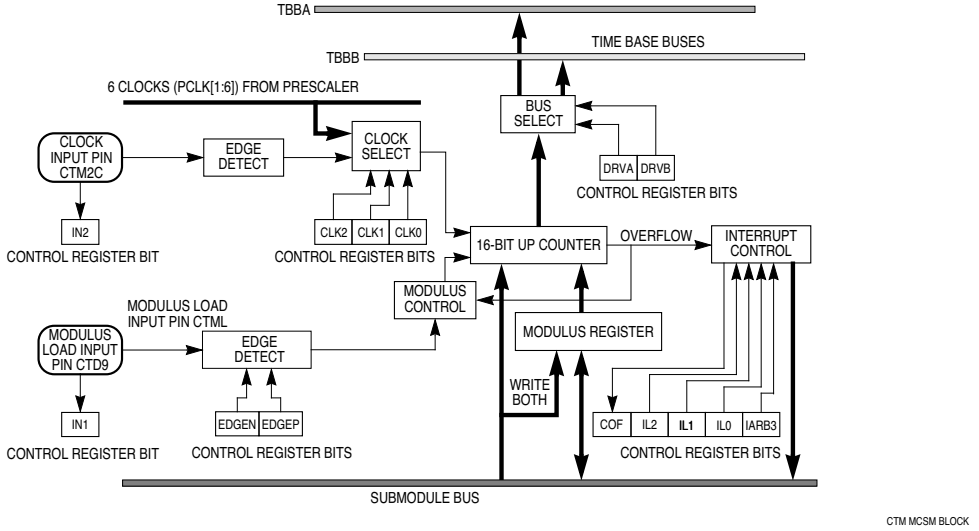


Figure 10-4 MCSM Block Diagram

10.7.1 MCSM Modulus Latch

The 16-bit modulus latch is a read/write register that is used to reload the counter automatically with a predetermined value. The contents of the modulus latch register can be read at any time. Writing to the register loads the modulus latch with the new value. This value is then transferred to the counter register when the next load condition occurs. However, writing to the corresponding counter register loads the modulus latch and the counter register immediately with the new value. The modulus latch register is cleared to $0000 by reset.

10.7.2 MCSM Counter

The counter is composed of a 16-bit read/write register associated with a 16-bit incrementer. Reading the counter transfers the contents of the counter register to the data bus. Writing to the counter loads the modulus latch and the counter register immediately with the new value.

10.7.2.1 Loading the MCSM Counter Register

The MCSM counter is loaded either by writing to the counter register or by loading it from the modulus latch when a counter overflow occurs. Counter overflow will set the COF bit in the MCSM status/interrupt/control register (MCSMSIC).

NOTE

When the modulus latch is loaded with $FFFF, the overflow flag is set on every counter clock pulse.

11.2.3 Scheduler

When a service request is received, the scheduler determines which TPU channel is serviced by the microengine. A channel can request service for one of four reasons: for host service, for a link to another channel, for a match event, or for a capture event. The host system assigns each active channel one of three priorities: high, middle, or low. When multiple service requests are received simultaneously, a priority-scheduling mechanism grants service based on channel number and assigned priority.

11.2.4 Microengine

The microengine is composed of a control store and an execution unit. Control-store ROM holds the microcode for each factory-masked time function. When assigned to a channel by the scheduler, the execution unit executes microcode for a function assigned to that channel by the CPU32. Microcode can also be executed from the TPURAM module instead of the control store. The TPURAM allows emulation and development of custom TPU microcode without the generation of a microcode ROM mask. Refer to **11.3.6 Emulation Support** for more information.

11.2.5 Host Interface

The host interface registers allow communication between the CPU32 and the TPU, both before and during execution of a time function. The registers are accessible from the IMB through the TPU bus interface unit. Refer to **11.6 Host Interface Registers** and **D.8 Time Processor Unit (TPU)** for register bit/field definitions and address mapping.

11.2.6 Parameter RAM

Parameter RAM occupies 256 bytes at the top of the system address map. Channel parameters are organized as 128 16-bit words. Although all parameter word locations in RAM can be accessed by all channels, only 100 are normally used: channels 0 to 13 use six parameter words, while channels 14 and 15 each use eight parameter words. The parameter RAM address map in **D.8.15 TPU Parameter RAM** shows how parameter words are organized in memory.

The CPU32 specifies function parameters by writing to the appropriate RAM address. The TPU reads the RAM to determine channel operation. The TPU can also store information to be read by the CPU32 in the parameter RAM. Detailed descriptions of the parameters required by each time function are beyond the scope of this manual. Refer to the *TPU Reference Manual* (TPURM/AD) and the Motorola TPU Literature Package (TPULITPAK/D) for more information.

11.3 TPU Operation

All TPU functions are related to one of the two 16-bit time bases. Functions are synthesized by combining sequences of match events and capture events. Because the primitives are implemented in hardware, the TPU can determine precisely when a match or capture event occurs, and respond rapidly. An event register for each channel provides for simultaneity of match/capture event occurrences on all channels.

When a match or input capture event requiring service occurs, the affected channel generates a service request to the scheduler. The scheduler determines the priority of the request and assigns the channel to the microengine at the first available time. The microengine performs the function defined by the content of the control store or emulation RAM, using parameters from the parameter RAM.

11.3.1 Event Timing

Match and capture events are handled by independent channel hardware. This provides an event accuracy of one time-base clock period, regardless of the number of channels that are active. An event normally causes a channel to request service. The time needed to respond to and service an event is determined by which channels and the number of channels requesting service, the relative priorities of the channels requesting service, and the microcode execution time of the active functions. Worst-case event service time (latency) determines TPU performance in a given application. Latency can be closely estimated. For more information, refer to the *TPU Reference Manual* (TPURM/AD)

11.3.2 Channel Orthogonality

Most timer systems are limited by the fixed number of functions assigned to each pin. All TPU channels contain identical hardware and are functionally equivalent in operation, so that any channel can be configured to perform any time function. Any function can operate on the calling channel, and, under program control, on another channel determined by the program or by a parameter. The user controls the combination of time functions.

11.3.3 Interchannel Communication

The autonomy of the TPU is enhanced by the ability of a channel to affect the operation of one or more other channels without CPU32 intervention. Interchannel communication can be accomplished by issuing a link service request to another channel, by controlling another channel directly, or by accessing the parameter RAM of another channel.

11.3.4 Programmable Channel Service Priority

The TPU provides a programmable service priority level to each channel. Three priority levels are available. When more than one channel of a given priority requests service at the same time, arbitration is accomplished according to channel number. To prevent a single high-priority channel from permanently blocking other functions, other service requests of the same priority are performed in channel order after the lowest-numbered, highest-priority channel is serviced.

11.3.5 Coherency

For data to be coherent, all available portions of the data must be identical in age, or must be logically related. As an example, consider a 32-bit counter value that is read and written as two 16-bit words. The 32-bit value is read-coherent only if both 16-bit portions are updated at the same time, and write-coherent only if both portions take

13.7 Interrupts

The TouCAN is capable of generating one interrupt level on the IMB. This level is programmed into the priority level bits in the interrupt configuration register (CANICR). This value determines which interrupt signal is driven onto the bus when an interrupt is requested.

When an interrupt is requested, the CPU32 initiates an IACK cycle. The TouCAN decodes the IACK cycle and compares the CPU32 recognized level to the level that it is currently requesting. If a match occurs, then arbitration begins. If the TouCAN wins arbitration, it generates a uniquely encoded interrupt vector that indicates which event is requesting service. This encoding scheme is as follows:

- The higher-order bits of the interrupt vector come from the IVBA[2:0] field in CANICR.
- The low-order five bits are an encoded value that indicate which of the 19 TouCAN interrupt sources is requesting service.

Figure 13-5 shows a block diagram of the interrupt hardware.

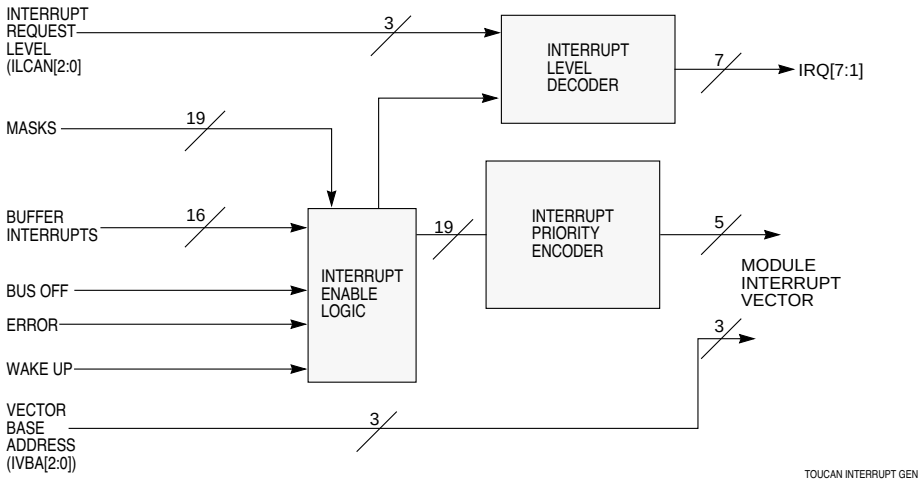


Figure 13-5 TouCAN Interrupt Vector Generation

Each one of the 16 message buffers can be an interrupt source, if its corresponding IMASK bit is set. There is no distinction between transmit and receive interrupts for a particular buffer. Each of the buffers is assigned a bit in the IFLAG register. An IFLAG bit is set when the corresponding buffer completes a successful transmission/reception. An IFLAG bit is cleared when the CPU32 reads IFLAG while the associated bit is set, and then writes it back as zero (and no new event of the same type occurs between the read and the write actions).

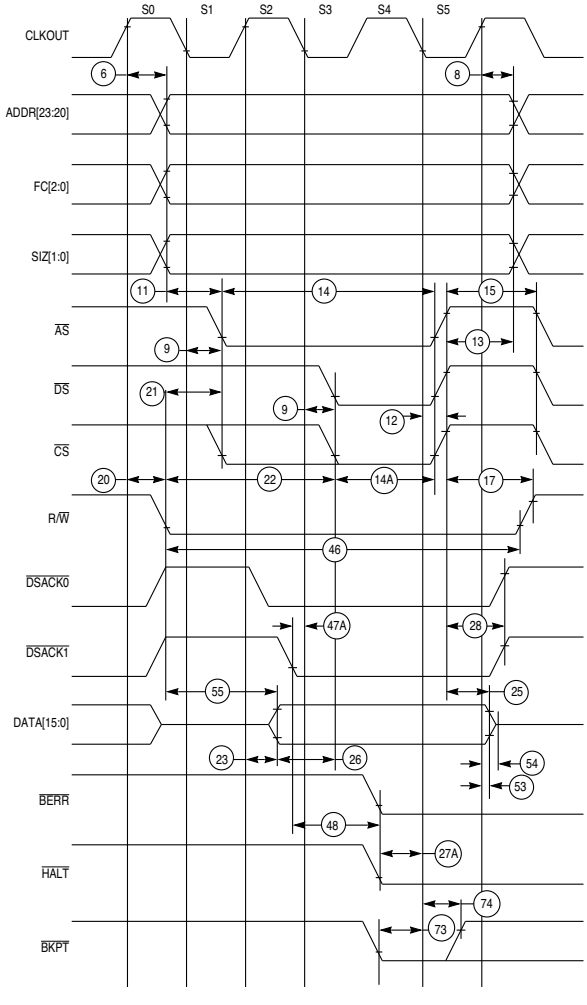


Figure A-5 Write Cycle Timing Diagram

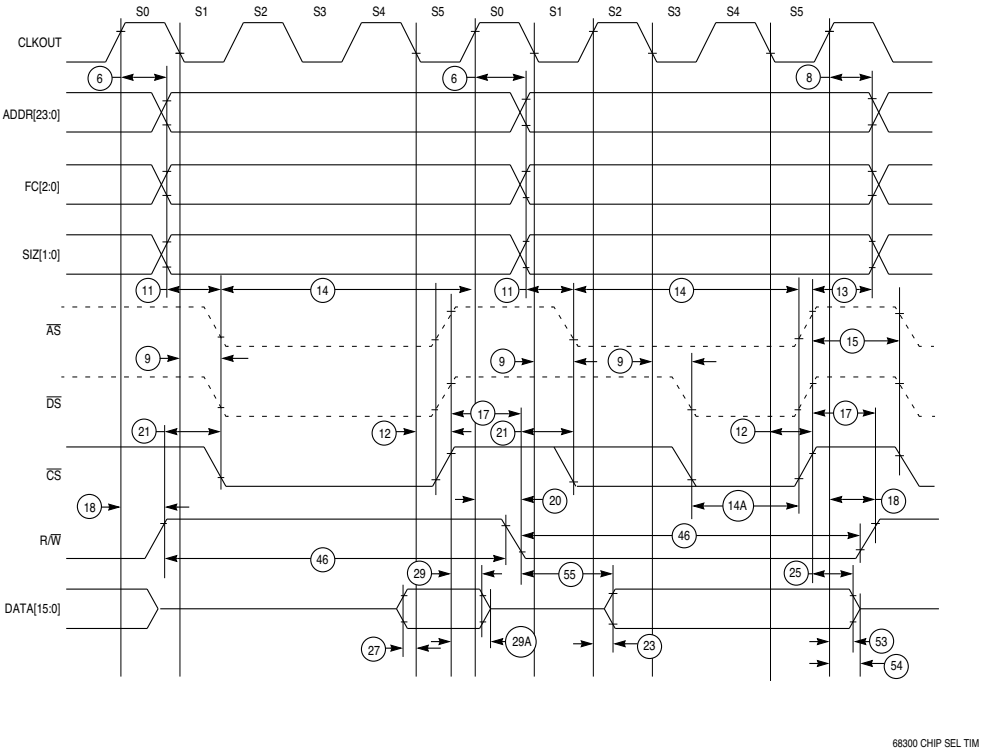


Figure A-11 Chip-Select Timing Diagram

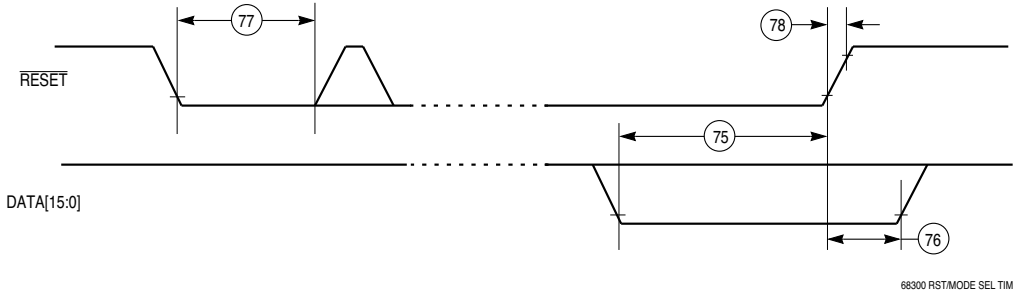


Figure A-12 Reset and Mode Select Timing Diagram

Table A-14 QADC Conversion Characteristics (Operating)

(V_{DDI} and V_{DDA} = 5.0 Vdc ± 5%, V_{SSI} and V_{SSA} = 0 Vdc, $T_A = T_L$ to T_H,
0.5 MHz ≤ f_{QCLK} ≤ 2.1 MHz, 2 clock input sample time)

Num	Parameter	Symbol	Min	Typ	Max	Unit
1	Resolution[1]	1 Count	—	5	—	mV
2	Differential nonlinearity[2]	DNL	—	—	± 0.5	Counts
3	Integral nonlinearity	INL	—	—	± 2.0	Counts
4	Absolute error[2, 3, 4] f_{QCLK} = 0.999 MHz[5] PQA PQB f_{QCLK} = 2.097 MHz[6] PQA PQB	AE	 — — — —	 — — — —	 ± 2.5 ± 2.5 ± 4.0 ± 4.0	Counts
5	Source impedance at input[7]	R_S	—	20	—	kΩ

NOTES:

1. At $V_{RH} - V_{RL}$ = 5.12 V, one count = 5 mV.
2. This parameter is periodically sampled rather than 100% tested.
3. Absolute error includes 1/2 count (2.5 mV) of inherent quantization error and circuit (differential, integral, and offset) error. Specification assumes that adequate low-pass filtering is present on analog input pins — capacitive filter with 0.01 µF to 0.1 µF capacitor between analog input and analog ground, typical source isolation impedance of 20 kΩ.
4. Assumes f_{sys} = 20.97 MHz.
5. Assumes clock prescaler values of:
   QACR0: PSH = %01111, PSA = %1, PSL = 100)
   CCW: BYP = %0
6. Assumes clock prescaler values of:
   QACR0: PSH = %00110, PSA = %1, PSL = 010)
   CCW: BYP = %0
7. Maximum source impedance is application-dependent. Error resulting from pin leakage depends on junction leakage into the pin and on leakage due to charge-sharing with internal capacitance.
   Error from junction leakage is a function of external source impedance and input leakage current. In the following expression, expected error in result value due to junction leakage is expressed in voltage (V_{errj}):
   $$V_{errj} = R_S \text{ X } I_{OFF}$$
   where I_{OFF} is a function of operating temperature. Refer to **Table A-12**.
   Charge-sharing leakage is a function of input source impedance, conversion rate, change in voltage between successive conversions, and the size of the decoupling capacitor used. Error levels are best determined empirically. In general, continuous conversion of the same channel may not be compatible with high source impedance.

D.8 Time Processor Unit (TPU)

Table D-51 shows the TPU address map. The column labeled “Access” indicates the privilege level at which the CPU32 must be operating to access the register. A designation of “S” indicates that supervisor mode is required. A designation of “S/U” indicates that the register can be programmed for either supervisor mode access or unrestricted access.

Table D-51 TPU Register Map

Access	Address[1]	15 0
S	$YFFE00	Module Configuration Register (TPUMCR)
S	$YFFE02	Test Configuration Register (TCR)
S	$YFFE04	Development Support Control Register (DSCR)
S	$YFFE06	Development Support Status Register (DSSR)
S	$YFFE08	TPU Interrupt Configuration Register (TICR)
S	$YFFE0A	Channel Interrupt Enable Register (CIER)
S	$YFFE0C	Channel Function Selection Register 0 (CFSR0)
S	$YFFE0E	Channel Function Selection Register 1 (CFSR1)
S	$YFFE10	Channel Function Selection Register 2 (CFSR2)
S	$YFFE12	Channel Function Selection Register 3 (CFSR3)
S/U	$YFFE14	Host Sequence Register 0 (HSQR0)
S/U	$YFFE16	Host Sequence Register 1 (HSQR1)
S/U	$YFFE18	Host Service Request Register 0 (HSRR0)
S/U	$YFFE1A	Host Service Request Register 1 (HSRR1)
S	$YFFE1C	Channel Priority Register 0 (CPR0)
S	$YFFE1E	Channel Priority Register 1 (CPR1)
S	$YFFE20	Channel Interrupt Status Register (CISR)
S	$YFFE22	Link Register (LR)
S	$YFFE24	Service Grant Latch Register (SGLR)
S	$YFFE26	Decoded Channel Number Register (DCNR)

NOTES:
1. Y = M111, where M represents the logic state of the module mapping (MM) bit in the SIMCR.

D.8.1 TPU Module Configuration Register

TPUMCR — TPU Module Configuration Register **$YFFE00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	TCR1P[1:0]		TCR2P[1:0]		EMU	T2CG	STF	SUPV	PSCK	0	0	IARB[3:0]			
RESET:															
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

STOP — Low-Power Stop Mode Enable
0 = Enable TPU clocks.
1 = Disable TPU clocks.

–N–

–O–

–P–