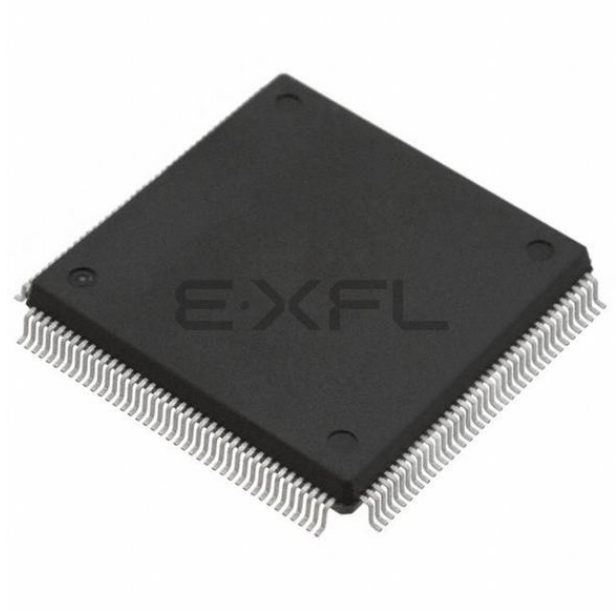


Welcome to E-XFL.COM

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	25MHz
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 ~ 85°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/sc68376bgcab25

3.1.10 CAN 2.0B Controller Module (TouCAN)

- Full implementation of CAN protocol specification, version 2.0 A and B
- 16 receive/transmit message buffers of 0 to 8 bytes data length
- Global mask register for message buffers 0 to 13
- Independent mask registers for message buffers 14 and 15
- Programmable transmit-first scheme: lowest ID or lowest buffer number
- 16-bit free-running timer for message time-stamping
- Low power sleep mode with programmable wake-up on bus activity

3.2 Intermodule Bus

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

3.3 System Block Diagram and Pin Assignment Diagrams

Figure 3-1 is a functional diagram of the MCU. There is not a one-to-one correspondence between location and size of blocks in the diagram and location and size of integrated-circuit modules. **Figure 3-2** shows the MC68336 pin assignment package; **Figure 3-3** shows the MC68376 pin assignment package. Note that the MC68376 is a pin-compatible upgrade for the MC68336 that provides a CAN protocol controller and an 8-Kbyte masked ROM module. Both devices use a 160-pin plastic surface-mount package. Refer to **B.1 Obtaining Updated MC68336/376 Mechanical Information** for package dimensions. Refer to subsequent paragraphs in this section for pin and signal descriptions.

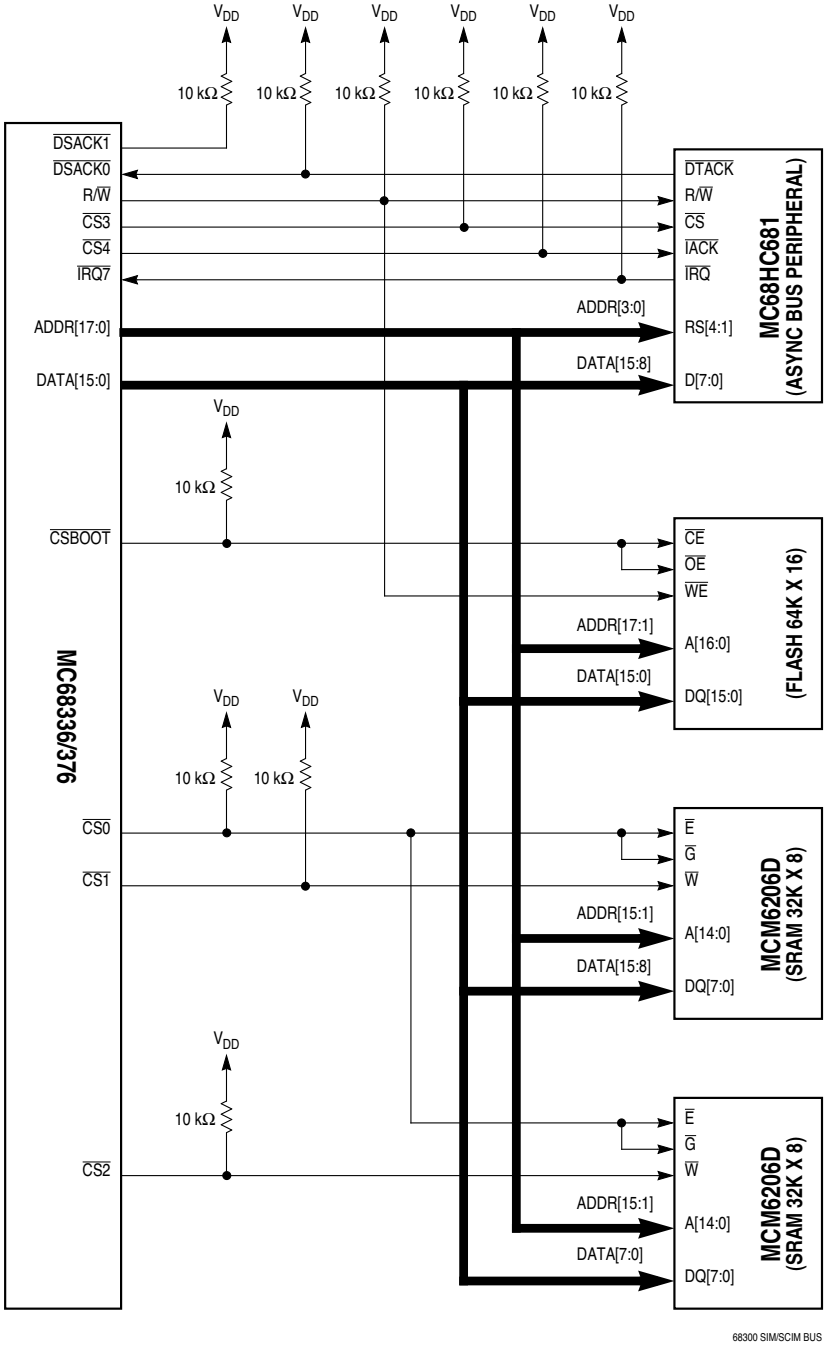


Figure 5-8 MCU Basic System

5.6.2.2 Write Cycle

During a write cycle, the MCU transfers data to an external memory or peripheral device. If the instruction specifies a long-word or word operation, the MCU attempts to write two bytes at once. For a byte operation, the MCU writes one byte. The portion of the data bus upon which each byte is written depends on operand size, peripheral address, and peripheral port size.

Refer to **5.5.2 Dynamic Bus Sizing** and **5.5.4 Misaligned Operands** for more information. **Figure 5-11** is a flowchart of a write-cycle operation for a word transfer. Refer to the *SIM Reference Manual* (SIMRM/AD) for more information.

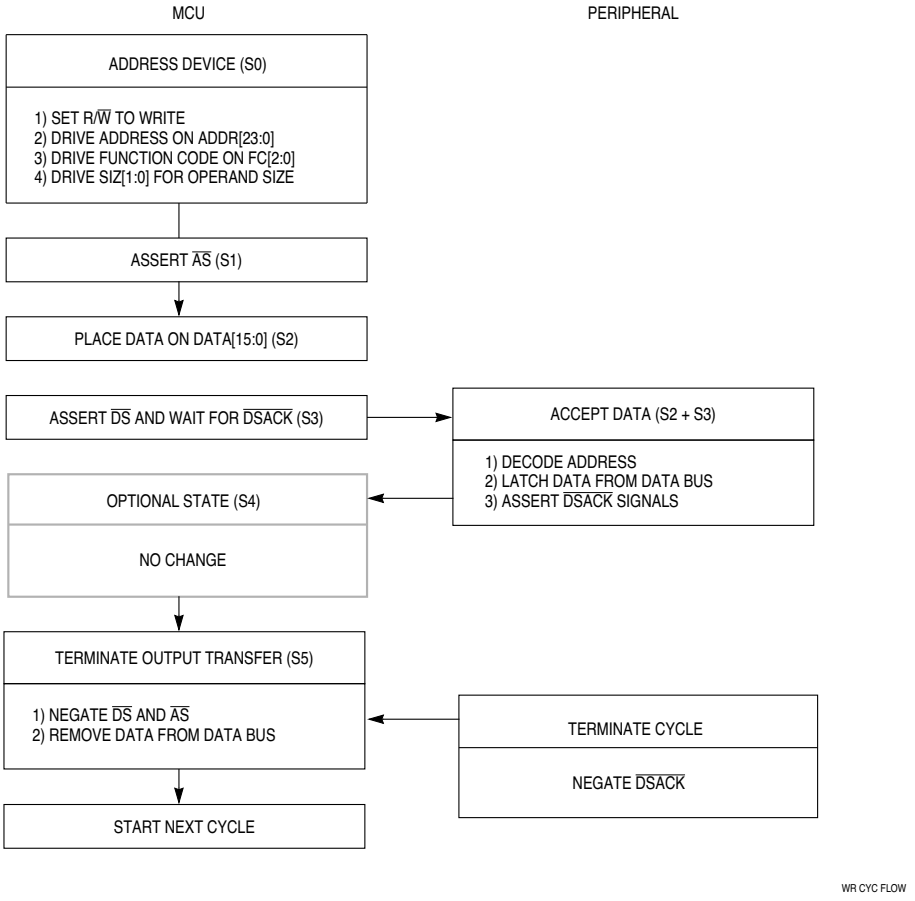


Figure 5-11 Write Cycle Flowchart

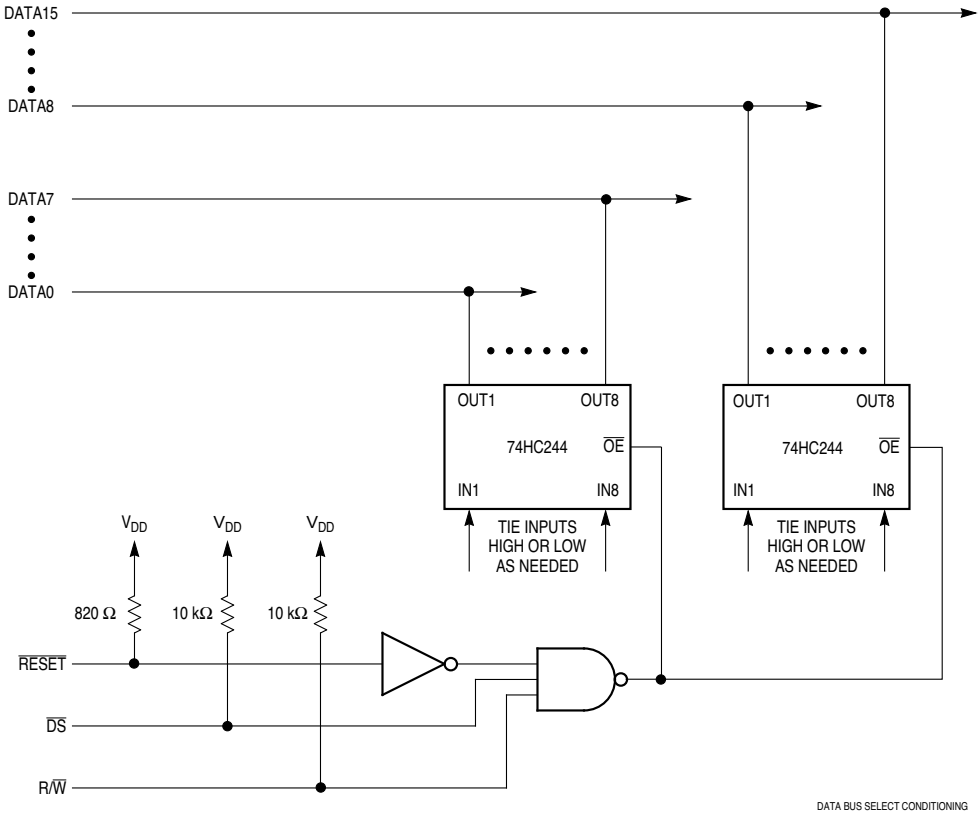


Figure 5-16 Preferred Circuit for Data Bus Mode Select Conditioning

Alternate methods can be used for driving data bus pins low during reset. **Figure 5-17** shows two of these options. The simplest is to connect a resistor in series with a diode from the data bus pin to the $\overline{\text{RESET}}$ line. A bipolar transistor can be used for the same purpose, but an additional current limiting resistor must be connected between the base of the transistor and the $\overline{\text{RESET}}$ pin. If a MOSFET is substituted for the bipolar transistor, only the 1 kΩ isolation resistor is required. These simpler circuits do not offer the protection from potential memory corruption during $\overline{\text{RESET}}$ assertion as does the circuit shown in **Figure 5-16**.

Table 5-19 Pin Assignment Field Encoding

CSxPA[1:0]	Description
00	Discrete output
01	Alternate function
10	Chip-select (8-bit port)
11	Chip-select (16-bit port)

Port size determines the way in which bus transfers to an external address are allocated. Port size of eight bits or sixteen bits can be selected when a pin is assigned as a chip-select. Port size and transfer size affect how the chip-select signal is asserted. Refer to **5.9.1.3 Chip-Select Option Registers** for more information.

Out of reset, chip-select pin function is determined by the logic level on a corresponding data bus pin. The data bus pins have weak internal pull-up drivers, but can be held low by external devices. Refer to **5.7.3.1 Data Bus Mode Selection** for more information. Either 16-bit chip-select function (%11) or alternate function (%01) can be selected during reset. All pins except the boot ROM select pin ($\overline{\text{CSBOOT}}$) are disabled out of reset. There are twelve chip-select functions and only eight associated data bus pins. There is not a one-to-one correspondence. Refer to **5.9.4 Chip-Select Reset Operation** for more detailed information.

The $\overline{\text{CSBOOT}}$ signal is enabled out of reset. The state of the DATA0 line during reset determines what port width $\overline{\text{CSBOOT}}$ uses. If DATA0 is held high (either by the weak internal pull-up driver or by an external pull-up device), 16-bit port size is selected. If DATA0 is held low, 8-bit port size is selected.

A pin programmed as a discrete output drives an external signal to the value specified in the port C register. No discrete output function is available on pins $\overline{\text{CSBOOT}}$, $\overline{\text{BR}}$, $\overline{\text{BG}}$, or $\overline{\text{BGACK}}$. ADDR23 provides the ECLK output rather than a discrete output signal.

When a pin is programmed for discrete output or alternate function, internal chip-select logic still functions and can be used to generate $\overline{\text{DSACK}}$ or $\overline{\text{AVEC}}$ internally on an address and control signal match.

5.9.1.2 Chip-Select Base Address Registers

Each chip-select has an associated base address register. A base address is the lowest address in the block of addresses enabled by a chip-select. Block size is the extent of the address block above the base address. Block size is determined by the value contained in BLKSZ[2:0]. Multiple chip-selects assigned to the same block of addresses must have the same number of wait states.

BLKSZ[2:0] determines which bits in the base address field are compared to corresponding bits on the address bus during an access. Provided other constraints determined by option register fields are also satisfied, when a match occurs, the associated chip-select signal is asserted. **Table 5-20** shows BLKSZ[2:0] encoding.

Serial transfers of eight to sixteen can be specified. Programmable transfer length simplifies interfacing to devices that require different data lengths.

An inter-transfer delay of 17 to 8192 system clocks can be specified (default is 17 system clocks). Programmable delay simplifies the interface to devices that require different delays between transfers.

A dedicated 80-byte RAM is used to store received data, data to be transmitted, and a queue of commands. The CPU32 can access these locations directly. This allows serial peripherals to be treated like memory-mapped parallel devices.

The command queue allows the QSPI to perform up to 16 serial transfers without CPU32 intervention. Each queue entry contains all the information needed by the QSPI to independently complete one serial transfer.

A pointer identifies the queue location containing the data and command for the next serial transfer. Normally, the pointer address is incremented after each serial transfer, but the CPU32 can change the pointer value at any time. Support of multiple-tasks can be provided by segmenting the queue.

The QSPI has four peripheral chip-select pins. The chip-select signals simplify interfacing by reducing CPU32 intervention. If the chip-select signals are externally decoded, 16 independent select signals can be generated.

Wrap-around mode allows continuous execution of queued commands. In wrap-around mode, newly received data replaces previously received data in the receive RAM. Wrap-around mode can simplify the interface with A/D converters by continuously updating conversion values stored in the RAM.

Continuous transfer mode allows transfer of an uninterrupted bit stream. Any number of bits in a range from 8 to 256 can be transferred without CPU32 intervention. Longer transfers are possible, but minimal intervention is required to prevent loss of data. A standard delay of 17 system clocks is inserted between the transfer of each queue entry.

9.3.1 QSPI Registers

The programmer's model for the QSPI consists of the QSM global and pin control registers, four QSPI control registers (SPCR[0:3]), the status register (SPSR), and the 80-byte QSPI RAM. Registers and RAM can be read and written by the CPU32. Refer to **D.6 Queued Serial Module** for register bit and field definitions.

9.3.1.1 Control Registers

Control registers contain parameters for configuring the QSPI and enabling various modes of operation. The CPU32 has read and write access to all control registers. The QSM has read access only to all bits except the SPE bit in SPCR1. Control registers must be initialized before the QSPI is enabled to insure defined operation. SPCR1 must be written last because it contains the QSPI enable bit (SPE).

Writing a new value to any control register except SPCR2 while the QSPI is enabled disrupts operation. SPCR2 is buffered. New SPCR2 values become effective after completion of the current serial transfer. Rewriting NEWQP in SPCR2 causes execu-

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.

8.12.3.3 Continuous-Scan Modes

When application software requires execution of multiple passes through a sequence of conversions defined by a queue, a continuous-scan queue operating mode is selected.

When a queue is programmed for a continuous-scan mode, the single-scan enable bit in the queue control register does not have any meaning or effect. As soon as the queue operating mode is programmed, the selected trigger event can initiate queue execution.

In the case of the software initiated continuous-scan mode, the trigger event is generated internally and queue execution begins immediately. In the other continuous-scan queue operating modes, the selected trigger event must occur before the queue can start. A trigger overrun is recorded if a trigger event occurs during queue execution in the external trigger continuous-scan mode and the periodic timer continuous-scan mode. When a pause is encountered during a scan, another trigger event is required for queue execution to continue. Software involvement is not required for queue execution to continue from the paused state.

After queue execution is complete, the queue status is shown as idle. Since the continuous-scan queue operating modes allow an entire queue to be scanned multiple times, software involvement is not required for queue execution to continue from the idle state. The next trigger event causes queue execution to begin again, starting with the first CCW in the queue.

NOTE

It may not be possible to guarantee coherent samples when using the continuous-scan queue operating modes since the relationship between any two conversions may be variable due to programmable trigger events and queue priorities.

By programming the MQ1 field in QACR1 or the MQ2 field in QACR2, the following modes can be selected for queue 1 and/or 2:

- Software initiated continuous-scan mode
  - When this mode is programmed, the trigger event is generated automatically by the QADC, and queue execution begins immediately. If a pause is encountered, queue execution ceases for two QCLKs, while another trigger event is generated internally; execution then continues. When the end-of-queue is reached, another internal trigger event is generated, and queue execution begins again from the beginning of the queue.
  - While the time to internally generate and act on a trigger event is very short, software can momentarily read the status conditions, indicating that the queue is paused or idle. The trigger overrun flag is never set while in the software initiated continuous-scan mode.

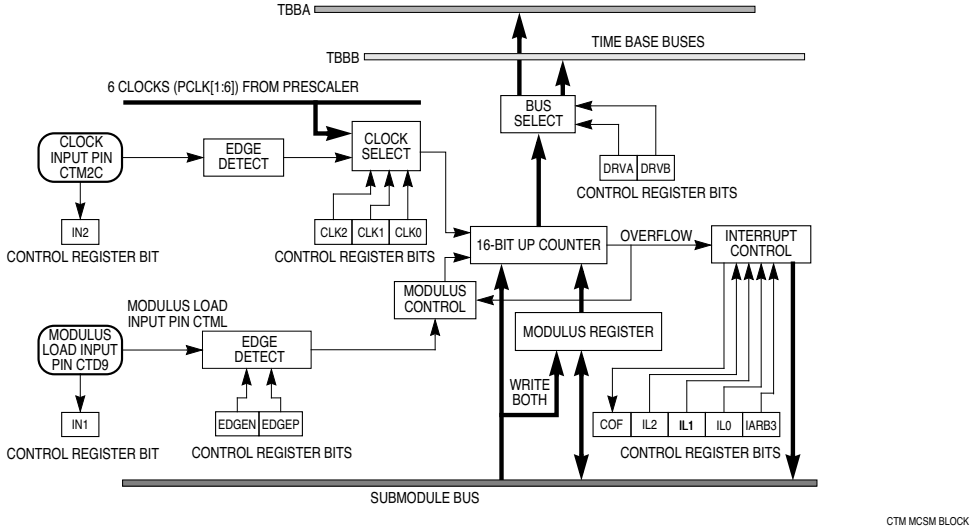


Figure 10-4 MCSM Block Diagram

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.

ator, though internally, channel B has two data registers (B1 and B2). DASM operating mode determines which register is software accessible. Refer to **Table 10-3**.

Table 10-3 Channel B Data Register Access

Mode	Data Register
Input Capture (IPWM, IPM, IC)	Registers A and B2 are used to hold the captured values. In these modes, the B1 register is used as a temporary latch for channel B.
Output Compare (OCA, OCAB)	Registers A and B2 are used to define the output pulse. Register B1 is not used in these modes.
Output Pulse Width Modulation Mode (OPWM)	Registers A and B1 are used as primary registers and hidden register B2 is used as a double buffer for channel B.

Register contents are always transferred automatically at the correct time so that the minimum pulse (measured or generated) is just one time base bus count. The A and B data registers are always read/write registers, accessible via the CTM4 submodule bus.

The CTM4 has four DASMs. **Figure 10-5** shows a block diagram of the DASM.

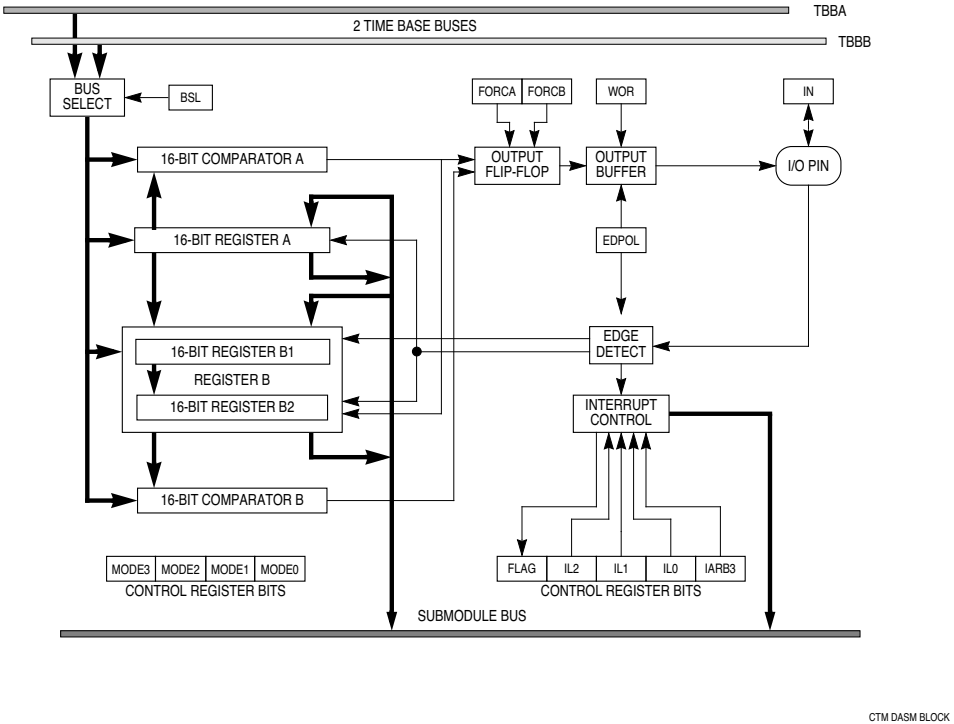


Figure 10-5 DASM Block Diagram

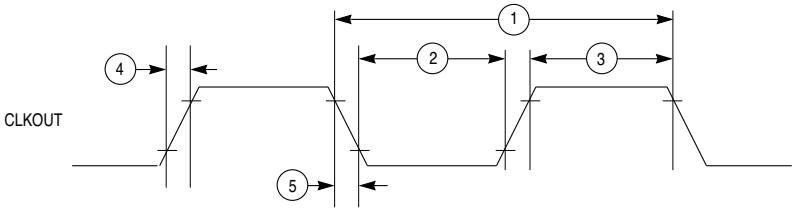
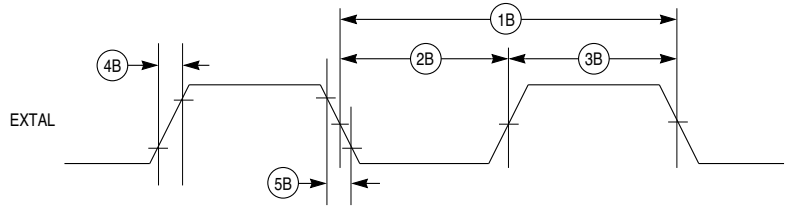
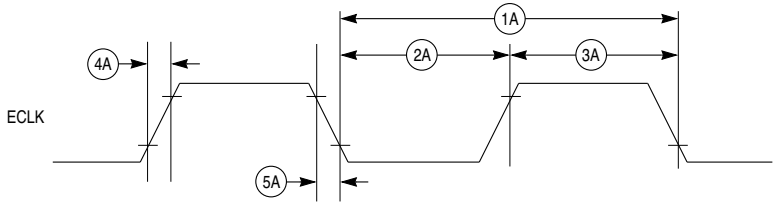


Figure A-1 CLKOUT Output Timing Diagram



NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V_{DD}.
PULSE WIDTH SHOWN WITH RESPECT TO 50% V_{DD}.

Figure A-2 External Clock Input Timing Diagram

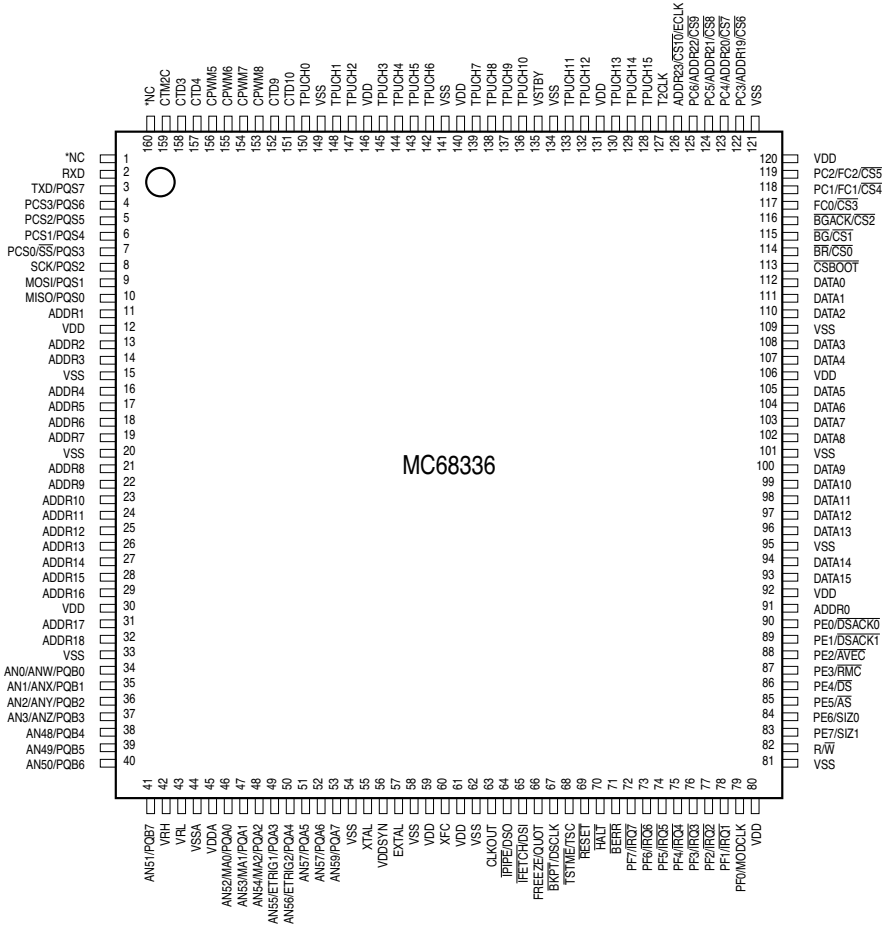


NOTE: TIMING SHOWN WITH RESPECT TO 20% AND 70% V_{DD}.

Figure A-3 ECLK Output Timing Diagram

APPENDIX B MECHANICAL DATA AND ORDERING INFORMATION

The MC68336 and the MC68376 are both available in 160-pin plastic surface mount packages. This appendix provides package pin assignment drawings, a dimensional drawing and ordering information.



*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 B-1 MC68336 Pin Assignments for 160-Pin Package

B.1 Obtaining Updated MC68336/376 Mechanical Information

Although all devices manufactured by Motorola conform to current JEDEC standards, complete mechanical information regarding MC68336/376 microcontrollers is available through Motorola's website at motorola.com

To download updated package specifications, go to website

B.2 Ordering Information

Refer to **Table B-1** for MC68336 ordering information and **Table B-2** for MC68376 ordering information. Contact a Motorola sales representative for information on ordering a custom ROM device.

Table B-1 MC68336 Ordering Information

Part Number	Package Type	Frequency (MHz)	TPU	Temperature	Package Order Quantity	Order Number
MC68336	160-pin QFP	20.97 MHz	A	–40 to +85 °C	2	SPMC68336ACFT20
					24	MC68336ACFT20
					120	MC68336ACFT20B1
				–40 to +105 °C	2	SPMC68336AVFT20
					24	MC68336AVFT20
					120	MC68336AVFT20B1
				–40 to +125 °C	2	SPMC68336AMFT20
					24	MC68336AMFT20
					120	MC68336AMFT20B1
			G	–40 to +85 °C	2	SPMC68336GCFT20
					24	MC68336GCFT20
					120	MC68336GCFT20B1
				–40 to +105 °C	2	SPMC68336GVFT20
					24	MC68336GVFT20
					120	MC68336GVFT20B1
				–40 to +125 °C	2	SPMC68336GMFT20
					24	MC68336GMFT20
					120	MC68336GMFT20B1

Table D-5 Port E Pin Assignments

PEPAR Bit	Port E Signal	Bus Control Signal
PEPA7	PE7	SIZ1
PEPA6	PE6	SIZ0
PEPA5	PE5	$\overline{AS}$
PEPA4	PE4	$\overline{DS}$
PEPA3	PE3	$\overline{RMC}$
PEPA2	PE2	$\overline{AVEC}$
PEPA1	PE1	$\overline{DSACK1}$
PEPA0	PE0	$\overline{DSACK0}$

D.2.9 Port F Data Register

PORTF0— Port F Data Register 0 **$YFFA19**
PORTF1— Port F Data Register 1 **$YFFA1B**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
RESET:	U	U	U	U	U	U	U	U

PORTF is an internal data latch that can be accessed at two locations. It can be read or written at any time. If a port F I/O pin is configured as an output, the corresponding bit value is driven out on the pin. When a pin is configured as an output, a read of PORTF returns the latched bit value; when a pin is configured as an input, a read returns the pin logic level.

D.2.10 Port F Data Direction Register

DDRF — Port F Data Direction Register **$YFFA1D**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0
RESET:	0	0	0	0	0	0	0	0

Bits in this register control the direction of the port F pin drivers when pins are configured for I/O. Setting a bit configures the corresponding pin as an output; clearing a bit configures the corresponding pin as an input. This register can be read or written at any time.

D.2.11 Port F Pin Assignment Register

PFPAR — Port F Pin Assignment Register **$YFFA1F**

15 … 8	7	6	5	4	3	2	1	0
NOT USED	PFPA7	PFPA6	PFPA5	PFPA4	PFPA3	PFPA2	PFPA1	PFPA0
RESET:	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9

PIV[7:0] — Periodic Interrupt Vector

This field specifies the periodic interrupt vector number supplied by the SIM when the CPU32 acknowledges an interrupt request.

D.2.14 Periodic Interrupt Timer Register

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PTP	PITM[7:0]							
RESET:															
0	0	0	0	0	0	0	$\overline{\text{MODCLK}}$	0	0	0	0	0	0	0	0

PITR specifies the prescaling and modulus value for the PIT. This register can be read or written at any time.

PTP — Periodic Timer Prescaler Control

0 = Periodic timer clock is not prescaled.

1 = Periodic timer clock is prescaled by 512.

PITM[7:0] — Periodic Interrupt Timing Modulus

This field determines the periodic interrupt rate. Use the following expressions to calculate timer period.

When a fast reference frequency is used, the PIT period can be calculated as follows:

$$\text{PIT Period} = \frac{(128)(\text{PITM[7:0]})(1 \text{ if PTP} = 0, 512 \text{ if PTP} = 1)(4)}{f_{ref}}$$

When an externally input clock frequency is used, the PIT period can be calculated as follows:

$$\text{PIT Period} = \frac{(\text{PITM[7:0]})(1 \text{ if PTP} = 0, 512 \text{ if PTP} = 1)(4)}{f_{ref}}$$

D.2.15 Software Watchdog Service Register

SWSR — Software Watchdog Service Register[1] **$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

NOTES:

1. Register shown with read value.

–G–

–H–

–I–

–Q–

–T–