Order this document
by MC68F333TS/D

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

MC68F333

Technical Summary
32-Bit Microcontroller

1 MC68F333 Overview

The MC68F333, 32-bit highly-integrated microcontroller, combines high-performance data manipulation capabilities with powerful peripheral subsystems. Based on the powerful MC68020, the CPU32 instruction processing module provides enhanced system performance and also uses the extensive software base for the Motorola M68000 Family.

The MC68F333 incorporates a 32-bit CPU (CPU32), a single-chip integration module (SCIM), an 8-channel, 10-bit analog-to-digital converter (ADC), a time processor unit (TPU), a queued serial module (QSM), a 512-byte standby RAM (SRAM), a 3.5-Kbyte RAM with TPU emulation capabilities (TPURAM), and two flash EEPROM modules. These modules are connected to the CPU32 through the intermodule bus (IMB).

The maximum system clock for the MC68F333 is 16.78 MHz. A phase-locked loop circuit synthesizes the clock from a frequency reference. Either a crystal with a nominal frequency of 32.768 kHz or an externally generated signal can be used. System hardware and software support changes in the clock rate during operation. Because the MC68F333 is a fully static design, register and memory contents are not affected by clock rate changes.

High-density complementary metal-oxide semiconductor (HCMOS) architecture makes the basic power consumption of the MC68F333 low. Power consumption can be minimized by stopping the system clock. The instruction set includes a low-power stop (LPSTOP) command that implements this capability efficiently.

Standard MC68F333 Ordering Information

Package Type	Frequency (MHz)	Temperature	Order Number
Ceramic Surface Mount FE Suffix	16.78	–40°C to +85°C	MC68F333FE
Plastic Surface Mount FC Suffix	16.78	–40°C to +85°C	MC68F333FC

This document contains information on a new product. Specifications and information herein are subject to change without notice.

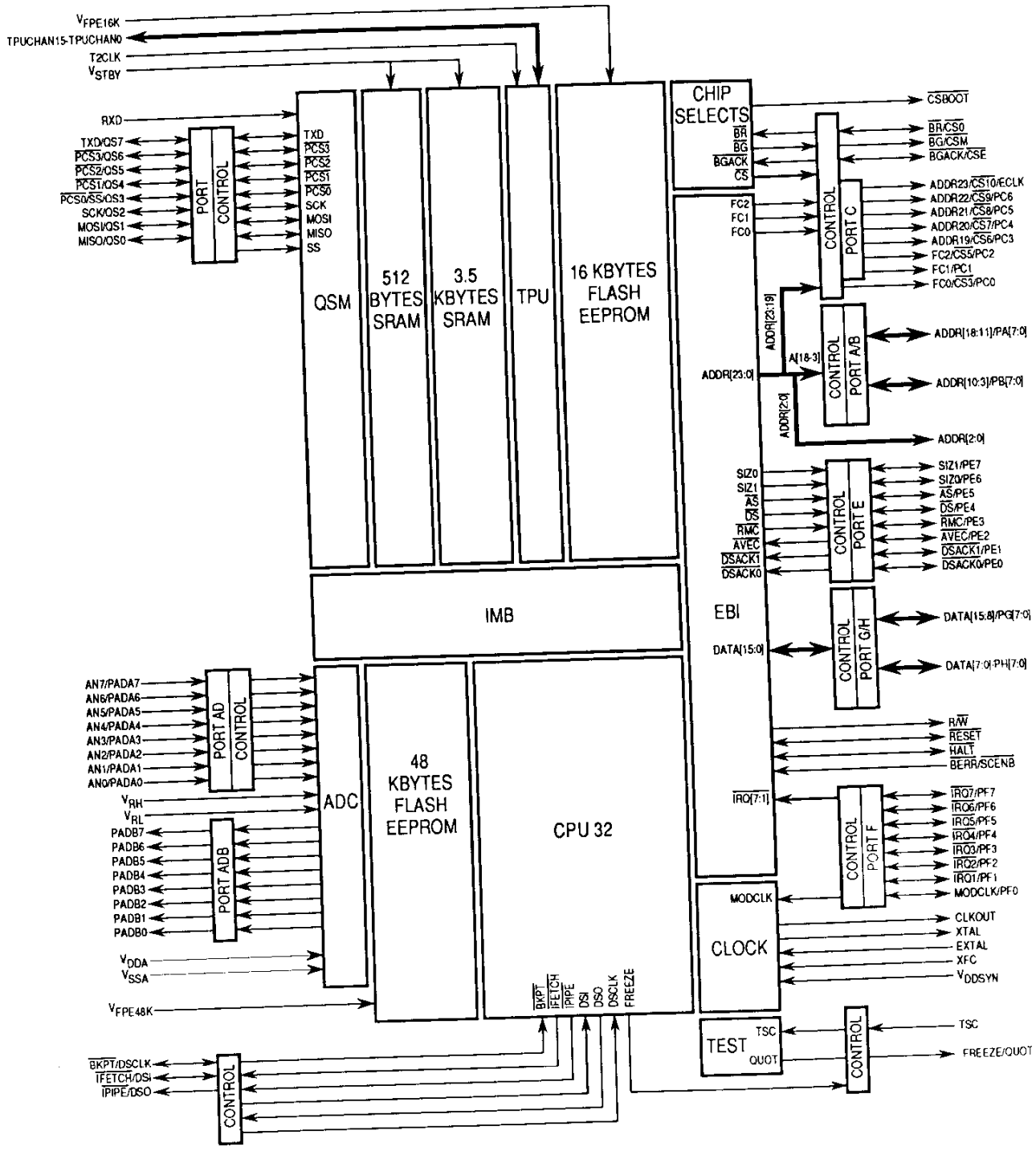
Table of Contents

1.1 Features

- CPU32
  - — 32-Bit Architecture
  - — Virtual Memory Implementation
  - — Table Lookup and Interpolate Instruction
  - — Improved Exception Handling for Controller Applications
- Single-Chip Integration Module (SCIM)
  - — Programmable Chip-Select Outputs
  - — External Bus Support
  - — System Protection Logic
  - — Watchdog Timer, Clock Monitor, and Bus Monitor
  - — PLL Clock System
  - — Up to Six General-Purpose I/O Ports
- Intelligent 16-Bit Time Processor Unit (TPU)
  - — Dedicated Microengine Operating Independently of CPU32
  - — 16 Independent, Programmable Channels and Pins
  - — Any Channel Can Perform Any Time Function
  - — Two Timer-Count Registers with Programmable Prescalers
  - — Selectable Channel Priority Levels
- 8/10-Bit Analog-to-Digital Converter (ADC)
  - — Eight Channels, Eight Result Registers
  - — Eight Conversion Modes
  - — Three Result Alignment Modes
  - — One 8-Bit Digital Input Port
  - — One 8-Bit Digital Output Port
- Flash EEPROM
  - — Two Flash EEPROM Memory Modules (16 Kbyte, 48 Kbyte)
  - — Optional Bootstrap Operation
- Queued Serial Monitor (QSM)
  - — Enhanced Serial Communication Interface
  - — Queued Serial Peripheral Interface
  - — One 8-Bit Dual Function Port
- 3.5-Kbyte Standby RAM with TPU Emulation (TPURAM)
  - — External Standby Voltage Supply Input
  - — May be Used as Standby RAM or TPU Microcode Emulation RAM
- 512-Byte Standby RAM (SRAM)
  - — External Standby Voltage Supply Input

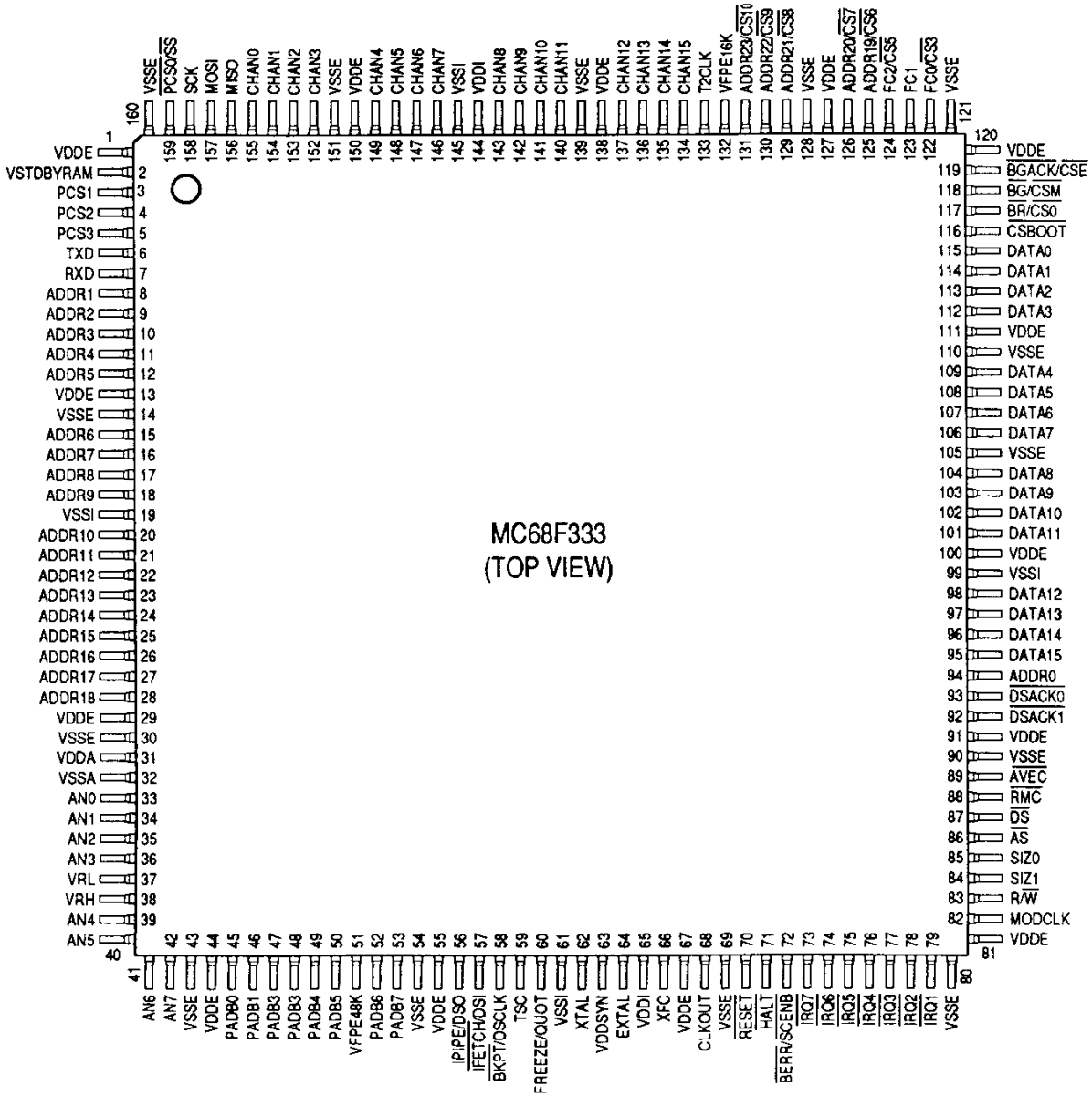
1.2 Block Diagram

A block diagram of the MC68F333 is shown below.



MC68F333 Block Diagram

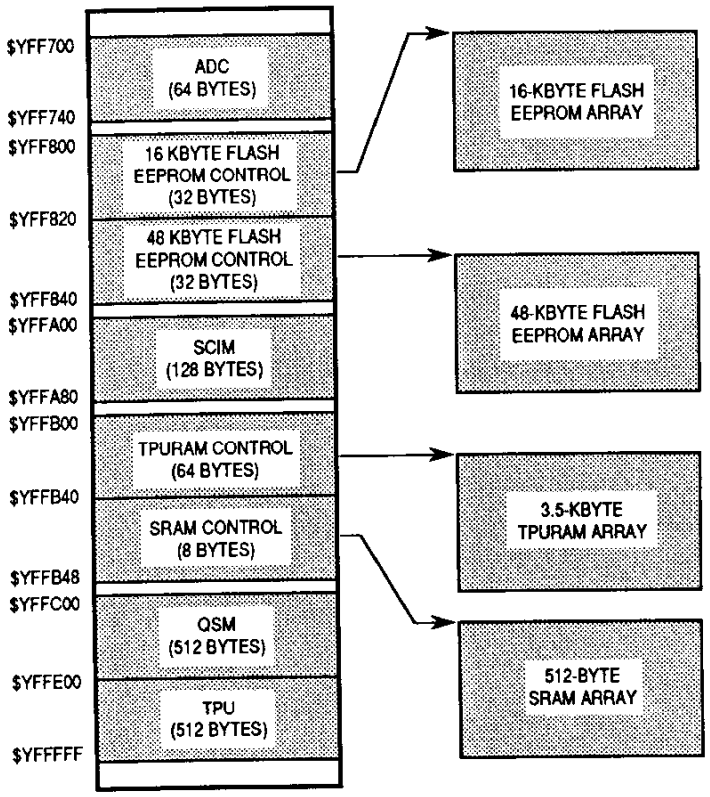
1.3 Pin Assignments



MC68F333 Pin Assignments

1.4 MC68F333 Module Memory Map

The internal address map of the MC68F333 is shown below. The TPURAM, SRAM, 16-Kbyte and 48-Kbyte EEPROM arrays can be mapped via control registers in the corresponding control blocks.



Y = M111, where M is the MODMAP signal state of the MODMAP bit in the SCIM module configuration register (Y = $7 or Y = $F).

MC68F333 Memory Map

2 Signal Descriptions

2.1 MC68F333 Pin Characteristics

The table below describes MC68F333 pin characteristics. All input pins detect CMOS logic levels. All I/O pins can be put in a high-impedance state, but the method of doing this differs depending upon pin function. An entry in the Discrete I/O column indicates that the pin has an alternate I/O function. Port designation is given when it applies. Refer to **2.2 MC68F333 Driver Types** for a description of output drivers.

MC68F333 Pin Characteristics

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
ADDR23/$\overline{CS10}$/ECLK	A	Y	N	—	—
ADDR[22:19]/$\overline{CS[9:6]}$	A	Y	N	O	PC[6:3]
ADDR[18:3]	A	Y	Y	I/O	A[7:0], B[7:0]
ADDR[2:0]	A	Y	N	—	—
AN[7:0]	—	Y[1]	Y	I	PADA[7:0]
$\overline{AS}$	B	Y	Y	I/O	PE5
$\overline{AVEC}$	B	Y	N	I/O	PE2
$\overline{BERR}$	B	Y[2]	N	—	—
$\overline{BG}$/$\overline{CSM}$	B	—	—	—	—
$\overline{BGACK}$/$\overline{CSE}$	B	Y	N	—	—
$\overline{BKPT}$/DSCLK	—	Y	Y	—	—
$\overline{BR}$/$\overline{CS0}$	B	Y	N	—	—
CLKOUT	A	—	—	—	—
$\overline{CSBOOT}$	B	—	—	—	—
DATA[15:0]	AW	Y[1]	Y	I/O	PG[7:0], PH[7:0]
$\overline{DS}$	B	Y	Y	I/O	PE4
$\overline{DSACK1}$	B	Y	N	I/O	PE1
$\overline{DSACK0}$	B	Y	N	I/O	PE0
DSI/$\overline{IFETCH}$	A	Y	Y	—	—
DSO/$\overline{IPIPE}$	A	—	—	—	—
EXTAL	—	—	Special	—	—
FC[2:0]/$\overline{CS[5:3]}$	A	Y	N	O	PC[2:0]
FREEZE/QUOT	A	—	—	—	—
$\overline{HALT}$	BO	Y[2]	N	—	—
$\overline{IRQ[7:1]}$	B	Y	Y	I/O	PF[7:1]
MISO	BO	Y[1]	Y	I/O	QS0
MODCLK	B	Y[1]	Y	I/O	PF0
MOSI	BO	Y[1]	Y	I/O	QS1
PADB[7:0]	A	—	—	O	PADB[7:0]

MC68F333 Pin Characteristics (Continued)

Pin Mnemonic	Output Driver	Input Synchronized	Input Hysteresis	Discrete I/O	Port Designation
PCS0/$\overline{SS}$	BO	Y[1]	Y	I/O	QS3
PCS[3:1]	BO	Y[1]	Y	I/O	QS[6:4]
R/$\overline{W}$	A	Y	N	—	—
$\overline{RESET}$	Bo	Y	Y	—	—
$\overline{RMC}$	B	Y	N	I/O	PE3
RXD	A	N	Y	—	—
SCK	BO	Y[1]	Y	I/O	QS2
SIZ[1:0]	B	Y	Y	I/O	PE[7:6]
TPUCHAN[15:0]	A	Y	Y	—	—
TSC	—	Y	Y	—	—
TXD	BO	Y	Y	I/O	QS7
T2CLK	A	Y	Y	—	—
XFC	—	—	—	—	—
XTAL	—	—	—	—	—

1. DATA[15:0] are synchronized during reset only. MODCLK and ADC pins are synchronized only when used as input port pins.
2. $\overline{BERR}$, HALT only synchronized if late $\overline{BERR}$ or HALT.

2.2 MC68F333 Driver Types

Type	I/O	Description
A	O	Output-only signals that are always driven. No external pullup required.
AW	O	Type A output with weak P-channel pullup during reset.
B	O	Three-state output that includes circuitry to pull up asserted output before high impedance is established, to ensure rapid rise time. An external holding resistor is required to maintain logic level while in the high-impedance state. Pins with this type of driver may only go into high-impedance state under certain conditions. The TSC signal can put all pins with this type of driver in high-impedance state.
BO	O	Type B output that can be operated in an open-drain mode.

2.3 MC68F333 Power Connections

$V_{STBYRAM}$	Standby RAM Power
V_{DDSYN}	Clock Synthesizer Power
V_{DDA}/V_{SSA}	A/D Converter Power
V_{RH}/V_{RL}	A/D Reference Voltage
V_{SSE}/V_{DDE}	External Periphery Power (Source and Drain)
V_{SSI}/V_{DDI}	Internal Module Power (Source and Drain)
V_{FPE16K}, V_{FPE48K}	External Flash EEPROM Programming/Erase Power

2.4 MC68F333 Signal Characteristics

Signal Name	MCU Module	Signal Type	Active State
AN[7:0]	ADC	Input	—
ADDR[23:0]	SCIM	Bus	1/0
$\overline{AS}$	SCIM	Output	0
$\overline{AVEC}$	SCIM	Input	0
$\overline{BERR}$	SCIM	Input	0
$\overline{BG}$	SCIM	Output	0
$\overline{BGACK}$	SCIM	Input	0
$\overline{BKPT}$	CPU32	Input	0
$\overline{BR}$	SCIM	Input	0
CLKOUT	SCIM	Output	—
$\overline{CS}$[10:5], $\overline{CS3}$, $\overline{CS0}$	SCIM	Output	0
$\overline{CSBOOT}$	SCIM	Output	0
$\overline{CSE}$	SCIM	Output	0
$\overline{CSM}$	SCIM	Output	0
DATA[15:0]	SCIM	Bus	1/0
$\overline{DS}$	SCIM	Output	0
$\overline{DSACK1}$, $\overline{DSACK0}$	SCIM	Input	0
DSCLK	CPU32	Input	Serial Clock
DSI	CPU32	Input	Serial Data
DSO	CPU32	Output	Serial Data
ECLK	SCIM	Output	—
EXTAL	SCIM	Input	—
FC[2:0]	SCIM	Output	1/0
FREEZE	SCIM	Output	1
$\overline{HALT}$	SCIM	Input/Output	0
$\overline{IFETCH}$	CPU32	Output	0
$\overline{IPIPE}$	CPU32	Output	0
$\overline{IRQ}$[7:1]	SCIM	Input	0
MISO	QSM	Input/Output	—
MODCLK	SCIM	Input	—
MOSI	QSM	Input/Output	—
PADB[7:0]	ADC	Output	—
PCS[3:0]	QSM	Input/Output	—
QUOT	SCIM	Output	—

MC68F333 Signal Characteristics (Continued)

Signal Name	MCU Module	Signal Type	Active State
R/$\overline{W}$	SCIM	Output	1/0
$\overline{RESET}$	SCIM	Input/Output	0
$\overline{RMC}$	SCIM	Input/Output	0
RXD	QSM	Input	—
SCK	QSM	Input/Output	—
SIZ0/SIZ1	SCIM	Output	1
$\overline{SS}$	QSM	Output	0
TPUCHAN[15:0]	TPU	Input/Output	—
TSC	SCIM	Input	1
TXD	QSM	Output	—
T2CLK	TPU	Input	—
XFC	SCIM	Input	—
XTAL	SCIM	Output	—

2.5 MC68F333 Signal Functions

Mnemonic	Signal Name	Function
ADDR[23:0]	Address Bus	24-bit address bus used by CPU32
AN[7:0]	ADC Analog Input	Inputs to ADC multiplexer
$\overline{AS}$	Address Strobe	Indicates that a valid address is on the address bus
$\overline{AVEC}$	Autovector	Requests an automatic vector during interrupt acknowledge
$\overline{BERR}$	Bus Error	Indicates that a bus error has occurred
$\overline{BG}$	Bus Grant	Indicates that the MCU has relinquished the bus
$\overline{BGACK}$	Bus Grant Acknowledge	Indicates that an external device has assumed bus mastership
$\overline{BKPT}$	Breakpoint	Signals a hardware breakpoint to the CPU
$\overline{BR}$	Bus Request	Indicates that an external device requires bus mastership
CLKOUT	System Clockout	System clock output
$\overline{CS[10:5]}$, $\overline{CS3}$, $\overline{CS0}$	Chip Selects	Select external devices at programmed addresses
$\overline{CSBOOT}$	Boot Chip Select	Chip select for external boot startup ROM
$\overline{CSE}$	Emulator Chip Select	Select external emulation device at internally-mapped address. Used to emulate I/O ports.
$\overline{CSM}$	Internal Module Chip Select	Select external emulation device at internally-mapped address. Not supported by the MC68F333.
DATA[15:0]	Data Bus	16-bit data bus

MC68F333 Signal Functions (Continued)

Mnemonic	Signal Name	Function
$\overline{DS}$	Data Strobe	During a read cycle, indicates that an external device should place valid data on the data bus. During a write cycle, indicates that valid data is on the data bus.
$\overline{DSACK[1:0]}$	Data and Size Acknowledge	Provide asynchronous data transfers and dynamic bus sizing
DSI, DSO, DSCLK	Development Serial In, Out, Clock	Serial I/O and clock for background debug mode
ECLK	E-Clock	External M6800 bus clock output
EXTAL, XTAL	Crystal Oscillator	Connections for clock synthesizer circuit reference; a crystal or an external oscillator can be used
FC[2:0]	Function Codes	Identify processor state and current address space
FREEZE	Freeze	Indicates that the CPU has entered background mode
$\overline{HALT}$	Halt	Suspend external bus activity
$\overline{IFETCH}$	Instruction Fetch	Identifies bus cycles in which operand is loaded into pipeline
$\overline{IPIPE}$	Instruction Pipeline	Indicates instruction pipeline activity
$\overline{IRQ[7:1]}$	Interrupt Request Level	Provide prioritized interrupts to the CPU
MISO	Master In Slave Out	Serial data input to QSPI or output from QSPI
MODCLK	Clock Mode Select	Selects the source and type of system clock
MOSI	Master Out Slave In	Serial output from QSPI or input to QSPI
PADA[7:0]	ADC Digital Input	ADC input port
PADB[7:0]	ADC Digital Output	ADC output port
PCS[3:0]	Peripheral Chip-Selects	Provide QSPI chip selects
QUOT	Quotient Out	Provides the quotient bit of the polynomial divider
$\overline{RESET}$	Reset	System reset
$\overline{BR}$	Read-Modify-Write Cycle	Indicates an indivisible read-modify-write instruction
R/$\overline{W}$	Read/Write	Indicates the direction of data transfer on the bus
RXD	Receive Data	Serial data input to SCI
SCK	QSPI Serial Clock	Clock output from QSPI in master mode; clock input to QSPI in slave mode
SIZ[1:0]	Size	Indicates the number of bytes to be transferred during a bus cycle
$\overline{SS}$	Slave Select	Selects the QSPI when in slave mode
T2CLK	TCR2 Clock	TPU clock input
TPUCHAN[15:0]	TPU I/O Channels	Bidirectional TPU channels
TSC	Three-State Control	Places all output drivers in a high-impedance state
TXD	Transmit Data	Serial data output from SCI
XFC	External Filter Capacitor	Connection for external phase-locked loop filter capacitor

3 Central Processor Unit (CPU32)

The CPU32 is fully object code compatible with the M68000 Family, which excels at processing calculation-intensive algorithms and supporting high-level languages. The CPU32 supports all of the MC68010 and most of the MC68020 enhancements, such as virtual memory support, loop mode operation, instruction pipeline, and 32-bit mathematical operations. Powerful addressing modes provide compatibility with existing software programs and increase the efficiency of high-level language compilers. New instructions, such as table lookup and interpolate and low-power stop, support the specific requirements of controller applications. Also included is the background debug mode, an alternate operating mode that suspends normal operation and allows the CPU to accept debugging commands from the development system.

Ease of programming is an important consideration in using a microcontroller. The CPU32 instruction set is optimized for high performance. The eight 32-bit general-purpose data registers readily support 8-bit (byte), 16-bit (word), and 32-bit (long word) operations. Ease of program checking and diagnosis is further enhanced by trace and trap capabilities at the instruction level.

Use of high-level languages is increasing as controller applications become more complex and control programs become larger. High-level languages aid rapid development of software, with less error, and are readily portable. The CPU32 instruction set supports high-level languages.

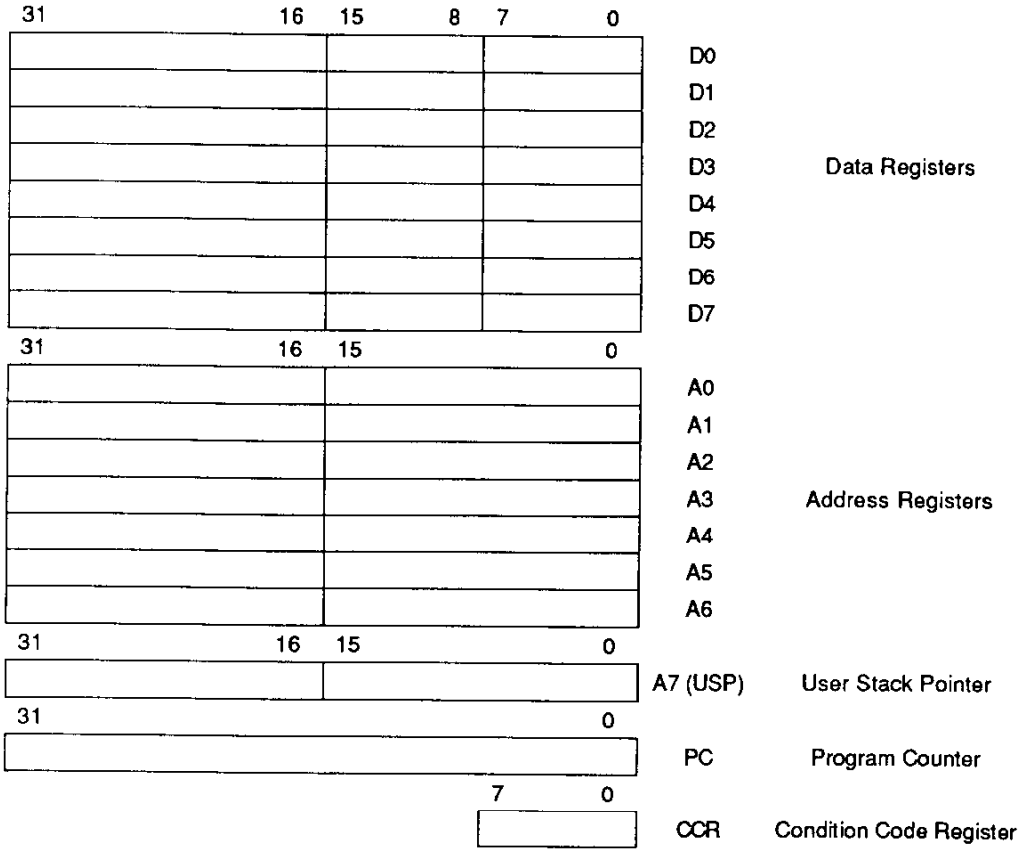
3.1 CPU32 Programming Model

The CPU32 has sixteen 32-bit general registers, a 32-bit program counter, one 32-bit supervisor stack pointer, a 16-bit status register, two alternate function code registers, and a 32-bit vector base register.

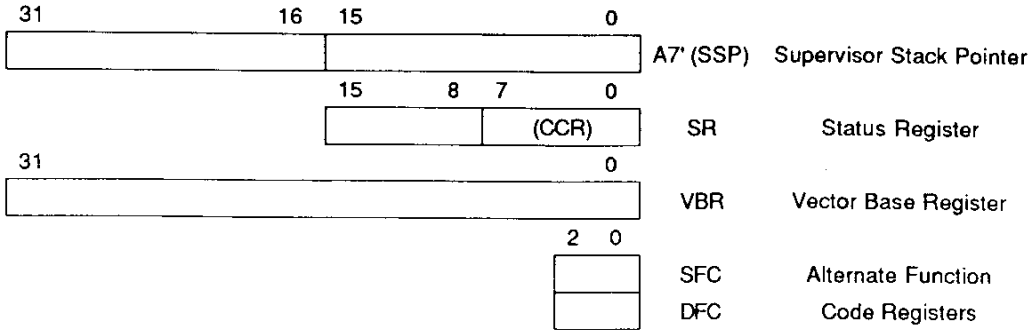
The programming model of the CPU32 consists of a user model and supervisor model, corresponding to the user and supervisor privilege levels. Some instructions available at the supervisor level are not available at the user level, allowing the supervisor to protect system resources from uncontrolled access. Bit S in the status register determines the privilege level.

The user programming model remains unchanged from previous M68000 Family microprocessors. Application software written to run at the nonprivileged user level migrates without modification to the CPU32 from any M68000 platform. The move from SR instruction, however, is privileged in the CPU32. It is not privileged in the M68000.

User Programming Model



Supervisor Programming Model Supplement



3.2 Status Register

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

SR — Status Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T1	T0	S	0	0	I2	I1	I0	0	0	0	X	N	Z	V	C
RESET:															
0	0	1	0	0	1	1	1	0	0	0	U	U	U	U	U

System Byte

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

User Byte (Condition Code Register)

Bits [7:5] — Unimplemented
X — Extend
N — Negative
Z — Zero
V — Overflow
C — Carry

3.3 Data Types

Six basic data types are supported:

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

3.4 Addressing Modes

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

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

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

3.5 Instruction Set Summary

Mnemonic	Description
ABCD	Add Decimal with Extend
ADD	Add
ADDA	Add Address
ADDI	Add Immediate
ADDQ	Add Quick
ADDX	Add with Extend
AND	Logical AND
ANDI	Logical AND Immediate
ASL, ASR	Arithmetic Shift Left and Right
Bcc	Branch Conditionally
BCHG	Test Bit and Change
BCLR	Test Bit and Clear
BGND	Background
BKPT	Breakpoint
BRA	Branch
BSET	Test Bit and Set
BSR	Branch to Subroutine
BTST	Test Bit
CHK, CHK2	Check Register Against Upper and Lower Bounds
CLR	Clear
CMP	Compare
CMPA	Compare Address
CMPI	Compare Immediate
CMPM	Compare Memory to Memory
CMP2	Compare Register Against Upper and Lower Bounds
DBcc	Test Condition, Decrement and Branch
DIVS, DIVSL	Signed Divide
DIVU, DIVUL	Unsigned Divide
EOR	Logical Exclusive OR
EORI	Logical Exclusive OR Immediate
EXG	Exchange Registers
EXT, EXTB	Sign Extend
LEA	Load Effective Address
LINK	Link and Allocate
LPSTOP	Low Power Stop
LSL, LSR	Logical Shift Left and Right
ILLEGAL	Take Illegal Instruction Trap
JMP	Jump
JSR	Jump to Subroutine

Mnemonic	Description
MOVE	Move
MOVE CCR	Move Condition Code Register
MOVE SR	Move Status Register
MOVE USP	Move User Stack Pointer
MOVEA	Move Address
MOVEC	Move Control Register
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral
MOVEQ	Move Quick
MOVES	Move Alternate Address Space
MULS, MULS.L	Signed Multiply
MULU, MULU.L	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NEGX	Negate with Extend
NOP	No Operation
OR	Logical Inclusive OR
ORI	Logical Inclusive OR Immediate
PEA	Push Effective Address
RESET	Reset External Devices
ROL, ROR	Rotate Left and Right
ROXL, ROXR	Rotate with Extend Left and Right
RTD	Return and Deallocate
RTE	Return from Exception
RTR	Return and Restore Codes
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditionally
STOP	Stop
SUB	Subtract
SUBA	Subtract Address
SUBI	Subtract Immediate
SUBQ	Subtract Quick
SUBX	Subtract with Extend
SWAP	Swap Register Words
TBLS,TBLSN	Table Lookup and Interpolate (Signed)
TBLU, TBLUN	Table Lookup and Interpolate (Unsigned)
TAS	Test Operand and Set
TRAP	Trap
TRAPcc	Trap Conditionally
TRAPV	Trap on Overflow
TST	Test Operand
UNLK	Unlink

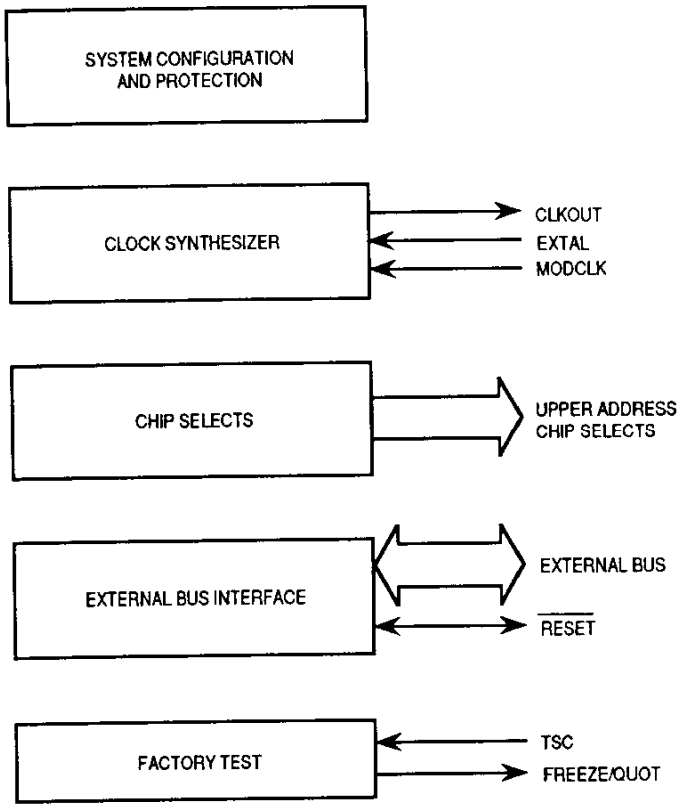
3.6 Background Debug Mode

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

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

4 Single-Chip Integration Module (SCIM)

The MC68F333 single-chip integration module (SCIM) consists of submodules that control system startup, initialization, configuration, external bus operation, and general-purpose I/O with a minimum of external devices. A block diagram of the SCIM is shown below.



SCIM Block Diagram

4.1 SCIM Register Map

FC	Address	15	8 7	0	Submodule
101	YFFA00	SCIM Module Configuration Register (SCIMCR)			Configuration
101	YFFA02	Module Test (SCIMTR)			Test
101	YFFA04	Clock Synthesizer Control (SYNCR)			Clock
101	YFFA06	Unused	Reset Status Register (RSR)		EBI
101	YFFA08	Module Test E (SCIMTRE)			Test
X01	YFFA0A	Port A Data (PORTA)	Port B Data (PORT B)		Port
X01	YFFA0C	Port G Data (PORTG)	Port H Data (PORTH)		Port
X01	YFFA0E	Port G Data Direction (DDRG)	Port H Data Direction (DDRH)		Port
X01	YFFA10	Unused	Port E Data (PORTE)		Port
X01	YFFA12	Unused	Port E Data (PORTE)		Port
X01	YFFA14	Port A/B Data Direction (DDRAB)	Port E Data Direction (DDRE)		Port
101	YFFA16	Unused	Port E Pin Assignment (PEPAR)		Port
X01	YFFA18	Unused	Port F Data (PORTF)		Port
X01	YFFA1A	Unused	Port F Data (PORTF)		Port
X01	YFFA1C	Unused	Port F Data Direction (DDRF)		Port
101	YFFA1E	Unused	Port F Pin Assignment (PFPAR)		EBI
101	YFFA20	Unused	System Protection Control (SYPCR)		System Protect.
101	YFFA22	Periodic Interrupt Control (PICR)			Configuration
101	YFFA24	Periodic Interrupt Timing (PITR)			Configuration
101	YFFA26	Unused	Software Service (SWSR)		System Protect.
101	YFFA28	Unused	Port F Edge Detect Flags (PORTFE)		Port
101	YFFA2A	Unused	Port F Edge Detect Vector (PFIVR)		Port
101	YFFA2C	Unused	Port F Edge Detect Level (PFLVR)		Port
X01	YFFA2E	Unused			
101	YFFA30	Test Module Master Shift A (TSTMSRA)			Test
101	YFFA32	Test Module Master Shift B (TSTMSRB)			Test
101	YFFA34	Test Module Shift Count A (TSTSCA)	Test Module Shift Count B (TSTSCB)		Test
101	YFFA36	Test Module Repetition Counter (TSTRC)			Test
101	YFFA38	Test Module Control (CREG)			Test
X01	YFFA3A	Test Module Distributed Register (DREG)			Test
X01	YFFA3C	Unused	Unused		
X01	YFFA3E	Unused	Unused		
X01	YFFA40	Unused	Port C Data (CSPDR)		Chip Select
X01	YFFA42	Unused	Unused		
101	YFFA44	Chip-Select Pin Assignment (CSPAR0)			Chip Select
101	YFFA46	Chip-Select Pin Assignment (CSPAR1)			Chip Select
101	YFFA48	Chip-Select Base Boot (CSBARBT)			Chip Select
101	YFFA4A	Chip-Select Option Boot (CSORBT)			Chip Select
101	YFFA4C	Chip-Select Base 0 (CSBAR0)			Chip Select

SCIM Register Map (Sheet 2 of 2)

FC	Address	15 8 7 0	Submodule
101	YFFA4E	Chip-Select Option 0 (CSOR0)	Chip Select
101	YFFA50	Unused	
101	YFFA52	Unused	
101	YFFA54	Unused	
101	YFFA56	Unused	
101	YFFA58	Chip-Select Base 3 (CSBAR3)	Chip Select
101	YFFA5A	Chip-Select Option 3 (CSOR3)	Chip Select
101	YFFA5C	Unused	
101	YFFA5E	Unused	
101	YFFA60	Chip-Select Base 5 (CSBAR5)	Chip Select
101	YFFA62	Chip-Select Option 5 (CSOR5)	Chip Select
101	YFFA64	Chip-Select Base 6 (CSBAR6)	Chip Select
101	YFFA66	Chip-Select Option 6 (CSOR6)	Chip Select
101	YFFA68	Chip-Select Base 7 (CSBAR7)	Chip Select
101	YFFA6A	Chip-Select Option 7 (CSOR7)	Chip Select
101	YFFA6C	Chip-Select Base 8 (CSBAR8)	Chip Select
101	YFFA6E	Chip-Select Option 8 (CSOR8)	Chip Select
101	YFFA70	Chip-Select Base 9 (CSBAR9)	Chip Select
101	YFFA72	Chip-Select Option 9 (CSOR9)	Chip Select
101	YFFA74	Chip-Select Base 10 (CSBAR10)	Chip Select
101	YFFA76	Chip-Select Option 10 (CSOR10)	Chip Select
	YFFA78	Unused	
	YFFA7A	Unused	
	YFFA7C	Unused	
	YFFA7E	Unused	

X = Depends on state of SUPV bit in the SCIMCR

Y = M111, where M is the MODMAP bit in the SCIMCR (Y = $7 or $F)

4.2 System Configuration

The MC68F333 can operate as a stand-alone device (single-chip mode), with a 24-bit external address bus and an 8-bit external data bus (partially expanded mode), or with a 24-bit external address bus and a 16-bit external data bus (fully expanded mode). In addition, SCIM pins can be configured for use as I/O ports or programmable chip-select signals. System configuration is determined by setting bits in the SCIM module configuration register (SCIMCR), and by asserting certain MCU pins during reset.

4.2.1 SCIM Module Configuration Register

The module configuration register controls system configuration. It can be read or written at any time, except for the module mapping (MM) bit, which must remain set to one.

SCIMCR — SCIM Module Configuration Register **$YFFA00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXOFF	FRZSW	FRZBM	CPUD	SLVEN	0	SHEN		SUPV	MM	ABD	RWD	IARB			
RESET:															
0	1	1	*	*	0	0	0	1	1	*	*	1	1	1	1

* Reset state is mode dependent — see bit description below.

EXOFF — External Clock Off

0 = The CLKOUT pin is driven from an internal clock source.

1 = The CLKOUT pin is placed in a high-impedance state.

FRZSW — Freeze Software Enable

0 = When FREEZE is asserted, the software watchdog continues to run.

1 = When FREEZE is asserted, the software watchdog is disabled.

FRZBM — Freeze Bus Monitor Enable

0 = When FREEZE is asserted, the periodic interrupt timer counters continue to run.

1 = When FREEZE is asserted, the periodic interrupt timer counters are disabled, preventing interrupts during software debug.

CPUD — CPU Development Support Disable

0 = Instruction pipeline signals available on pins $\overline{\text{IPIPE}}$ and $\overline{\text{IFETCH}}$

1 = Pins $\overline{\text{IPIPE}}$ and $\overline{\text{IFETCH}}$ placed in high-impedance state unless a breakpoint occurs

CPUD is reset to one when the MCU is in single-chip mode and cleared to zero when the MCU is in an expanded mode.

SLVEN — Slave Mode Enabled

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

0 = IMB is not available to an external master.

1 = An external bus master has direct access to the IMB.

This bit is a read-only status bit that reflects the state of DATA11 during reset. Slave mode is used for factory testing. Reset state is the complement of DATA11 during reset in fully expanded mode.

SHEN[1:0] — Show Cycle Enable

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

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

SUPV — Supervisor/Unrestricted Data Space

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

MM — Module Mapping

0 = Internal modules are addressed from $7FF000 — $7FFFFF.

1 = Internal modules are addressed from $FFF000 — $FFFFFF.

This bit can be written only once. Subsequent attempts to change this bit are ignored.

ABD — Address Bus Disable

0 = Pins ADDR[2:0] operate normally.

1 = Pins ADDR[2:0] are disabled.

ABD is reset to one when the MCU is in single-chip mode and cleared to zero when the MCU is in an expanded mode. ABD can be written only once after reset.

RWD — Read/Write Disable

0 = R/$\overline{W}$ signal operates normally

1 = R/$\overline{W}$ signal placed in high-impedance state.

RWD is reset to one when the MCU is in single-chip mode and cleared to zero when the MCU is in an expanded mode. RWD can be written only once after reset.

IARB[3:0] — Interrupt Arbitration

Each module that can generate interrupts, including the SCIM, has an IARB field. Each IARB field has a different value. During an interrupt acknowledge cycle, IARB permits arbitration among simultaneous interrupts of the same priority level. The reset value of IARB is $F. This prevents SCIM interrupts from being discarded. Initialization software must set the IARB field to a lower value in the range $F (highest priority) to $1 (lowest priority) if lower priority interrupts are to be arbitrated.

4.2.2 Operating Modes

During reset, the SCIM configures itself according to the states of the DATA, $\overline{\text{BERR}}$, MODCLK, and $\overline{\text{BKPT}}$ pins. DATA[11:0] provide pin configuration information. $\overline{\text{BERR}}$, MODCLK, and $\overline{\text{BKPT}}$ determine basic operation.

The SCIM can be configured to operate in one of three modes: 16-bit (fully) expanded, 8-bit (partially) expanded, and single chip. Operating mode is determined by the value of the DATA1 and $\overline{\text{BERR}}$ signals coming out of reset.

Basic Configuration Options

Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
MODCLK	Synthesized System Clock	External System Clock
$\overline{\text{BKPT}}$	Background Mode Disabled	Background Mode Enabled
$\overline{\text{BERR}}$	Expanded Mode	Single-Chip Mode
DATA1 (if $\overline{\text{BERR}}$ = 1)	8-Bit Expanded Mode	16-Bit Expanded Mode

$\overline{\text{BERR}}$, $\overline{\text{BKPT}}$, and MODCLK do not have internal pullups and must be driven to the desired state during reset.

Operating mode determines which address and data bus lines are used and which general-purpose I/O ports are available. The table below summarizes address and data bus configuration.

Address and Data Bus Configuration Options

Mode	Address Bus	Data Bus	I/O Ports
16-Bit Expanded	ADDR[18:3]	DATA[15:0]	—
8-Bit Expanded	ADDR[18:3]	DATA[15:8]	DATA[7:0] = Port H
Single Chip	None	None	ADDR[18:11] = Port A ADDR[10:3] = Port B DATA[15:8] = Port G DATA[7:0] = Port H

Many pins on the MC68F333, including data and address bus pins, have multiple functions. Reset value for these pins depends on operating mode. In expanded mode, the value of DATA[11:0] coming out of reset determines the function of these pins. The function of some of these pins can be changed subsequently by writing to the appropriate pin assignment register. Data bus pins have internal pullups and must be pulled low to achieve the desired alternate configuration. The following tables summarize pin configuration options for each operating mode.

4.2.2.1 16-Bit Expanded Mode

In 16-bit expanded mode ($\overline{BERR}$ = 1, DATA1 = 0), pins ADDR[18:3] and DATA[15:0] are configured as address and data pins, respectively. The alternate functions for these pins as ports A, B, G, and H are unavailable.

16-Bit Expanded Mode Reset Configuration

Pin(s) Affected	Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
$\overline{CSBOOT}$	DATA0	$\overline{CSBOOT}$ 16-Bit	$\overline{CSBOOT}$ 8-Bit
DATA[7:0]	DATA1	—	DATA[15:0]
BR/$\overline{CS0}$ FC0/$\overline{CS3}$/PC0 FC1/PC1 FC2/$\overline{CS5}$/PC2	DATA2	$\overline{CS0}$ $\overline{CS3}$ FC1 $\overline{CS5}$	BR FC0 — FC2
ADDR19/$\overline{CS6}$/PC3 ADDR20/$\overline{CS7}$/PC4 ADDR21/$\overline{CS8}$/PC5 ADDR22/$\overline{CS9}$/PC6 ADDR23/$\overline{CS10}$/ECLK	DATA3 . . . DATA7[1] DATA4. . . DATA7[1] DATA5. . . DATA7[1] DATA5. . . DATA7[1] DATA6. . . DATA7[1]	$\overline{CS6}$ $\overline{CS[7:6]}$ $\overline{CS[8:6]}$ $\overline{CS[9:6]}$ $\overline{CS[10:6]}$	ADDR19 ADDR[20:19] ADDR[21:19] ADDR[22:19] ADDR[23:19]
$\overline{DSACK0}$/PE0 $\overline{DSACK1}$/PE1 $\overline{AVEC}$/PE2 $\overline{RMC}$/PE3 $\overline{DS}$/PE4 $\overline{AS}$/PE5 $\overline{SIZ0}$/PE6 $\overline{SIZ1}$/PE7	DATA8	$\overline{DSACK0}$ $\overline{DSACK1}$ $\overline{AVEC}$ $\overline{RMC}$ $\overline{DS}$ $\overline{AS}$ $\overline{SIZ0}$ $\overline{SIZ1}$	PE0 PE1 PE2 PE3 PE4 PE5 PE6 PE7
MODCLK/PF0 $\overline{IRQ[7:1]}$/PF[7:1]	DATA9	MODCLK $\overline{IRQ[7:1]}$	PF0 PF[7:1]
$\overline{BGACK}$/$\overline{CSE}$ $\overline{BG}$/CSM	DATA10	$\overline{BGACK}$ $\overline{BG}$	$\overline{CSE}$[2] $\overline{CSM}$
DATA11	DATA11	Slave Mode Disabled[3]	Slave Mode Enabled[3]

Notes
1. Any mode-select pin in the specified range selects this configuration option.
2. $\overline{CSE}$ is enabled when DATA10 and DATA8 = 0 during reset.
3. Slave mode used for factory test only.

4.2.2.2 8-Bit Expanded Mode

In 8-bit expanded mode ($\overline{BERR}$ = 1, DATA1 = 1), pins DATA[7:0] are configured as port H, an 8-bit I/O port. Pins DATA[15:8] are configured as data pins. Pins ADDR[18:3] are configured as address pins. Emulator mode is always disabled.

8-Bit Expanded Mode Reset Configuration

Pin(s) Affected	Select Pin	Default Function (Pin Left High)	Alternate Function (Pin Pulled Low)
$\overline{CSBOOT}$	None[1]	$\overline{CSBOOT}$ 8-Bit	$\overline{CSBOOT}$ 8-Bit
BR/$\overline{CS0}$ FC0/$\overline{CS3}$/PC0 FC1/PC1 FC2/$\overline{CS5}$/PC2	None[1]	$\overline{CS0}$ $\overline{CS3}$ $\overline{FC1}$ $\overline{CS5}$	$\overline{CS0}$ $\overline{CS3}$ $\overline{FC1}$ $\overline{CS5}$
ADDR19/$\overline{CS6}$/PC3 ADDR20/$\overline{CS7}$/PC4 ADDR21/$\overline{CS8}$/PC5 ADDR22/$\overline{CS9}$/PC6 ADDR23/CS10/ECLK	None[1]	$\overline{CS[10:6]}$	$\overline{CS[10:6]}$
$\overline{DSACK0}$/PE0 $\overline{DSACK1}$/PE1 $\overline{AVEC}$/PE2 $\overline{RMC}$/PE3 $\overline{DS}$/PE4 $\overline{AS}$/PE5 $\overline{SIZ0}$/PE6 $\overline{SIZ1}$/PE7	DATA8	$\overline{DSACK0}$ $\overline{DSACK1}$ $\overline{AVEC}$ $\overline{RMC}$ $\overline{DS}$ $\overline{AS}$ $\overline{SIZ0}$ $\overline{SIZ1}$	PE0 PE1 PE2 PE3 PE4 PE5 PE6 PE7
$\overline{MODCLK}$/PF0 $\overline{IRQ[7:1]}$/PF[7:1]	DATA9	$\overline{MODCLK}$ $\overline{IRQ[7:1]}$	PF0 PF[7:1]
$\overline{BGACK}$/$\overline{CSE}$ $\overline{BG}$/CSM	None[1]	$\overline{BGACK}$ $\overline{BG}$	$\overline{BGACK}$ $\overline{BG}$

[1] In 8-bit expanded mode, these pins have only one possible reset configuration.

4.2.2.3 Single-Chip Mode

In single-chip mode ($\overline{BERR}$ = 0), pins DATA[15:0] are configured as two 8-bit I/O ports. ADDR[18:3] are configured as two 8-bit I/O ports. There is no external data bus path. Expanded mode configuration options are not available: I/O ports A, B, C, E, F, G, and H are always selected. $\overline{BERR}$ can be tied low permanently to select single-chip mode.

Single-chip Mode Reset Configuration

Pin(s) Affected	Function
$\overline{\text{CSBOOT}}$	$\overline{\text{CSBOOT}}$ 8-Bit
ADDR[18:10]	PA[7:0]
ADDR[9:3]	PB[7:0]
BR/$\overline{\text{CS0}}$	$\overline{\text{CS0}}$
FC0/$\overline{\text{CS3}}$/PC0 FC1/PC1 FC2/$\overline{\text{CS5}}$/PC2 ADDR19/$\overline{\text{CS6}}$/PC3 ADDR20/$\overline{\text{CS7}}$/PC4 ADDR21/$\overline{\text{CS8}}$/PC5 ADDR22/$\overline{\text{CS9}}$/PC6	PC[6:0]
ADDR23/$\overline{\text{CS10}}$/ECLK	—
$\overline{\text{DSACK0}}$/PE0 $\overline{\text{DSACK1}}$PE1 $\overline{\text{AVEC}}$/PE2 $\overline{\text{RMC}}$/PE3 $\overline{\text{DS}}$/PE4 $\overline{\text{AS}}$/PE5 $\overline{\text{SIZ0}}$/PE6 $\overline{\text{SIZ1}}$/PE7	PE[7:0]
MODCLK/PF0 $\overline{\text{IRQ}}$[7:1]/PF[7:1]	PF0 PF[7:1]
DATA[15:8]	PG[7:0]
DATA[7:0]	PH[7:0]
$\overline{\text{BGACK}}$/$\overline{\text{CSE}}$ $\overline{\text{BG}}$/CSM	$\overline{\text{BGACK}}$ $\overline{\text{BG}}$

4.2.3 Emulation Support

The SCIM contains support for external port replacement logic that can be used to replicate on-chip ports externally. This emulation support allows system development of a single-chip application in expanded mode.

Emulator mode is a special type of 16-bit expanded operation. It is entered by holding DATA10 low, $\overline{\text{BERR}}$ high, and DATA1 low during reset. In emulator mode, all port A, B, E, G, and H data and data direction registers and the port E pin assignment register are mapped externally. An emulator chip select ($\overline{\text{CSE}}$) is asserted whenever any of these registers are addressed. The signal is asserted on the falling edge of $\overline{\text{AS}}$. The SCIM does not respond to these accesses, allowing external logic, such as a port replacement unit (PRU) to respond. Port C data, port F data and data direction registers, and port F pin assignment register are accessible normally in emulation mode. Refer to **4.7 Chip Selects** for more information concerning the $\overline{\text{CSE}}$ signal.

The internal module chip select ($\overline{\text{CSM}}$) signal is not supported by the MC68F333.

4.3 System Protection

System protection includes a bus monitor, a halt monitor, a spurious interrupt monitor, and a software watchdog timer. These functions reduce the number of external components in a complete control system.

4.3.1 System Protection Register

The system protection control register controls system monitor functions, software watchdog clock prescaling, and bus monitor timing. In operating mode, this register can be written only once following power-on or reset, but can be read at any time. In test mode, it is writable at any time.

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

7	6	5	4	3	2	1	0
SWE	SWP	SWT		DBE	BME	BMT	
RESET:							
1	*	0	0	0	0	0	0

SWE — Software Watchdog Enable

0 = Software watchdog disabled
1 = Software watchdog enabled

SWP — Software Watchdog Prescale

This bit controls the value of the software watchdog prescaler.

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

*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 ratio used to establish software watchdog timeout period. The following table gives the ratio for each combination of SWP and SWT bits.

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

DBE — Double Bus Fault Enable

0 = Disable double bus fault halt monitor function
1 = Enable double bus fault halt monitor function

BME — Bus Monitor External Enable

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

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

BMT[1:0] — Bus Monitor Timing

This field selects a bus monitor timeout period as shown in the table below.

BMT	Bus Monitor Timeout Period
00	64 System Clocks (CLK)
01	32 System Clocks (CLK)
10	16 System Clocks (CLK)
11	8 System Clocks (CLK)

4.3.2 Bus Monitor

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

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

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

4.3.3 Halt Monitor

The halt monitor responds to an assertion of $\overline{\text{HALT}}$ on the internal bus, caused by a double bus fault. This signal is asserted by the CPU after a double bus fault occurs. A flag in the reset status register (RSR) indicates that the last reset was caused by the halt monitor. The halt monitor reset can be inhibited by the DBE bit in the SYPCR.

4.3.4 Spurious Interrupt Monitor

The spurious interrupt monitor causes a bus error exception if no interrupt arbitration occurs during an interrupt-acknowledge cycle.

4.3.5 Software Watchdog

SWSR — Software Service Register **$YFFA27**

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
RESET:							
0	0	0	0	0	0	0	0

Register shown with read value.

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

Perform a software watchdog service sequence as follows:

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

Both writes must occur in the order listed prior to timeout, but any number of instructions can be executed between the two writes.

Watchdog clock rate is affected by SWP and SWT in SYPCR.

When SWT[1:0] are modified, a watchdog service sequence must be performed before the new timeout period will take effect.

The reset value of SWP is the complement of the state of the MODCLK pin on the rising edge of reset.

Software watchdog timeout period is given by the following equation:

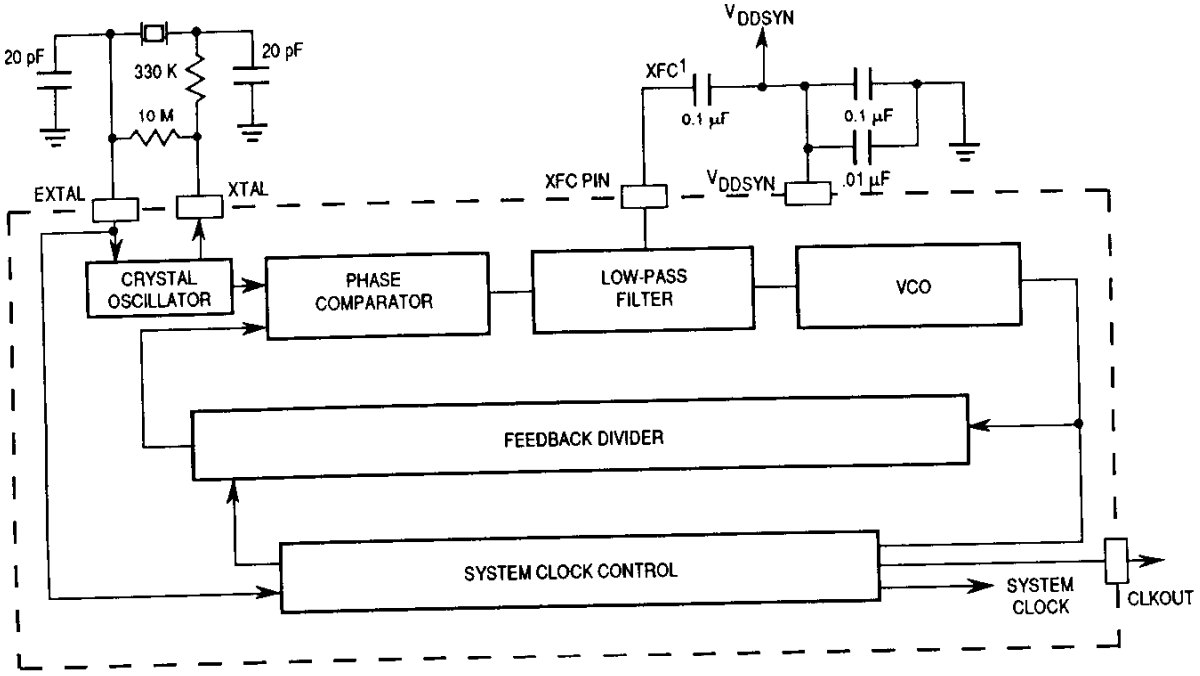
Timeout Period = Divide Count/EXTAL Frequency

4.4 System Clock

The system clock in the SCIM provides timing signals for the IMB modules and for an external peripheral bus. Because the MC68F333 is a fully static design, register and memory contents are not affected when clock rate changes. System hardware and software support changes in clock rate during operation.

The system clock signal can be generated in three ways. An internal phase-locked loop can synthesize the clock from either an internal or an external frequency source, or the clock signal can be input from an external source.

Following is a block diagram of the clock submodule.



Notes:
1. Must be low-leakage capacitor.
2. EXTAL can be driven with an external oscillator.

System Clock Block Diagram

4.4.1 Clock Sources

The state of the clock mode (MODCLK) pin during reset determines clock source. When MODCLK is held high during reset, the clock synthesizer generates a clock signal from either an internal or an external reference frequency — clock synthesizer control register SYNCR determines operating frequency and various modes of operation. When MODCLK is held low during reset, the clock synthesizer is disabled, and an external system clock signal must be applied to EXTAL — SYNCR control bits have no effect.

A reference crystal must be connected between the EXTAL and XTAL pins in order to use the internal oscillator. A 32.768-kHz watch crystal is recommended — these crystals are readily available and inexpensive. MC68F333 clock synthesizer specifications are based upon a typical 32.768-kHz crystal.

If an external reference signal or an external system clock signal is applied via the EXTAL pin, the XTAL pin must be left floating. External reference signal frequency must be less than or equal to maximum specified reference frequency. External system clock signal frequency must be less than or equal to maximum specified system clock frequency.

When an external system clock signal is applied, duty cycle of the input is critical, especially at operating frequencies close to maximum. The relationship between clock signal duty cycle and clock signal period is expressed:

$$\text{Minimum external clock period} = \frac{\text{minimum external clock high/low time}}{50\% - \text{percentage variation of external clock input duty cycle}}$$

4.4.2 Clock Synthesizer Operation

A voltage-controlled oscillator (VCO) generates the system clock signal. A portion of 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 internal oscillator or from an external source. The comparator generates a control signal proportional to the difference in phase between its two inputs. The signal is low-pass filtered and used to correct VCO output frequency.

The synthesizer locks when the frequencies of the two comparator inputs are equal. Lock time is affected by the filter time constant and by the amount of difference between the two comparator inputs. Whenever comparator input changes, the synthesizer must re-lock. Lock status is shown by the SLOCK bit in SYNCR.

The MC68F333 does not come out of reset state until the synthesizer locks. Crystal type, characteristic frequency, and layout of external oscillator circuitry affect lock time.

The low-pass filter requires an external low-leakage capacitor, typically 0.1 µF, connected between the XFC and V_{DDSYN} pins.

V_{DDSYN} is used to power the clock circuits. A separate clean power source increases MCU noise immunity and can be used to run the clock when the MCU is powered down. A quiet power supply must be used as the V_{DDSYN} source since PLL stability depends on the VCO, which uses this supply. Adequate external bypass capacitors should be placed as close as possible to the V_{DDSYN} pin to assure stable operating frequency.

When the clock synthesizer is used, control register SYNCR determines operating frequency and various modes of operation. SYNCR can be read or written in supervisor mode only.

The SYNCR X bit controls a divide by two prescaler that is not in the synthesizer feedback loop. Setting X doubles clock speed without changing VCO speed — there is no VCO relock delay. The SYNCR W bit controls a 3-bit prescaler in the feedback divider. Setting W increases VCO speed by a factor of four. The SYNCR Y field determines the count modulus for a modulo 64 down counter, causing it to divide by a value of Y + 1. When either W or Y value changes, there is a VCO relock delay.

Clock frequency is determined by SYNCR bit settings as follows:

$$F_{SYSTEM} = F_{REFERENCE}\,[4(Y + 1)(2^{2W + X})]$$

In order for the device to perform correctly, the clock frequency selected by the W, X, and Y bits must be within the limits specified for the MCU.

VCO frequency is determined by:

$$F_{VCO} = F_{SYSTEM}\,(2 - X)$$

The reset state of SYNCR ($3F00) produces a modulus-64 count — system frequency is 256 times reference frequency.

4.4.3 Clock Control

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

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

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

When the on-chip clock synthesizer is used (MODCLK = 1 during reset), system clock frequency is controlled by the bits in the upper byte of SYNCR. Bits in the lower byte show the status or control the operation of internal and external clocks. SYNCR can be read or written in supervisor mode only.

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

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

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

EDIV — ECLK Divide Rate
0 = ECLK frequency is system clock divided by 8.
1 = ECLK frequency is system clock divided by 16.
ECLK is an external M6800 bus clock available on pin ADDR23. See **4.7 Chip Selects** for more information.

SLIMP — Limp Mode Flag

0 = External crystal is VCO reference.

1 = Loss of crystal reference.

When the on-chip synthesizer is used, loss of reference frequency causes SLIMP to be set. The VCO continues to run using the base control voltage. Maximum limp frequency is the maximum specified system clock frequency. The state of the X bit affects limp frequency.

SLOCK — Synthesizer Lock Flag

0 = VCO is enabled, but has not locked.

1 = VCO has locked on the desired frequency (or system clock is external).

The MCU maintains reset state until the synthesizer locks, but SLOCK does not indicate synthesizer lock status until after the user writes to SYNCR.

RSTEN — Reset Enable

0 = Loss of crystal causes the MCU to operate in limp mode.

1 = Loss of crystal causes system reset.

STSCIM — Stop Mode System Integration Clock

0 = When LPSTOP is executed, the SCIM clock is driven from the crystal oscillator and the VCO is turned off to conserve power.

1 = When LPSTOP is executed, the SCIM clock is driven from the VCO.

STEXT — Stop Mode External Clock

0 = When LPSTOP is executed, the CLKOUT signal is held negated to conserve power.

1 = When LPSTOP is executed, the CLKOUT signal is driven from the SCIM clock, as determined by the state of the STSCIM bit.

System Frequencies from 32.768-kHz Reference

Y	W = 0; X = 0	W = 0; X = 1	W = 1; X = 0	W = 1; X = 1
0 = 000000	131	262	524	1049
1 = 000001	262	524	1049	2097
2 = 000010	393	786	1573	3146
3 = 000011	524	1049	2097	4194
4 = 000100	655	1311	2621	5243
5 = 000101	786	1573	3146	6291
6 = 000110	918	1835	3670	7340
7 = 000111	1049	2097	4194	8389
8 = 001000	1180	2359	4719	9437
9 = 001001	1311	2621	5243	10486
10 = 001010	1442	2884	5767	11534
11 = 001011	1573	3146	6291	12583
12 = 001100	1704	3408	6816	13631
13 = 001101	1835	3670	7340	14680
14 = 001110	1966	3932	7864	15729
15 = 001111	2097	4194	8389	16777
16 = 010000	2228	4456	8913	—

4.4.4 Periodic Interrupt Timer

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

PICR — Periodic Interrupt Control Register **$YFFA22**

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

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

PIRQL[2:0] — Periodic Interrupt Request Level

The table below shows what interrupt request level is asserted when a periodic interrupt is generated. If a PIT interrupt and an external IRQ of the same priority occur simultaneously, the PIT interrupt is serviced first. The periodic timer continues to run when the interrupt is disabled.

PIRQL	Interrupt Request Level
000	Periodic Interrupt Disabled
001	Interrupt Request Level 1
010	Interrupt Request Level 2
011	Interrupt Request Level 3
100	Interrupt Request Level 4
101	Interrupt Request Level 5
110	Interrupt Request Level 6
111	Interrupt Request Level 7

PIV[7:0] — Periodic Interrupt Vector

The bits of this field contain the vector generated in response to an interrupt from the periodic timer. When the SCIM responds, the periodic interrupt vector is placed on the bus.

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							
RESET:															
0	0	0	0	0	0	0	*	0	0	0	0	0	0	0	0

PITR contains the count value for the periodic timer. A zero value turns off the periodic timer. This register can be read or written at any time.

PTP — Periodic Timer Prescaler Control

1 = Periodic timer clock prescaled by a value of 512

0 = Periodic timer clock not prescaled

*The reset state of PTP is the complement of the state of the MODCLK signal during reset.

PITM[7:0] — Periodic Interrupt Timing Modulus

This is an 8-bit timing modulus. The period of the timer can be calculated as follows:

$$\text{PIT Period} = [(\text{PITM})(\text{Prescaler})(4)]/\text{EXTAL}$$

where

PIT Period = Periodic interrupt timer period
PITM = Periodic interrupt timer register modulus (PITR[7:0])
EXTAL = Crystal frequency
Prescaler = 512 or 1 depending on the state of the PTP bit in the PITR

4.5 External Bus Interface

The external bus interface (EBI) transfers information between the internal MCU bus and external devices when the MCU is operating in partially or fully expanded mode. In fully expanded mode, the external bus has 24 address lines and 16 data lines. In partially expanded mode, the external bus has 24 address lines and 8 data lines.

The EBI provides dynamic sizing between 8-bit and 16-bit data accesses. It supports byte, word, and long-word transfers. Ports are accessed through the use of asynchronous cycles controlled by the data transfer (SIZ1 and SIZ0) and data size acknowledge pins ($\overline{\text{DSACK1}}$ and $\overline{\text{DSACK0}}$). In fully expanded mode, both 8-bit and 16-bit data ports can be accessed; in partially expanded mode, only 8-bit ports can be accessed. Multiple bus cycles may be required for a transfer to an 8-bit port.

Port width is the maximum number of bits accepted or provided during a bus transfer. External devices must follow the handshake protocol described below. Control signals indicate the beginning of the cycle, the address space, the size of the transfer, and the type of cycle. The selected device controls the length of the cycle. Strobe signals, one for the address bus and another for the data bus, indicate the validity of an address and provide timing information for data. The EBI operates in an asynchronous mode for any port width.

To add flexibility and minimize the necessity for external logic, MCU chip-select logic can be synchronized with EBI transfers. Chip-select logic can also provide internally-generated bus control signals for these accesses. See **4.7 Chip Selects** for more information.

4.5.1 Bus Control Signals

The CPU initiates a bus cycle by driving the address, size, function code, and read/write outputs. At the beginning of the cycle, size signals SIZ0 and SIZ1 are driven along with the function code signals. The size signals indicate the number of bytes remaining to be transferred during an operand cycle. They are valid while the address strobe ($\overline{\text{AS}}$) is asserted. The table below shows SIZ0 and SIZ1 encoding. The read/write (R/$\overline{\text{W}}$) signal determines the direction of the transfer during a bus cycle. This signal changes state, when required, at the beginning of a bus cycle, and is valid while $\overline{\text{AS}}$ is asserted. R/$\overline{\text{W}}$ only transitions when a write cycle is preceded by a read cycle or vice versa. The signal may remain low for two consecutive write cycles.

Size Signal Encoding

SIZ1	SIZ0	Transfer Size
0	1	Byte
1	0	Word
1	1	3 Byte
0	0	Long Word

4.5.2 Function Codes

Function code signals FC[2:0] are automatically generated by the CPU32. The function codes can be considered address extensions that automatically select one of eight address spaces to which an address applies. These spaces are designated as either user or supervisor, and program or data spaces. Address space 7 is designated CPU space. CPU space is used for control information not normally associated with read or write bus cycles. Function codes are valid while $\overline{AS}$ is asserted.

CPU32 Address Space Encoding

FC2	FC1	FC0	Address Space
0	0	0	Reserved (Motorola)
0	0	1	User Data Space
0	1	0	User Program Space
0	1	1	Reserved
1	0	0	Reserved
1	0	1	Data Space
1	1	0	Program Space
1	1	1	CPU Space

4.5.3 Address Bus

Address bus signals ADDR[19:0] define the address of the most significant byte to be transferred during a bus cycle. The MCU places the address on the bus at the beginning of a bus cycle. The address is valid while $\overline{AS}$ is asserted.

4.5.4 Address Strobe

$\overline{AS}$ is a timing signal that indicates the validity of an address on the address bus and of many control signals. It is asserted one-half clock after the beginning of a bus cycle.

4.5.5 Data Bus

Data bus signals DATA[15:0] comprise a bidirectional, nonmultiplexed parallel bus that transfers data to or from the MCU. A read or write operation can transfer 8 or 16 bits of data in one bus cycle. During a read cycle, the data is latched by the MCU on the last falling edge of the clock for that bus cycle. For a write cycle, all 16 bits of the data bus are driven, regardless of the port width or operand size. The MCU places the data on the data bus one-half clock cycle after $\overline{AS}$ is asserted in a write cycle.

4.5.6 Data Strobe

Data strobe ($\overline{DS}$) is a timing signal. For a read cycle, the MCU asserts $\overline{DS}$ to signal an external device to place data on the bus. $\overline{DS}$ is asserted at the same time as $\overline{AS}$ during a read cycle. For a write cycle, $\overline{DS}$ signals an external device that data on the bus is valid. The MCU asserts DS one full clock cycle after the assertion of $\overline{AS}$ during a write cycle.

4.5.7 Bus Cycle Termination Signals

During bus cycles, external devices assert the data transfer and size acknowledge signals ($\overline{DSACK1}$ and $\overline{DSACK0}$). During a read cycle, the signals tell the MCU to terminate the bus cycle and to latch data. During a write cycle, the signals indicate that an external device has successfully stored data and that the cycle may terminate. These signals also indicate to the MCU the size of the port for the bus cycle just completed. (Refer to **4.5.8.1 Dynamic Bus Sizing**.)

The bus error ($\overline{BERR}$) signal is also a bus cycle termination indicator and can be used in the absence of $\overline{DSACK1}$ and $\overline{DSACK0}$ to indicate a bus error condition. It can also be asserted in conjunction with these signals, provided it meets the appropriate timing requirements. The internal bus monitor can be used to generate the $\overline{BERR}$ signal for internal and internal-to-external transfers. When $\overline{BERR}$ and $\overline{HALT}$ are asserted simultaneously, the CPU32 takes a bus error exception.

Finally, autovector signal ($\overline{AVEC}$) can be used to terminate external $\overline{IRQ}$ pin interrupt-acknowledge cycles. $\overline{AVEC}$ indicates that the MCU will internally generate a vector number to locate an interrupt handler routine. If it is continuously asserted, autovectors will be generated for all external interrupt requests. $\overline{AVEC}$ is ignored during all other bus cycles.

4.5.8 Data Transfer Mechanism

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

4.5.8.1 Dynamic Bus Sizing

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

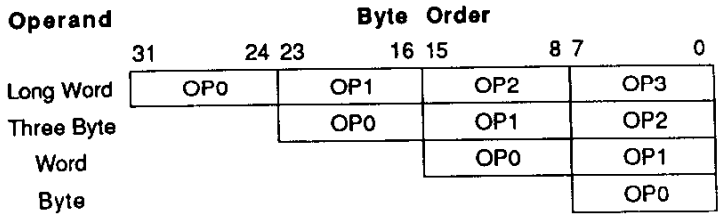
Effect of $\overline{DSACK}$ Signals

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

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

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

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



4.5.8.2 Operand Alignment

Refer to the operand byte order table for required organization of 8- and 16-bit data ports on the MCU bus. The data multiplexer establishes the necessary connections for different combinations of address and data sizes. The multiplexer takes the two bytes of the 16-bit bus and routes them to their required positions. Positioning of bytes is determined by the size and address outputs. SIZ1 and SIZ0 indicate the remaining number of bytes to be transferred during the current bus cycle. The number of bytes transferred is equal to or less than the size indicated by SIZ1 and SIZ0, depending on port width.

ADDR0 also affects the operation of the data multiplexer. During an operand transfer, ADDR[23:1] indicate the word base address of the portion of the operand to be accessed, and ADDR0 indicates the byte offset from the base.

4.5.8.3 Misaligned Operands

In the MC68F333 architecture, the basic operand size is 16 bits. An operand is misaligned when it overlaps a word boundary. This is determined by ADDR0. When ADDR0 = 0, the address is a word and byte boundary. When ADDR0 = 1, the address is a byte boundary only. A byte operand is properly aligned at any address; a word or long-word operand is misaligned at an odd address. The MC68F333 does not support misaligned operand transfers.

A bus cycle can transfer at most a word of data. If the MCU transfers a long-word operand over a 16-bit port, the most significant operand word is transferred on the first bus cycle and the least significant operand word on a following bus cycle.

4.5.8.4 Operand Transfer Cases

The following table summarizes how operands are aligned for various types of transfers. OPn entries are portions of a requested operand that are read or written during a bus cycle and are defined by SIZ1, SIZ0, and ADDR0 for that bus cycle.

Operand Transfer Cases

Transfer Case	SIZ1	SIZ0	ADDR 0	$\overline{\text{DSACK1}}$	$\overline{\text{DSACK0}}$	DATA [15:8]	DATA [7:0]
Byte to 8-bit Port (Even/Odd)	0	1	X	1	0	OP0	(OP0)
Byte to 16-bit Port (Even)	0	1	0	0	X	OP0	(OP0)
Byte to 16-bit Port (Odd)	0	1	1	0	X	(OP0)	OP0
Word to 8-bit Port (Aligned)	1	0	0	1	0	OP0	(OP1)
Word to 8-bit Port (Misaligned)[3]	1	0	1	1	0	OP0	(OP0)
Word to 16-bit Port (Aligned)	1	0	0	0	X	OP0	OP1
Word to 16-bit Port (Misaligned)[3]	1	0	1	0	X	(OP0)	OP0
3 Byte to 8-bit Port (Aligned)[2]	1	1	0	1	0	OP0	(OP1)
3 Byte to 8-bit Port (Misaligned)[2, 3]	1	1	1	1	0	OP0	(OP0)
3 Byte to 16-bit Port (Aligned)[3]	1	1	0	0	X	OP0	OP1
3 Byte to 16-bit Port (Misaligned)[2, 3]	1	1	1	0	X	(OP0)	OP0
Long Word to 8-bit Port (Aligned)	0	0	0	1	0	OP0	(OP1)
Long Word to 8-bit Port (Misaligned)[3]	0	0	1	1	0	OP0	(OP0)
Long Word to 16-bit Port (Aligned)	0	0	0	0	X	OP0	OP1
Long Word to 16-bit Port (Misaligned)[3]	0	0	1	0	X	(OP0)	OP0

NOTES:
1. Operands in parentheses are ignored by the CPU32 during read cycles.
2. Three-byte transfer cases occur only as a result of a long word to byte transfer.
3. The CPU32 does not support misaligned word and long-word transfers.

4.6 General-Purpose Input/Output

The SCIM contains six general-purpose input/output ports: ports A, B, E, F, G, and H. (Port C, an output-only port, is included under the discussion of chip selects.) Ports A, B, and G are available in single-chip mode only, and port H is available in single-chip or 8-bit expanded modes only. Ports E, F, G, and H have an associated data direction register (DDR) to configure each pin as input or output. Ports A and B share a DDR that configures each port as input or output. Ports E and F have associated pin assignment registers which configure each pin as digital I/O or an alternate function. Port F has an edge-detect flag register which indicates whether a transition has occurred on any of its pins.

The following table shows the shared functions of the general-purpose I/O ports and the modes in which they are available.

General-Purpose I/O Ports

Port	Shared Function	Modes
A	ADDR[18:11]	Single chip
B	ADDR[10:3]	Single chip
E	Bus Control	All
F	$\overline{\text{IRQ}}$[7:1]/MODCLK	All
G	DATA[15:8]	Single chip
H	DATA[7:0]	Single chip, 8-bit expanded

Access to Port A, B, E, G, and H data and data direction registers and Port E pin assignment register requires three clock cycles, to ensure timing compatibility with external port replacement logic. Port registers are byte-addressable and are grouped to allow coherent word access to port data register pairs A-B and G-H, as well as word-aligned long word coherency of A-B-G-H port data registers. Port registers are not affected by CPU reset.

If emulator mode is enabled, accesses to ports A, B, E, G, and H data and data direction registers and port E pin assignment register are mapped externally. Port F registers remain accessible.

A write to port A, B, E, F, G, or H data register is stored in the internal data latch, and if any port pin is configured as an output, the value stored for that bit is driven on the pin. A read of the port data register returns the value at the pin only if the pin is configured as a discrete input. Otherwise, the value read is the value stored in the register.

4.6.1 Ports A and B

Ports A and B are available in single-chip mode only. One data direction register controls data direction for both ports. Port A and B registers can be read or written at any time the MCU is not in emulator mode.

PORTA — Port A Data Register **$YFFA0A**

7	6	5	4	3	2	1	0
PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
RESET:							
U	U	U	U	U	U	U	U

PORTB — Port B Data Register **$YFFA0B**

7	6	5	4	3	2	1	0
PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
RESET:							
U	U	U	U	U	U	U	U

DDRAB — Port A/B Data Direction Register **$YFFA14**

7	6	5	4	3	2	1	0
0	0	0	0	0	0	DDA	DDB
RESET:							
0	0	0	0	0	0	0	0

DDA and DDB control the direction of the pin drivers for ports A and B, respectively, when the pins are configured for I/O. Setting DDA or DDB configures all pins in the corresponding port as outputs. Clearing DDA or DDB to zero configures all pins in the corresponding port as inputs.

4.6.2 Port E

Port E can be made available in all operating modes. The state of BERR and DATA8 during reset controls whether the port E pins are used as bus control signals or discrete I/O lines.

If the MCU is in emulator mode, an access of the port E data, data direction, or pin assignment registers (PORTE, DDRE, PEPAR) is forced to go external. This allows port replacement logic to be built externally, giving an emulator access to the bus control signals.

PORTE — Port E Data Register **$YFFA11, $YFFA13**

7	6	5	4	3	2	1	0
PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
RESET:							
U	U	U	U	U	U	U	U

PORTE is a single register that can be accessed in two locations. It can be read or written at any time the MCU is not in emulator mode.

DDRE — Port E Data Direction Register **$YFFA15**

7	6	5	4	3	2	1	0
DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0
RESET:							
0	0	0	0	0	0	0	0

The bits in this register control the direction of the pin drivers when the pins are configured as I/O. Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input. This register can be read or written at any time the MCU is not in emulator mode.

PEPAR — Port E Pin Assignment Register **$YFFA17**

7	6	5	4	3	2	1	0
PEPA7	PEPA6	PEPA5	PEPA4	PEPA3	PEPA2	PEPA1	PEPA0
RESET (Expanded, Single chip):							
DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8
0	0	0	0	0	0	0	0

The bits in this register control the function of each port E pin. Any bit set to one defines the corresponding pin to be a bus control signal, with the function shown in the following table. Any bit cleared to zero defines the corresponding pin to be an I/O pin, controlled by PORTE and DDRE.

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	$\overline{DSACK0}$

$\overline{BERR}$ and DATA8 control the state of this register following reset. If $\overline{BERR}$ or DATA8 are low during reset, this register is set to $00, defining all port E pins to be I/O pins. If $\overline{BERR}$ and DATA8 are both high during reset, the register is set to $FF, which defines all port E pins to be bus control signals.

4.6.3 Port F

Port F pins can be configured as interrupt-request inputs, edge-detect inputs or outputs, or discrete inputs or outputs. When port F pins are configured for edge detection, and a priority level greater than zero is specified in the port F edge-detect interrupt level register (PFLVR), port F control logic generates an interrupt request when the specified edge is detected. Interrupt vector assignment is made by writing a value to the port F edge-detect interrupt vector register (PFIVR). The edge-detect interrupt has the lowest arbitration priority in the SCIM. The state of the $\overline{BERR}$ and DATA9 pins determines the function of port F pins out of reset.

PORTF — Port F Data Register **$YFFA19, $YFFA1B**

7	6	5	4	3	2	1	0
PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
RESET: U	U	U	U	U	U	U	U

A write to the port F data register is stored in the internal data latch, and if any port F pin is configured as an output, the value stored for that bit is driven onto the pin. A read of PORTF returns the value at the pin only if the pin is configured as a discrete input. Otherwise, the value read is the value stored in the register.

PORTF is a single register that can be accessed in two locations. It can be read or written at any time, including when the MCU is in emulator mode.

PORTFE — Port F Edge-Detect-Flag Register **$YFFA29**

7	6	5	4	3	2	1	0
EF7	EF6	EF5	EF4	EF3	EF2	EF1	EF0
RESET:							
0	0	0	0	0	0	0	0

PORTFE contains eight flags that indicate whether a rising or falling edge has been detected on the corresponding port F input or output pin. When the specified edge (rising or falling) is detected on a port F pin configured in PFPAR for edge detection, the corresponding edge-detect flag in PORTFE is set. If a port F pin is configured as an interrupt-request pin or an I/O pin without edge detection, the flag for that pin is not set on a transition.

If the PFLVR is programmed to a value greater than zero, an interrupt is generated whenever a flag in PORTFE is set. The interrupt service routine can read PORTFE to determine which pin transition caused the interrupt. To clear PORTFE flags, read PORTFE and then write a zero value to the register.

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

7	6	5	4	3	2	1	0
DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0
RESET:							
0	0	0	0	0	0	0	0

The bits in this register control the direction of the pin drivers when the pins are configured for I/O. Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input.

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

7	6	5	4	3	2	1	0
PFPA[7:6]		PFPA[5:4]		PFPA[3:2]		PFPA[1:0]	
RESET (Expanded, Single chip):							
DATA9	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9	DATA9
0	0	0	0	0	0	0	0

The fields in this register determine the functions of pairs of port F pins as shown in the following tables. $\overline{\text{BERR}}$ and DATA9 determine the reset state of this register. If $\overline{\text{BERR}}$ or DATA9 are low during reset, this register is set to $00, defining all port F pins to be I/O pins. If $\overline{\text{BERR}}$ and DATA9 are both high during reset, the register is set to $FF, which defines all port F pins except PF0 to be interrupt signals.

Port F Pin Assignments

PFPAR Field	Port F Signal	Alternate Signal
PFPA3	PF[7:6]	$\overline{\text{IRQ}}$[7:6]
PFPA2	PF[5:4]	$\overline{\text{IRQ}}$[5:4]
PFPA1	PF[3:2]	$\overline{\text{IRQ}}$[3:2]
PFPA0	PF[1:0]	$\overline{\text{IRQ}}$1, MODCLK*

*MODCLK is recognized only during reset

PFPAR Pin Functions

PFPA Bits	Port F Signal
00	I/O pin without edge detect
01	Rising edge detect input
10	Falling edge detect input
11	Interrupt request

PFIVR — Port F Edge-Detect Interrupt Vector Register **$YFFA2B**

7	6	5	4	3	2	1	0
PFIVR7	PFIVR6	PFIVR5	PFIVR4	PFIVR3	PFIVR2	PFIVR1	PFIVR0
RESET:							
0	0	0	0	1	1	1	1

This register determines which vector in the exception vector table is used for interrupts generated by the port F edge-detect logic. Program PFIVR[7:0] to point to the appropriate interrupt vector. Once this register has been initialized, PFIVR[7:0] are placed on the data bus during an interrupt-acknowledge cycle whenever port F wins interrupt arbitration. This register can be read or written at any time in supervisor mode.

PFLVR — Port F Edge-Detect Interrupt Level Register **$YFFA2D**

7	6	5	4	3	2	1	0
0	0	0	0	0	PFLV2	PFLV1	PFLV0
RESET:							
0	0	0	0	1	0	0	0

This 3-bit read/write register determines the priority level of the port F edge-detect interrupt. The reset value is $00, indicating that the interrupt is disabled. When several sources of interrupts from the SCIM are arbitrating for the same level, the port F edge-detect interrupt has the lowest arbitration priority.

4.6.4 Port G

Port G is available in single-chip mode only. In single-chip mode, these pins are always configured for general-purpose I/O.

PORTG — Port G Data Register **$YFFA0C**

7	6	5	4	3	2	1	0
PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
RESET:							
U	U	U	U	U	U	U	U

This register can be read or written any time the MCU is not in emulator mode.

DDRG — Port G Data Direction Register **$YFFA0E**

7	6	5	4	3	2	1	0
DDG7	DDG6	DDG5	DDG4	DDG3	DDG2	DDG1	DDG0
RESET:							
0	0	0	0	0	0	0	0

The bits in this register control the direction of the pin drivers when the pins are configured as I/O. Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input.

4.6.5 Port H

Port H is available in single-chip and 8-bit expanded modes only. The function of these pins is determined by operating mode — there is no pin assignment register associated with this port.

PORTH — Port H Data Register **$YFFA0D**

7	6	5	4	3	2	1	0
PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
RESET:							
U	U	U	U	U	U	U	U

This register can be read or written any time the MCU is not in emulator mode. Reset has no effect.

DDRH — Port H Data Direction Register **$YFFA0F**

7	6	5	4	3	2	1	0
DDH7	DDH6	DDH5	DDH4	DDH3	DDH2	DDH1	DDH0
RESET:							
0	0	0	0	0	0	0	0

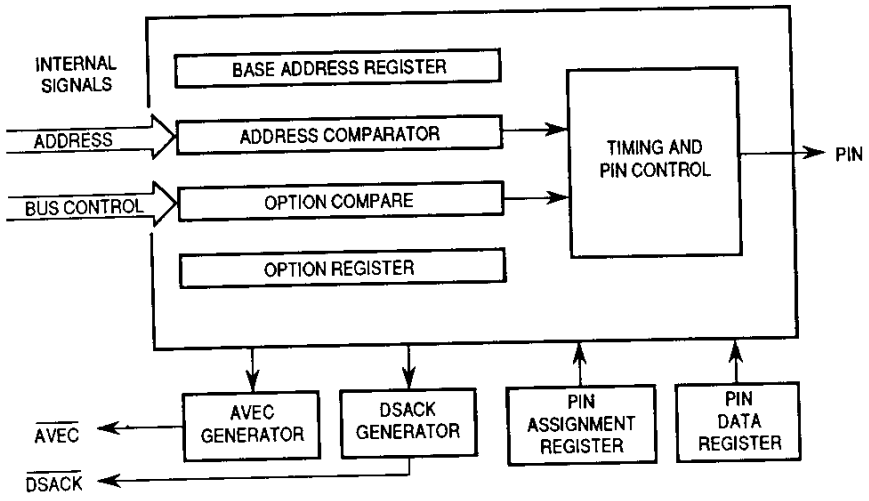
The bits in this register control the direction of the pin drivers when the pins are configured as I/O. Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input.

4.7 Chip Selects

Typical microcontrollers require additional hardware to provide external chip-select signals. The MC68F333 includes nine programmable chip-select circuits that can provide up to 13 wait states during access to external memory and peripherals. An additional chip select ($\overline{CSE}$) provides emulator support. Block sizes of 2 Kbytes to 1 Mbyte can be selected.

Assertion of chip selects can be synchronized with bus control signals to provide output enable, read/write strobes, or interrupt-acknowledge signals. Logic can also generate $\overline{DSACK}$ signals internally. A single $\overline{DSACK}$ generator is shared by all circuits — multiple chip selects assigned to the same address and control must have the same number of wait states. Chip selects can also be synchronized with the ECLK signal available on ADDR23.

When a memory access occurs, chip-select logic compares address space type, address, type of access, transfer size, and interrupt priority (in the case of interrupt acknowledge) to parameters stored in chip-select registers. If all parameters match, the appropriate chip-select signal is asserted. Select signals are active low. A block diagram of a single chip-select circuit is shown below.



Chip-Select Circuit Block Diagram

Each chip-select pin can have two or more functions. Chip-select configuration out of reset is determined by operating mode. In all modes, the boot ROM select signal is automatically asserted out of reset. In single-chip mode, all chip-select pins except $\overline{CS10}$ and $\overline{CS0}$ are configured for alternate functions or discrete output. In expanded modes, appropriate pins are configured for chip-select operation, but chip-select signals cannot be asserted until a transfer size is chosen. In fully expanded mode, data bus pins can be held low to enable alternate functions for chip-select pins.

The following table lists allocation of chip selects and port C discrete outputs on the pins of the MCU.

Chip Select Pin Allocation

Pin	Chip Select	Discrete Outputs
$\overline{\text{CSBOOT}}$	$\overline{\text{CSBOOT}}$	—
$\overline{\text{BR}}$	$\overline{\text{CS0}}$	—
$\overline{\text{BG}}$	$\overline{\text{CSM}}$	—
$\overline{\text{BGACK}}$	$\overline{\text{CSE}}$	—
FC0	$\overline{\text{CS3}}$	PC0
FC1	—	PC1
FC2	$\overline{\text{CS5}}$	PC2
ADDR19	$\overline{\text{CS6}}$	PC3
ADDR20	$\overline{\text{CS7}}$	PC4
ADDR21	$\overline{\text{CS8}}$	PC5
ADDR22	$\overline{\text{CS9}}$	PC6
ADDR23	$\overline{\text{CS10}}$	ECLK

Emulation mode chip select signals are used during external register emulation. Pin function is controlled by a chip select pin assignment register, but the other chip select registers do not affect these signals.

During emulator mode operation, all port A, B, E, G, and H data and data direction registers, and the port E pin assignment register are mapped externally. The emulator chip select signal ($\overline{\text{CSE}}$) is asserted whenever one of these registers is addressed. The SCIM does not respond to these accesses; an external device, such as a port replacement unit, can respond instead. Accesses to externally-mapped registers take three clock cycles.

The internal module chip select ($\overline{\text{CSM}}$) signal is not supported by the MC68F333. Refer to **4.2.3 Emulation Support** for additional information on the $\overline{\text{CSE}}$ signal.

4.7.1 Chip-Select Registers

Each chip-select pin can have one or more functions. Pin assignment registers (CSPAR) determine the functions of chip-select pins. Pin assignment registers also determine port size (8- or 16-bit) for dynamic bus allocation. A pin data register (CSPDR) latches data for chip-select pins that are used for discrete output.

Blocks of addresses are assigned to each chip-select function. Block sizes of 2 Kbytes to 1 Mbyte can be selected by writing values to the appropriate base address register (CSBAR). Address blocks for separate chip-select functions can overlap.

Chip-select option registers (CSOR) determine timing of and conditions for assertion of chip-select signals. Eight parameters, including operating mode, access size, synchronization, and wait state insertion can be specified.

Initialization code often resides in a peripheral memory device controlled by the chip-select circuits. Special chip-select functions and registers (CSORBT, CSBARBT) are provided to support bootstrap operation.

4.7.2 Pin Assignment Registers

The pin assignment registers contain pairs of bits that determine the function of pins in other chip-select registers. Alternate functions of the associated pins are shown in parentheses. The reset value depends on the operating mode.

In the following register diagrams, the notation "DATA#" indicates that a bit goes to the logic level of that data bus pin on reset. DATA lines have weak pull-ups: during reset in fully-expanded mode, an active external device can pull the data lines low to select alternate functions.

CSPAR0 — Chip-Select Pin Assignment Register 0 **$YFFA44**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	CS5 (FC2)		0 (FC1)		CS3 (FC0)		CSE (BGACK)		CSM (BG)		CS0 (BR)		CSBOOT	
RESET (partially expanded, fully expanded, single chip):															
0	0	1	0	0	1	1	0	0	1	0	1	1	0	1	0
0	0	DATA2	1	0	1	DATA2	1	DATA10	1	DATA10	1	DATA2	1	1	DATA0
0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1

CSPAR0[15:14] — Not used.
These bits always read zero; write has no effect.

CSPAR0[11] — Not used.
Bit 10 determines whether signal is FC1 or a discrete output.

CSPAR1 — Chip-Select Pin Assignment Register 1 **$YFFA46**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CS10 (ADDR23)		CS9 (ADDR22)		CS8 (ADDR21)		CS7 (ADDR20)		CS6 (ADDR19)	
RESET (partially expanded, fully expanded, single chip):															
0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0
0	0	0	0	0	0	DATA7	1	DATA6	1	DATA5	1	DATA4	1	DATA3	1
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

CSPAR1[15:10] — Not used.
These bits always read zero; write has no effect. Clearing both CS10 select bits (CSPAR1[9:8]) enables the M6800 bus clock (ECLK) on ADDR23.

The table below shows pin assignment register encoding.

Bit Pair	Description
00	Discrete Output
01	Alternate Function
10	Chip Select (8-Bit Port)
11	Chip Select (16-Bit Port)

A pin programmed as a discrete output drives an external signal to the value specified in the pin data register, with the following exceptions:

a. No discrete output function is available on pins $\overline{BR}$, $\overline{BG}$, or $\overline{BGACK}$.
b. ADDR23 provides ECLK output rather than a discrete output signal.

When a pin is programmed for discrete output, internal chip-select logic is inhibited, and $\overline{DSACK}$ or $\overline{AVEC}$ signals are never generated.

Port size is determined when a pin is assigned as a chip select. When a pin is assigned to an 8-bit port, the chip select is asserted at all addresses within the block range. If a pin is assigned to a 16-bit port, the upper/lower byte field of the option register selects the byte with which the chip select is associated.

4.7.3 Base-Address Registers

A base address is the starting address for the block enabled by a given chip select. Block size determines the extent of the block above the base address. Each chip select has an associated base register, so that an efficient address map can be constructed for each application.

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		
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

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

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

BLKSZ — Block Size Field

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

Block Size Field	Block Size	Address Lines Compared
000	2 K	ADDR[23:11]
001	8 K	ADDR[23:13]
010	16 K	ADDR[23:14]
011	64 K	ADDR[23:16]
100	128 K	ADDR[23:17]
101	256 K	ADDR[23:18]
110	512 K	ADDR[23:19]
111	1 M	ADDR[23:20]

ADDR[15:3] — Base-Address Field

This field sets the starting address of a particular address space. The address-compare logic uses only the most significant bits to match an address within a block — the value of the base address must be a multiple of block size. Base address register diagrams show how base register bits correspond to address lines.

4.7.4 Option Registers

The option registers contain eight fields that determine timing of chip-select signals and conditions for their assertion. These make the chip selects useful for generating peripheral control signals. Certain constraints set by fields in the base address register and in the option register must be satisfied in order to assert a chip-select signal and to provide $\overline{\text{DSACK}}$ or autovector support.

CSORBT — Chip-Select Option Register Boot ROM **$YFFA4A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE	BYTE		R/W		STRB	DSACK				SPACE		IPL			AVEC
RESET:															
0	1	1	1	1	0	1	1	0	1	1	1	0	0	0	0

CSOR10 – CSOR0 — Chip-Select Option Registers **$YFFA4E–$YFFA76**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE	BYTE		R/W		STRB	DSACK				SPACE		IPL			AVEC
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CSORBT, the option register for $\overline{\text{CSBOOT}}$, contains special reset values that support bootstrap operations from peripheral memory devices.

The following bit descriptions apply to both CSORBT and CSOR10–CSOR0 option registers.

MODE — Asynchronous/Synchronous Mode

0 = Asynchronous mode selected

1 = Synchronous mode selected

In asynchronous mode, the chip select is asserted synchronized with $\overline{\text{AS}}$ or $\overline{\text{DS}}$. In synchronous mode, the DSACK field is not used, because a bus cycle is only performed as a synchronous operation. When a match condition occurs on a chip select programmed for synchronous operation, the chip select signals the EBI that an ECLK cycle is pending.

BYTE — Upper/Lower Byte Option

This field is used only when the chip-select 16-bit port option is selected in the pin assignment register. The following table lists upper/lower byte options.

Byte	Description
00	Disable
01	Lower Byte
10	Upper Byte
11	Both Bytes

R/W — 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. The table below shows the options.

R/W	Description
00	Reserved
01	Read Only
10	Write Only
11	Read/Write

STRB — Address Strobe/Data Strobe

0 = Address strobe

1 = Data strobe

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

DSACK — Data Strobe Acknowledge

This field specifies the source of $\overline{\text{DSACK}}$ in asynchronous mode. It also allows the user to adjust bus timing with internal $\overline{\text{DSACK}}$ generation by controlling the number of wait states that are inserted to optimize bus speed in a particular application. The following table shows the DSACK field encoding. A no-wait encoding (%0000) corresponds to a 3 clock-cycle bus. The fast termination encoding (%1110) corresponds to a 2 clock-cycle bus (also referred to as –1 wait states). Microcontroller modules typically respond at this rate, but fast termination can also be used to access fast external memory.

DSACK	Description
0000	No Wait States
0001	1 Wait State
0010	2 Wait States
0011	3 Wait States
0100	4 Wait States
0101	5 Wait States
0110	6 Wait States
0111	7 Wait States
1000	8 Wait States
1001	9 Wait States
1010	10 Wait States
1011	11 Wait States
1100	12 Wait States
1101	13 Wait States
1110	Fast Termination
1111	External $\overline{\text{DSACK}}$

SPACE — Address Space

This option field is used to select an address space to be used by the chip-select logic.

Space Field	Address Space
00	CPU Space
01	User Space
10	Supervisor Space
11	Supervisor/User Space

IPL — Interrupt Priority Level

When the SPACE field is set for CPU space (%00), chip-select logic can be used for interrupt acknowledge. During an interrupt-acknowledge cycle, the priority level on address lines ADDR[3:1] is compared to the value in the IPL field. If the values are the same, then a chip select can be asserted, provided other option register conditions are met. When the SPACE field has any value except %00, the IPL field determines whether an access takes place in program or data space. The following table shows IPL field encoding.

IPL	Space = 00	Space = 01, 10, 11
000	All	Data or Program
001	IPL1	Data
010	IPL2	Program
011	IPL3	Reserved
100	IPL4	Reserved
101	IPL5	Data
110	IPL6	Program
111	IPL7	Reserved

This field only affects the response of chip selects and does not affect interrupt recognition by the CPU. "All" means that a chip select signal is asserted regardless of the priority of the interrupt.

AVEC — Autovector Enable

0 = External interrupt vector enabled

1 = Autovector enabled

This field selects one of two methods of acquiring the interrupt vector during the interrupt acknowledge cycle. It is not generally used in conjunction with a chip-select pin.

If the chip select is configured to trigger on an interrupt-acknowledge cycle (SPACE = 00) and the AVEC field is set to one, the chip select automatically generates an $\overline{\text{AVEC}}$ signal in response to the interrupt-acknowledge cycle. Otherwise, the vector must be supplied by the requesting device.

CSPDR — Chip-Select Pin Data Register **$YFFA41**

7	6	5	4	3	2	1	0
0	DO6	DO5	DO4	DO3	DO2	DO1	DO0
RESET:							
0	0	0	0	0	0	0	0

The CSPDR controls the state of pins programmed as port C discrete outputs. When a pin is assigned as a discrete output, the value in this register appears at the output. DO[6:0] correspond to CS[9:3]. This is a read/write register. Bit 7 is not used. Writing to this bit has no effect, and it always reads zero.

4.7.5 Chip-Select Reset Operation

The reset values of the CS[10:0] pin assignment fields in CSPAR0 and CSPAR1 depend on the operating mode selected. Refer to **4.2.2 Operating Modes** and to the discussion of these registers for more information.

The CSBOOT assignment field in CSPAR0 is configured differently from the other chip-select assignment fields. The MSB, bit 1 of CSPAR0, is always one. This enables the $\overline{\text{CSBOOT}}$ signal to select a boot ROM containing initialization firmware. The LSB value, determined by the logic level of DATA0 during reset, selects boot ROM port size. When DATA0 is held low, port size is 8 bits. When internal connections pull the LSB high, port size is 16 bits.

After reset, the MCU fetches initialization vectors from addresses $0000 to $0006 in bank 0 of program space. To support bootstrap operation from reset, the base address field in chip-select base register boot (CSBARBT) has a reset value of all zeros. A ROM device containing vectors located at these addresses can be enabled by $\overline{\text{CSBOOT}}$ after a reset. The block size field in CSBARBT has a reset value of 512 Kbytes.

The BYTE field in option register CSORBT has a reset value of both bytes, but the BYTE fields in CSOR10-CSOR0 have reset values of disable, since they should not select external devices until an initial program sets up the base and option registers. The following table shows the reset values in the base and option registers for $\overline{\text{CSBOOT}}$.

Chip Select Reset Values

Field	Reset Value
Base Address	$0000 0000
Block Size	512 Kbyte
Async/Sync Mode	Asynchronous Mode
Upper/Lower Byte	Both Bytes
Read/Write	Read/Write
AS/DS	AS
DSACK	13 Wait States
Address Space	Supervisor/User
IPL	All
Autovector	External Interrupt Vector

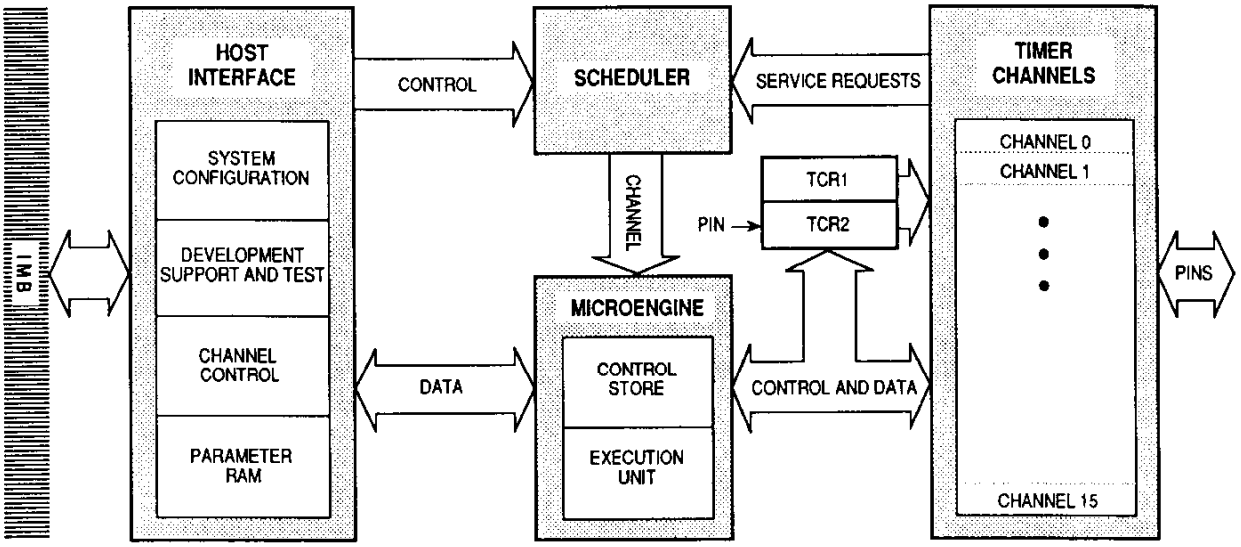
4.8 Factory Test Block

The test submodule supports scan-based testing of the various MCU modules. Test submodule registers are intended for Motorola use. Register names and addresses are provided to show the user that these addresses are occupied.

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

5 Time Processor Unit (TPU)

The TPU provides optimum performance in controlling time-related activity. The TPU contains a dedicated execution unit, a tri-level prioritized scheduler, data storage RAM, dual-time bases, and microcode ROM. The TPU controls 16 independent, orthogonal channels, each with an associated I/O pin, and is capable of performing any microcoded time function. Each channel also contains a dedicated event register, allowing both match and input capture functions.



TPU Block Diagram

5.1 TPU ROM Functions

The following microcoded time functions are implemented in the TPU control store ROM.

5.1.1 Discrete Input/Output (DIO)

When a pin is used as a discrete input, a parameter indicates the current input level and the previous 15 levels of a pin. Bit 15, the most significant bit of the parameter, indicates the most recent state. Bit 14 indicates the next most recent state, and so on.

5.1.2 Input Capture/Input Transition Counter (ITC)

Any channel of the TPU can capture the value of a specified timer count register (TCR1 or TCR2) upon the occurrence of each transition, and then generate an interrupt request to notify the bus master.

5.1.3 Output Compare (OC)

The OC function generates a rising edge, a falling edge, or a toggle of the previous edge in either of two ways:

a. OC generates the specified edge at a programmable delay time for a user-specified time. The CPU can also force an immediate output, thereby generating a pulse with a length of the programmable delay time.
b. Upon receiving a link from a channel, OC references, without CPU interaction, a specifiable period or parameter of another channel and adds an offset to that parameter.

5.1.4 Pulse-Width Modulation (PWM)

The PWM function generates a pulse-width modulated waveform with any duty cycle from 0% to 100% (within the resolution and latency capability of the TPU). The CPU provides one parameter that specifies waveform period and another parameter that specifes waveform high time. Updates to one or both of these parameters can change the output to take effect either immediately, or coherently, at the next low-to-high transition of the pin.

5.1.5 Synchronized Pulse-Width Modulation (SPWM)

The SPWM function generates a pulse-width modulated waveform. The CPU can change period or high time at any time. When synchronized to a time function on a second channel, the SPWM low-to-high transitions have a time relationship to transitions on the second channel.

5.1.6 Period Measurement with Additional Transition Detect (PMA)

The PMA function allows special-purpose 16-bit period measurement. It detects the occurrence of an additional transition indicated by the current measurement being less than a programmed ratio of a previous measurement. When detected, this condition can be counted and compared to a programmed number of additional transitions.

5.1.7 Period Measurement with Missing Transition Detect (PMM)

The PMM function allows special-purpose 16-bit period measurement. It detects the occurrence of a missing transition indicated by the current measurement being more than a previous measurement multiplied by a programmed ratio. When detected, this condition can be counted and compared to a programmed number of transitions.

5.1.8 Position-Synchronized Pulse Generator (PSP)

Any channel of the TPU can generate an output transition or pulse, which is a projection in time based on a reference period previously calculated on another channel.

5.1.9 Stepper Motor (SM)

The stepper motor control algorithm uses a programmable number of step rates to control the linear acceleration and deceleration of a stepper motor. Any group of up to eight channels can be programmed to generate the control logic necessary to drive a stepper motor. Nominally, only two or four channels are used for a two-phase motor.

5.1.10 Period/Pulse-Width Accumulator (PPWA)

The PPWA function continuously accumulates high time or total elapsed interval of a waveform over a programmed number of input periods. It continuously tracks current and most recent accumulated times.

5.2 TPU Register Map

Access	Address	15 Byte N 8	7 Byte N+1 0
S	$YFFE00	TPU Module Configuration Register (TMCR)	
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)	

S = Supervisor access only
S/U = Supervisor access if SUPV = 1, or unrestricted if SUPV = 0
Y = M111, where M is the state of the MODMAP bit in the SCIMCR (Y = $7 or $F)

5.3 Parameter RAM Map

Channel	Parameter							
	0	1	2	3	4	5	6	7
0	$YFFF00	02	04	06	08	0A	—	—
1	$YFFF10	12	14	16	18	1A	—	—
2	$YFFF20	22	24	26	28	2A	—	—
3	$YFFF30	32	34	36	38	3A	—	—
4	$YFFF40	42	44	46	48	4A	—	—
5	$YFFF50	52	54	56	58	5A	—	—
6	$YFFF60	62	64	66	68	6A	—	—
7	$YFFF70	72	74	76	78	7A	—	—
8	$YFFF80	82	84	86	88	8A	—	—
9	$YFFF90	92	94	96	98	9A	—	—
10	$YFFFA0	A2	A4	A6	A8	AA	—	—
11	$YFFFB0	B2	B4	B6	B8	BA	—	—
12	$YFFFC0	C2	C4	C6	C8	CA	—	—
13	$YFFFD0	D2	D4	D6	D8	DA	—	—
14	$YFFFE0	E2	E4	E6	E8	EA	EC	EE
15	$YFFFF0	F2	F4	F6	F8	FA	FC	FE

— = Not Implemented

Y = M111, where M is the MODMAP bit in the SCIMCR (Y = $7 or $F).

All parameter RAM addresses are located in supervisor or unrestricted space, depending on state of SUPV bit.

5.4 TPU Registers

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

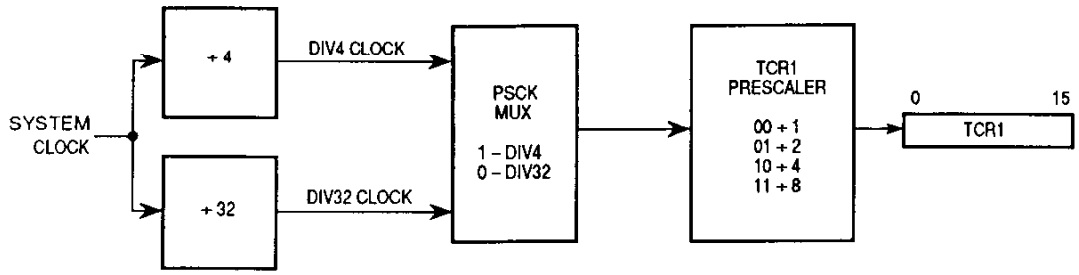
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	TCR1P		TCR2P		EMU	T2CG	STF	SUPV	PSCK	0	0	IARB			
RESET:															
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

STOP — Stop Bit

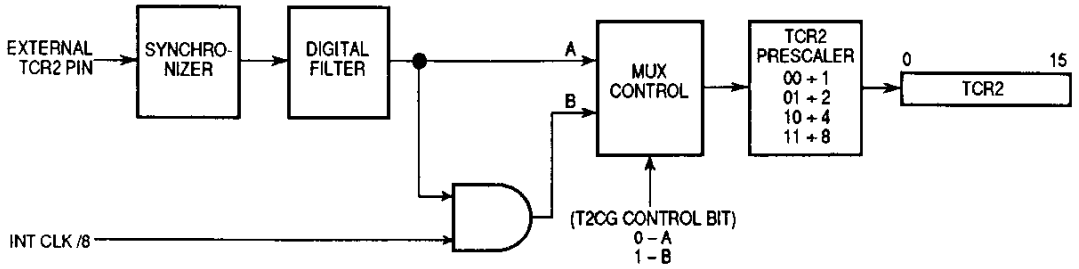
0 = Internal clocks not shut down (reset condition)

1 = Internal clocks shut down

TCR1P — TCR1 Prescaler Control



TCR2P — TCR2 Prescaler Control



EMU — Emulation Control

0 = TPU and RAM not in emulation mode
1 = TPU and RAM in emulation mode

T2CG — TCR2 Clock/Gate Control

0 = TCR2 pin used as clock source for TCR2
1 = TCR2 pin used as gate of DIV8 clock for TCR2

STF — Stop Flag

0 = TPU operating
1 = TPU stopped

SUPV — Supervisor Data Space

0 = Assignable registers are unrestricted (FC2 is ignored)
1 = Assignable registers are restricted (FC2 is decoded)

PSCK — Prescaler Clock

0 = DIV32 (system clock/32) is input to TCR1 prescaler
1 = DIV4 (system clock/4) is input to TCR1 prescaler

IARB — Interrupt Arbitration ID Bits

This field contains the arbitration number of the TPU that is used to arbitrate for the intermodule bus when two or more modules or peripherals have an interrupt on the same priority level.

TCR — Test Configuration Register **$YFFE02**

The TCR is used for factory test of the MCU.

DSCR — Development Support Control Register **$YFFE04**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HOT4	0	0	0	0	BLC	CLKS	FRZ1	FRZ0	CCL	BP	BC	BH	BL	BM	BT
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

HOT4 — Hang on T4

0 = Exit wait on T4 state caused by assertion of HOT4

1 = Enter wait on T4 state

Bits [14:11] — Not Implemented

BLC — Branch Latch Control

0 = Latch conditions into branch condition register prior to exiting halted state.

1 = Do not latch conditions into branch condition register before exiting the halted state or during the time-slot transition period.

CLKS — Stop Clocks (to TCRs)

0 = Do not stop TCRs.

1 = Stop TCRs during the halted state.

FRZ[1:0] — IMB FREEZE Response

The FRZ bits specify the TPU microengine response to the FREEZE signal.

FRZ[1:0]	TPU Response
00	Ignore Freeze
01	Reserved
10	Freeze at End of Current Microcycle
11	Freeze at Next Time-Slot Boundary

CCL — Channel Conditions Latch

CCL controls the latching of channel conditions (MRL and TDL) when the CHAN register is written.

0 = Only the pin state condition of the new channel is latched as a result of the write CHAN register microinstruction.

1 = Pin state, MRL, and TDL conditions of the new channel are latched as a result of a write CHAN register microinstruction.

BP, BC, BH, BL, BM, and BT — Breakpoint Enable Bits

DSCR[5:0] are TPU breakpoint enables. Setting a bit enables a breakpoint condition.

BP — Break if µPC equals µPC breakpoint register.

BC — Break if CHAN register equals channel breakpoint register at beginning of state or when CHAN is changed through microcode.

BH — Break if host service latch is asserted at beginning of state.

BL — Break if link service latch is asserted at beginning of state.

BM — Break if MRL is asserted at beginning of state.

BT — Break if TDL is asserted at beginning of state.

DSSR — Development Support Status Register **$YFFE06**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	BKPT	PCBK	CHBK	SRBK	TPUF	0	0	0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits [15:8], [2:0] — Not Implemented

BKPT — Breakpoint Asserted Flag

If an internal breakpoint caused the TPU to enter the halted state, the TPU asserts the BKPT signal on the IMB and the BKPT flag. The TPU continues to assert BKPT until it recognizes a breakpoint acknowledge cycle from a host, or until the FREEZE signal on the IMB is asserted.

PCBK — µPC Breakpoint Flag

PCBK is asserted if a breakpoint occurs because of a µPC register match with the µPC breakpoint register. PCBK is negated when the BKPT flag is negated.

CHBK — Channel Register Breakpoint Flag

CHBK is asserted if a breakpoint occurs because of a CHAN register match with the channel register breakpoint register. CHBK is negated when the BKPT flag is negated.

SRBK — Service Request Breakpoint Flag

SRBK is asserted if a breakpoint occurs because of any of the service request latches being asserted along with their corresponding enable flag in the development support control register. SRBK is negated when the BKPT flag is negated.

TPUF — TPU FREEZE Flag

TPUF is asserted whenever the TPU is in a halted state as a result of FREEZE being asserted. This flag is automatically negated when the TPU exits the halted state because of FREEZE being negated.

TICR — TPU Interrupt Configuration Register **$YFFE08**

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

TICR[15:11] — Not Implemented

CIRL — Channel Interrupt Request Level

The interrupt request level for all channels is specified by this three-bit encoded field. Level seven for this field indicates a nonmaskable interrupt; level zero indicates that all channel interrupts are disabled.

CIBV — Channel Interrupt Base Vector

This field specifies the most significant nibble of all 16 TPU channel interrupt vector numbers.

TICR[3:0] — Not Implemented

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

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

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

CFSR0–CFSR3 — Channel Function Select Registers **$YFFE0C–$YFFE12**

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

CFSR[15:0] — Encoded One of 16 Time Functions for Each Channel

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

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

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

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

CH[15:0] — Encoded Host Sequence

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

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH7		CH6		CH5		CH4		CH3		CH2		CH1		CH0	
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

CHX[1:0]	Service
00	No Host Service (Reset Condition)
01	Type 1 Host Service
10	Type 2 Host Service
11	Type 3 Host Service

Function Name	Function Code	Host Service Request Code	Host Sequence Code*
DIO Discrete Input/Output	$8	1= Force Output High	0 = Trans Mode — Record Pin on Transition
		2 = Force Output Low	0 = Trans Mode — Record Pin on Transition
		3 = Initialization, Input Specified	0 = Trans Mode — Record Pin on Transition
		3 = Initialization, Periodic Input	1 = Match Mode — Record Pin at Match_Rate
		3 = Update Pin Status Parameter	2 = Record Pin State on HSR 11
ITC Input Capture/ Input Transition Counter	$A	0 = None 1 = Initialization 2 = (Not Implemented) 3 = (Not Implemented)	0 = No Link, Single Mode 1 = No Link, Continuous Mode 2 = Link, Single Mode 3 = Link, Continuous Mode
OC Output Compare	$E	0 = None 1 = Host-Initiated Pulse Mode 2 = (Not Implemented) 3 = Continuous Pulse Mode	0 = Execute All Functions 1 = Execute All Functions 2 = Only Update TCRn Parameters 3 = Only Update TCRn Parameters
PWM Pulse-Width Modulation	$9	0 = None 1 = Immediate Update Request 2 = Initialization 3 = (Not Implemented)	(None Implemented)
SPWM Synchronized Pulse-Width Modulation	$7	0 = None 1 = (Not Implemented) 2 = Initialization 3 = Immediate Update Request	0 = Mode 0 1 = Mode 1 2 = Mode 2 3 =(Not Implemented)
PMA/PMM Period Measurement with Additional/Missing Transition Detect	$B	0 = None 1 = Initialization 2 = (Not Implemented) 3 = (Not Implemented)	0 = PMA Bank Mode 1 = PMA Count Mode 2 = PMM Bank Mode 3 = PMM Count Mode
PSP Position-Synchronized Pulse Generator	$C	0 = None 1 = Immediate Update Request 2 = Initialization 3 = Force Change	0 = Pulse Width Set by Angle 1 = Pulse Width Set by Time 2 = Pulse Width Set by Angle 3 = Pulse Width Set by Time
SM Stepper Motor	$D	0 = None 1 = None 2 = Initialization 3 = Step Request	(None Implemented)
PPWA Period/Pulse-Width Accumulator	$F	0 = None 1 = (Not Implemented) 2 = Initialization 3 = (Not Implemented)	0 = 24-Bit Period 1 = 16-Bit Period + Link 2 = 24-Bit Pulse Width 3 = 16-Bit Pulse Width + Link

*Host Sequence Code interpretation is determined by the function; some HSQ codes apply to all HSR codes, some to only one, such as Init.

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

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

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

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

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

CHX[1:0]	Service
00	Disabled
01	Low
10	Middle
11	High

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

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

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

LR — Link Register **$YFFE22**

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

CH[15:0] — Test Mode Link Service Request Enable Bit
0 = Link bit not asserted
1 = Link bit asserted

SGLR — Service Grant Latch Register **$YFFE24**

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

CH[15:0] — Service Granted Bits

DCNR — Decoded Channel Number Register **$YFFE26**

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

CH[15:0] — Service Status Bits

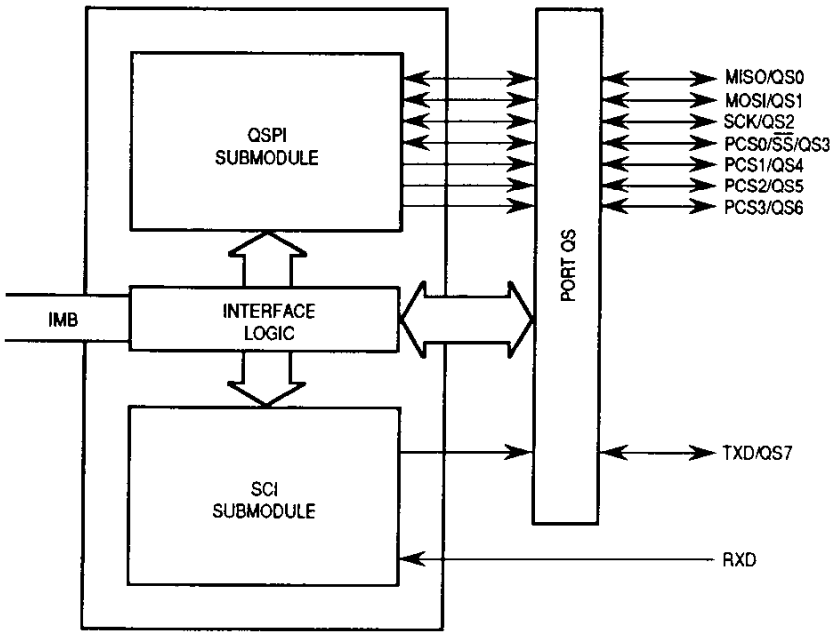
6 Queued Serial Module (QSM)

The QSM contains two serial interfaces: the queued serial peripheral interface (QSPI) and the serial communication interface (SCI).

The QSPI provides easy peripheral expansion or interprocessor communication via a full-duplex, synchronous, three-line bus: data in, data out, and a serial clock. Four programmable chip-select pins provide addressability for up to 16 peripheral devices. Two self-contained RAM queues allow up to 16 serial transfers of 8–16 bits each, or transmission or reception of a 256-bit data stream without CPU intervention. A special wraparound mode supports continuous sampling of a serial peripheral, with automatic QSPI RAM updating, for efficient interfacing to A/D converters.

The SCI provides a standard nonreturn to zero (NRZ) mark/space format. It operates in either full- or half-duplex mode — there are separate transmitter and receiver enable bits and dual data buffers. A modulus-type baud rate generator provides baud rates from 64 to 524 Kbaud (with a 16.78-MHz system clock). Word length of either 8 or 9 bits is software selectable. Optional parity generation and detection provide either even or odd parity check capability. Advanced error detection circuitry catches glitches of up to 1/16 of a bit time in duration. Wakeup functions allow the CPU to run uninterrupted until meaningful data is available.

The figure below is a block diagram of the QSM.



QSM Block Diagram

6.1 QSM Register Map

Access	Address	15 8	7 0
S	$YFFC00	QMCR	
S	$YFFC02	QTEST	
S	$YFFC04	QILR	QIVR
S/U	$YFFC06	RESERVED	
S/U	$YFFC08	SCCR0	
S/U	$YFFC0A	SCCR1	
S/U	$YFFC0C	SCSR	
S/U	$YFFC0E	SCDR	
S/U	$YFFC10	RESERVED	
S/U	$YFFC12	RESERVED	
S/U	$YFFC14	RESERVED	QPDR
S/U	$YFFC16	QPAR	QDDR
S/U	$YFFC18	SPCR0	
S/U	$YFFC1A	SPCR1	
S/U	$YFFC1C	SPCR2	
S/U	$YFFC1E	SPCR3	SPSR
S/U	$YFFC20–FF	RESERVED	
S/U	$YFFD00–1F	REC.RAM	
S/U	$YFFD20–3F	TRAN.RAM	
S/U	$YFFD40–4F	COMD.RAM	

S = Supervisor access only

S/U = Supervisor access only or unrestricted, depending on SUPV bit in QMCR

Y = M111, where M = MODMAP bit in SCIMCR

6.2 QSM Pin Functions

The table below summarizes QSM pin functions.

	Pin	Mode	QDDR Bit	Pin Function
QSPI Pins	MISO	Master	0	Serial Data Input to QSPI
			1	Output Value from QPDR
		Slave	0	Input Value to QPDR
			1	Serial Data Output from QSPI
	MOSI	Master	0	Input Value to QPDR
			1	Serial Data Output from QSPI
		Slave	0	Serial Data Input to QSPI
			1	Output Value from QPDR
	SCK	Master	0	Input Value to QPDR
			1	Clock Output to QSPI
		Slave	0	Clock Input to QSPI
			1	Output Value from QPDR
	PCS0/$\overline{SS}$	Master	0	Input (May Cause Mode Fault)
			1	Output Selects Peripherals
		Slave	0	Input Selects to QSPI
			1	Output Value from QPDR
	PCS[3:1]	Master	0	Input Value to QPDR
			1	Output Selects Peripherals
		Slave	0	Input Value to QPDR
			1	Output Value from QPDR
SCI Pins	TXD	Transmit	X	Serial Data Output from SCI (TE = 1)
	RXD	Receive	NA	Serial Data Input to SCI

X = QDDR bit ignored

6.3 QSM Registers

QSM registers are divided into four categories: QSM global registers, QSM pin control registers, QSPI submodule registers, and SCI submodule registers. The QSPI and SCI registers are defined in separate sections below. Writes to unimplemented register bits have no meaning or effect, and reads from unimplemented bits always return a logic zero value.

The MODMAP bit of the SCIM module configuration register (SCIMCR) defines the most significant bit (ADDR23) of the address, shown in each register figure as "Y" (Y = $7 or $F). This bit, concatenated with the rest of the address given, forms the absolute address of each register.

6.3.1 Global Registers

The QSM global registers contain system parameters used by both the QSPI and the SCI submodules. These registers define parameters used by the QSM to interface with the CPU and other system modules.

QMCR — QSM Configuration Register **$YFFC00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ1	FRZ0	0	0	0	0	0	SUPV	0	0	0	IARB			
RESET:															
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

The QMCR contains parameters for interfacing to the CPU and the intermodule bus (IMB).

STOP — Stop Enable

0 = Normal QSM clock operation

1 = QSM clock operation stopped

STOP places the QSM in a low-power state by disabling the system clock in most parts of the module. QMCR is the only register guaranteed to be readable while STOP is asserted. The QSPI RAM is not readable; however, writes to RAM or any register are guaranteed valid while STOP is asserted. STOP may be negated by the CPU and by reset.

The system software must stop each submodule before asserting STOP to avoid complications at restart and to avoid data corruption. The SCI submodule receiver and transmitter should be disabled, and the operation should be verified for completion before STOP is asserted. The QSPI submodule should be stopped by asserting the HALT bit in SPCR3 and by asserting STOP after the HALTA flag is set.

FRZ1 — Freeze 1

0 = Ignore the FREEZE signal on the IMB

1 = Halt the QSPI (on a transfer boundary)

FRZ1 determines what action is taken by the QSPI when the FREEZE signal of the IMB is asserted. FREEZE is asserted whenever the CPU enters background mode.

FRZ0 — Freeze 0

Reserved for future enhancement.

Bits [12:8] — Not Implemented

SUPV — Supervisor/Unrestricted

0 = User access

1 = Supervisor access

SUPV defines the assignable QSM registers as either supervisor-only data space or unrestricted data space.

Bits [6:4] — Not Implemented

IARB — Interrupt Arbitration Identification Number

Each module that generates interrupts must have an IARB field. Each has a unique value in this field that is used to arbitrate for the IMB when modules generate simultaneous interrupts of the same priority. Refer to **4 Single-Chip Integration Module** for more information.

QTEST — QSM Test Register **$YFFC02**

QTEST is used during factory test of the QSM. Accesses to QTEST must be made while the MCU is in test mode.

QILR — QSM Interrupt Levels Register **$YFFC04**

15	14	13	12	11	10	9	8	7	0
0	0	ILQSPI			ILSCI			QIVR (QSM Interrupt Vector Register)	
RESET:									
0	0	0	0	0	0	0	0		

QILR determines the priority level of interrupts requested by the QSM and the vector used when an interrupt is acknowledged.

ILQSPI — Interrupt Level for QSPI

ILQSPI determines the priority of all QSPI interrupts. This field must be given a value between $0 (interrupts disabled) to $7 (highest priority).

ILSCI — Interrupt Level of SCI

ILSCI determines the priority of SCI interrupts. This field must be given a value between $0 (interrupts disabled) to $7 (highest priority).

If ILQSPI and ILSCI are the same (nonzero) value, and both submodules simultaneously request interrupt service, QSPI has priority.

QIVR — QSM Interrupt Vector Register **$YFFC05**

15	8	7	6	5	4	3	2	1	0
QILR (QSM Interrupt Levels Register)		INTV							
		RESET:							
		0	0	0	0	1	1	1	1

At reset, QIVR is initialized to $0F, which corresponds to the uninitialized interrupt vector in the exception table. This vector is selected until QIVR is written. A user-defined vector ($40–$FF) should be written to QIVR during QSM initialization.

After initialization, QIVR determines which two vectors in the exception vector table are to be used for QSM interrupts. The QSPI and SCI submodules have separate interrupt vectors adjacent to each other. Both submodules use the same interrupt vector with the least significant bit (LSB) determined by the submodule causing the interrupt.

The value of INTV0 used during an interrupt-acknowledge cycle is supplied by the QSM. During an interrupt acknowledge cycle, INTV[7:1] are driven on DATA[7:1] IMB lines. DATA0 is negated for an SCI interrupt and asserted for a QSPI interrupt. Writes to INTV0 have no meaning or effect. Reads of INTV0 return a value of one.

6.3.2 Pin Control Registers

The QSM uses nine pins, eight of which form a parallel port on the MCU. Although these pins are used by the serial subsystems, any pin may alternately be assigned as general-purpose I/O on a pin-by-pin basis.

Pins used for general-purpose I/O must not be assigned to the QSPI by register QPAR. To avoid driving incorrect data, the first byte to be output must be written before QDDR is configured. QDDR must then be written to determine the direction of data flow and to output the value contained in register QPDR. Subsequent data for output is written to QPDR.

QPDR — QSM Port Data Register **$YFFC15**

15 … 8	7	6	5	4	3	2	1	0
RESERVED	D7 (TXD)	D6 (PCS3)	D5 (PCS2)	D4 (PCS1)	D3 (PCS0/$\overline{SS}$)	D2 (SCK)	D1 (MOSI)	D0 (MISO)
	RESET:							
	0	0	0	0	0	0	0	0

QPDR is the data register for pins defined in QPAR. Writes to QPDR affect pins defined as outputs. Reads of QPDR return data present on the pins.

QPAR — QSM Pin Assignment Register **$YFFC16**

15	14	13	12	11	10	9	8	7 … 0
0	PCS3	PCS2	PCS1	PCS0/$\overline{SS}$	0	MOSI	MISO	QDDR (QSM Data Direction Register)
RESET:								
0	0	0	0	0	0	0	0	

QPAR determines whether certain pins are used by the QSPI submodule, or whether they are available for general-purpose I/O. Pins designated for general-purpose I/O are controlled by QDDR and QPDR. QPAR does not affect operation of the SCI submodule. Bits 15 and 10 are not implemented.

PCS[3:1] — Peripheral Chip Selects

PCS0/SS — Peripheral Chip Select 0/Slave Select

MOSI — Master Out Slave In

MISO — Master In Slave Out
0 = Used for general-purpose I/O
1 = Used by QSPI submodule

QDDR — QSM Data Direction Register **$YFFC17**

15 … 8	7	6	5	4	3	2	1	0
QPAR (QSM Pin Assignment Register)	TXD	PCS3	PCS2	PCS1	PCS0/$\overline{SS}$	SCK	MOSI	MISO
RESET:	0	0	0	0	0	0	0	0

QDDR determines whether a general-purpose I/O pin is an input or an output. During reset, all QSM pins are configured as general-purpose inputs. The following assignments apply to all bits in the QDDR:

0 = Input
1 = Output

TXD — Transmit Data

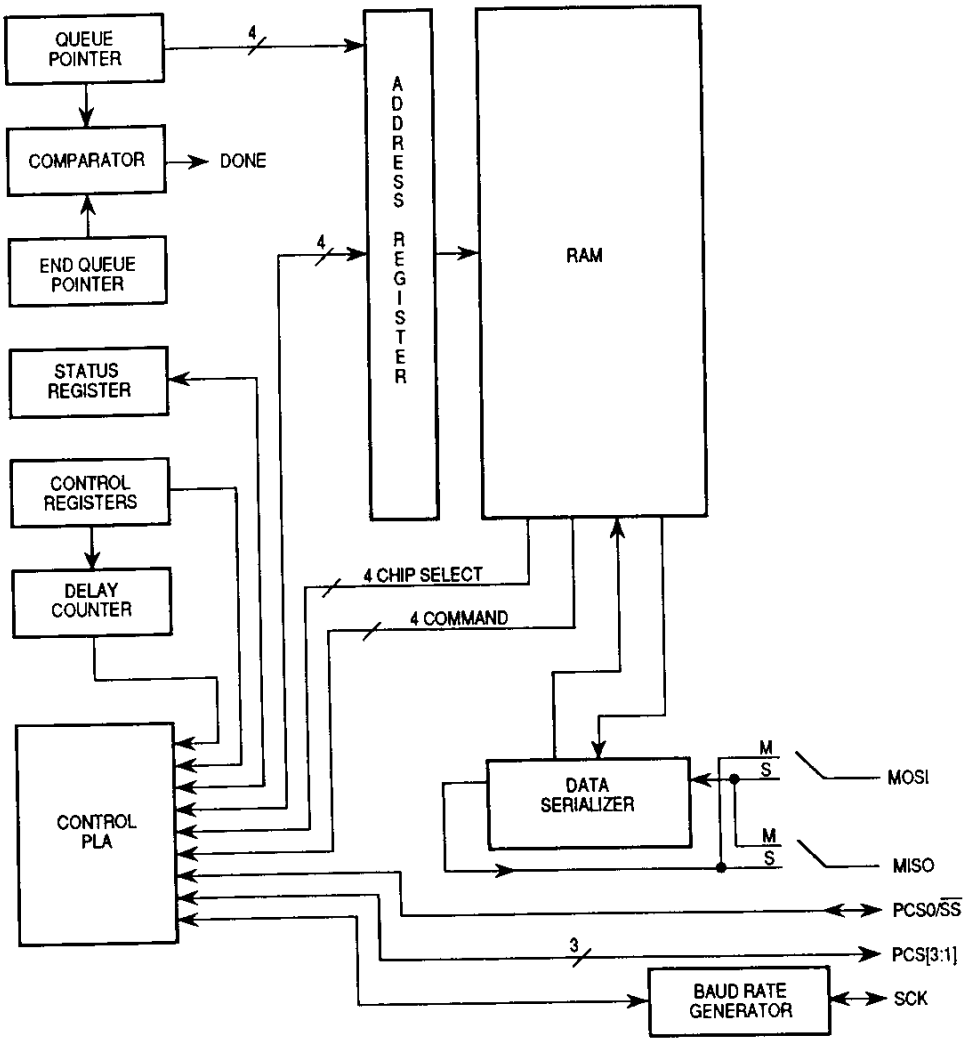
This bit determines the direction of the TXD pin only when the SCI transmitter is disabled. When the SCI transmitter is enabled, the TXD pin is an output.

The following bits determine the QSPI port pin operation to be input or output.

PCS[3:1] — Peripheral Chip Selects
PCS0/SS — Peripheral Chip Select 0/Slave Select
SCK — Serial Clock
MOSI — Master Out Slave In
MISO — Master In Slave Out

6.4 QSPI Submodule

The QSPI submodule communicates with external devices via a synchronous serial bus. The QSPI is fully compatible with the serial peripheral interface (SPI) systems found on other Motorola products. A block diagram of the QSPI follows.



QSPI Block Diagram

6.4.1 QSPI Pins

Seven pins are associated with the QSPI. When not needed for a QSPI application, they may be configured as general-purpose I/O pins. PCS0/$\overline{SS}$ can be programmed as a peripheral chip select or a slave select. The table below lists QSPI pins and their functions.

Pin Name	Mnemonic	Mode	Function
Master In Slave Out	MISO	Master Slave	Serial data input to QSPI Serial data output from QSPI
Master Out Slave In	MOSI	Master Slave	Serial data output from QSPI Serial data input to QSPI
Serial Clock	SCK[1]	Master Slave	Clock output from QSPI Clock input to QSPI
Peripheral Chip Selects	PCS[3:1]	Master	Outputs select peripherals
Peripheral Chip Select[2] Slave Select[3]	PCS0/ $\overline{SS}$	Master Slave	Select peripherals Input selects the QSPI
Slave Select[4]	$\overline{SS}$	Master	May cause mode fault

NOTES:

1. All QSPI pins except SCK can be used as general-purpose I/O if they are not used by the QSPI while the QSPI is operating.
2. An output (PCS0) when the QSPI is in master mode.
3. An input ($\overline{SS}$) when the QSPI is in slave mode.
4. An input ($\overline{SS}$) when the QSPI is in master mode; useful in multimaster systems.

6.4.2 QSPI Registers

The programmer's model for the QSPI submodule consists of the QSM global and pin control registers, four QSPI control registers, one status register, and the 80-byte QSPI RAM.

Registers and RAM can be read and written by the CPU. The four control registers must be initialized before the QSPI is enabled to ensure defined operation. SPCR1 should be written last because it contains QSPI enable bit SPE. Asserting this bit starts the QSPI. The QSPI control registers are reset to a defined state and may then be changed by the CPU. Reset values are shown below each register.

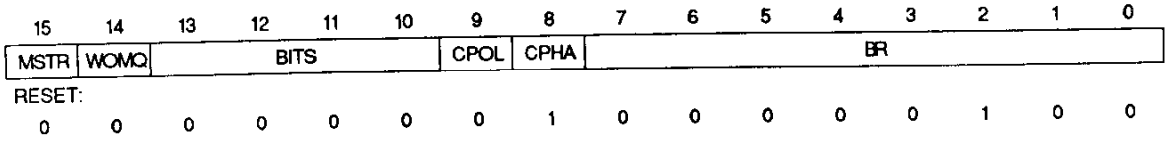
A memory map of the QSPI is shown below.

Address	Name	Usage
$YFFC18	SPCR0	QSPI Control Register 0
$YFFC1A	SPCR1	QSPI Control Register 1
$YFFC1C	SPCR2	QSPI Control Register 2
$YFFC1E	SPCR3	QSPI Control Register 3
$YFFC1F	SPSR	QSPI Status Register
$YFFD00	RAM	QSPI Receive Data (16 Words)
$YFFD20	RAM	QSPI Transmit Data (16 Words)
$YFFD40	RAM	QSPI Command Control (8 Words)

Writing a different value into any control register except SPCR2 while the QSPI is enabled disrupts operation. SPCR2 is buffered to prevent disruption of the current serial transfer. After completion of the current serial transfer, the new SPCR2 values become effective.

Writing the same value into any control register except SPCR2 while the QSPI is enabled has no effect on QSPI operation. Rewriting NEWQP in SPCR2 causes execution to restart at the designated location.

SPCR0 — QSPI Control Register 0 **$YFFC18**



SPCR0 contains parameters for configuring the QSPI before it is enabled. The CPU can read and write this register; the QSM has read-only access.

MSTR — Master/Slave Mode Select

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

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

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

WOMQ — Wired-OR Mode for QSPI Pins

0 = Outputs have normal MOS drivers.

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

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

BITS — Bits Per Transfer

In master mode, when BITSE in a command is set, the BITS field determines the number of data bits transferred. When BITSE is cleared, 8 bits are transferred. Reserved values default to 8 bits. BITSE is not used in slave mode.

The following table shows the number of bits per transfer.

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

CPOL — Clock Polarity

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

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

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

CPHA — Clock Phase

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

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

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

BR — Serial Clock Baud Rate

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

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

or

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

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

Giving BR a value of zero or one disables the baud rate generator. SCK is disabled and assumes its inactive state value. No serial transfers occur. At reset, BAUD is initialized to a 2.1 MHz SCK frequency ($04).

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

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

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

SPE — QSPI Enable

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

DSCKL — Delay before SCK

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

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

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

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

DTL — Length of Delay after Transfer

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

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

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

A zero value for DTL causes a delay-after-transfer value of [(32 • 256)/System Clock].

If DT equals zero, a standard delay is inserted.

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

$$= 1\ \mu\text{s with a 16.78 MHz system clock}$$

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

SPCR2 — QSPI Control Register 2 **$YFFC1C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPIFIE	WREN	WRTO	0	ENDQP				0	0	0	0	NEWQP			
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPCR2 contains QSPI configuration parameters. Although the CPU can read and write this register, the QSM has read access only. Writes to SPCR2 are buffered. A write to SPCR2 that changes a bit value while the QSPI is operating is ineffective for the current serial transfer, but becomes effective on the next serial transfer. Reads of SPCR2 return the current value of the register, not of the buffer.

SPIFIE — SPI Finished Interrupt Enable
0 = QSPI interrupts disabled
1 = QSPI interrupts enabled
SPIFIE enables the QSPI to generate a CPU interrupt upon assertion of the status flag SPIF.

WREN — Wrap Enable
0 = Wraparound mode disabled
1 = Wraparound mode enabled
WREN enables or disables wraparound mode.

WRTO — Wrap To
When wraparound mode is enabled, after the end of the queue has been reached, WRTO determines which address the QSPI executes next.

Bit 12 — Not Implemented

ENDQP — Ending Queue Pointer
This field contains the last QSPI queue address.

Bits [7:4] — Not Implemented

NEWQP — New Queue Pointer Value
This field contains the first QSPI queue address.

SPCR3 — QSPI Control Register 3 **$YFFC1E**

15	14	13	12	11	10	9	8	7	0
0	0	0	0	0	LOOPQ	HMIE	HALT	SPSR	
RESET:									
0	0	0	0	0	0	0	0		

SPCR3 contains QSPI configuration parameters. The CPU can read and write SPCR3, but the QSM has read-only access.

Bits [15:11] — Not Implemented

LOOPQ — QSPI Loop Mode

0 = Feedback path disabled

1 = Feedback path enabled

LOOPQ controls feedback on the data serializer for testing.

HMIE — HALTA and MODF Interrupt Enable

0 = HALTA and MODF interrupts disabled

1 = HALTA and MODF interrupts enabled

HMIE controls CPU interrupts caused by the HALTA status flag or the MODF status flag in SPSR.

HALT — Halt

0 = Halt not enabled

1 = Halt enabled

When HALT is asserted, the QSPI stops on a queue boundary. The QSPI is then in a defined state from which it can later be restarted.

SPSR — QSPI Status Register **$YFFC1F**

15 — 8	7	6	5	4	3	2	1	0
SPCR3	SPIF	MODF	HALTA	0	CPTQP			
	RESET:							
	0	0	0	0	0	0	0	0

SPSR contains QSPI status information. Only the QSPI can assert the bits in this register. The CPU reads this register to obtain status information and writes it to clear status flags.

SPIF — QSPI Finished Flag

0 = QSPI not finished

1 = QSPI finished

SPIF is set after execution of the command at the address in ENDQP.

MODF — Mode Fault Flag

0 = Normal operation

1 = Another SPI node requested to become the network SPI master while the QSPI was enabled in master mode ($\overline{SS}$ input taken low).

MODF is asserted by the QSPI when the QSPI is the serial master (MSTR = 1) and the $\overline{SS}$ input pin is negated by an external driver.

HALTA — Halt Acknowledge Flag

0 = QSPI not halted

1 = QSPI halted

HALTA is asserted when the QSPI halts in response to CPU assertion of HALT.

Bit 4 — Not Implemented

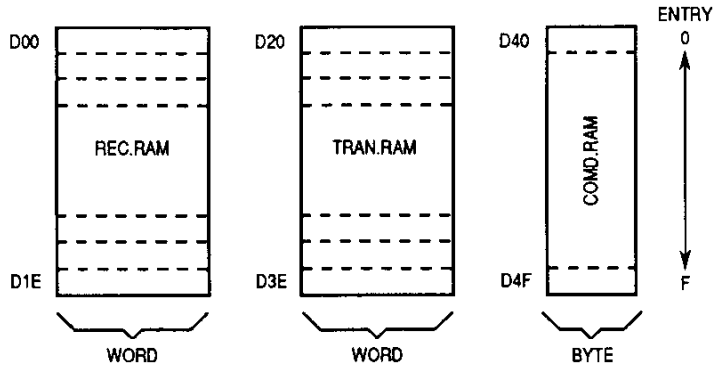
CPTQP — Completed Queue Pointer

CPTQP points to the last command executed. It is updated when the current command is complete. When the first command in a queue is executing, CPTQP contains either the reset value ($0) or a pointer to the last command completed in the previous queue.

6.4.3 QSPI RAM

The QSPI contains an 80-byte block of dual-access static RAM that is used by both the QSPI and the CPU. The RAM is divided into three segments: receive data (REC.RAM), transmit data (TRAN.RAM), and command control (COMD.RAM). Receive data is information received from a serial device external to the MCU. Transmit data is information stored by the CPU for transmission to an external peripheral. Command control data is used to perform the transfer.

Organization of the RAM is illustrated below:



QSPI RAM Organization

Once the CPU has set up the queue of QSPI commands and enabled the QSPI, the QSPI can operate independently of the CPU. The QSPI executes all of the commands in its queue, sets a flag indicating that it is finished, and then either interrupts the CPU or waits for CPU intervention. It is possible to repeatedly execute a queue of commands without CPU intervention.

REC.RAM — Receive Data RAM **$YFFD00**

Data received by the QSPI is stored in this segment. The CPU reads this segment to retrieve data from the QSPI. Data stored in receive RAM is right-justified. Unused bits in a receive queue entry are set to zero by the QSPI upon completion of the individual queue entry. The CPU can access the data using byte, word, or long-word addressing.

The CPTQP value in SPSR shows which queue entries have been executed. The CPU uses this information to determine which locations in receive RAM contain valid data before reading them.

TRAN.RAM — Transmit Data RAM **$YFFD20**

Data that is to be transmitted by the QSPI is stored in this segment. The CPU normally writes one word of data into this segment for each queue command to be executed.

Information to be transmitted must be written to transmit data RAM in a right-justified format. The QSPI cannot modify information in the transmit data RAM. The QSPI copies the information to its data serializer for transmission. Information remains in transmit RAM until overwritten.

COMD.RAM — Command RAM **$YFFD40**

7	6	5	4	3	2	1	0
CONT	BITSE	DT	DSCK	PCS3	PCS2	PCS1	PCS0*
—	—	—	—	—	—	—	—
CONT	BITSE	DT	DSCK	PCS3	PCS2	PCS1	PCS0*
COMMAND CONTROL				PERIPHERAL CHIP SELECT			

*The PCS0 bit represents the dual-function PCS0/$\overline{SS}$.

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

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

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

PCS[3:0] — Peripheral Chip Select

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

$\overline{SS}$ — Slave Mode Select

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

CONT — Continue

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

BITSE — Bits per Transfer Enable

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

DT — Delay after Transfer

The QSPI provides a variable delay at the end of serial transfer to facilitate interfacing with peripherals that have a latency requirement. The delay between transfers is determined by the DTL field in SPCR1 and can have a value in the range of 1 to 100 µs.

DSCK — PCS to SCK Delay

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

6.4.4 Operating Modes

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

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

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

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

6.5 SCI Submodule

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

6.5.1 SCI Pins

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

SCI pins and their functions are shown below.

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

6.5.2 SCI Registers

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

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

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

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

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

Bits [15:13] — Not Implemented

BR — Baud Rate

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

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

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

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

Writing a value of zero to BR disables the baud rate generator. Examples of SCI baud rates are shown in the following table.

Nominal Baud Rate	Actual Baud Rate	Percent Error	Value of SCBR
500,000.00	524,288.00	4.86	1
38,400.00	37,449.14	–2.48	14
32,768.00	32,768.00	0.00	16
19,200.00	19,418.07	1.14	27
9,600.00	9,532.51	–0.70	55
4,800.00	4,809.98	0.21	109
2,400.00	2,404.99	0.21	218
1,200.00	1,199.74	–0.02	437
600.00	599.87	–0.02	874
300.00	299.94	–0.02	1,748
110.00	110.01	0.01	4,766
64.00	64.00	0.01	8,191

NOTE: These rates are based on a 16.78-MHz system clock.

SCCR1 — SCI Control Register 1 **$YFFC0A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	LOOPS	WOMS	ILT	PT	PE	M	WAKE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SCCR1 contains SCI configuration parameters. The CPU can read and write this register at any time. The SCI can modify RWU in some circumstances. In general, interrupts enabled by these control bits are cleared by reading SCSR, then reading (receiver status bits) or writing (transmitter status bits) the SCDR.

Bit 15 — Not Implemented

LOOPS — Loop Mode

0 = Normal SCI operation, no looping, feedback path disabled

1 = Test SCI operation, looping, feedback path enabled

LOOPS controls a feedback path on the data serial shifter. When loop mode is enabled, SCI transmitter output is fed back into the receive serial shifter. TXD is asserted (idle line). Both transmitter and receiver must be enabled prior to entering loop mode.

WOMS — Wired-OR Mode for SCI Pins

0 = If configured as an output, TXD is a normal CMOS output.

1 = If configured as an output, TXD is an open-drain output.

WOMS determines whether the TXD pin is an open-drain output or a normal CMOS output. This bit is used only when TXD is an output. If TXD is used as a general-purpose input pin, WOMS has no effect.

ILT — Idle-Line Detect Type

0 = Short idle-line detect (start count on first one)

1 = Long idle-line detect (start count on first one after stop bit(s))

PT — Parity Type

0 = Even parity

1 = Odd parity

When parity is enabled, PT determines whether parity is even or odd for both the receiver and the transmitter.

PE — Parity Enable

0 = SCI parity disabled

1 = SCI parity enabled

PE determines whether parity is enabled or disabled for both the receiver and the transmitter. If the received parity bit is not correct, the SCI sets the PF error flag in SCSR.

When PE is set, the MSB of the data field is used for the parity function, which results in either seven or eight bits of user data, depending on the condition of M bit. The following table lists the available choices.

M	PE	Result
0	0	8 Data Bits
0	1	7 Data Bits, 1 Parity Bit
1	0	9 Data Bits
1	1	8 Data Bits, 1 Parity Bit

M — Mode Select
0 = SCI frame: 1 start bit, 8 data bits, 1 stop bit (10 bits total)
1 = SCI frame: 1 start bit, 9 data bits, 1 stop bit (11 bits total)

WAKE — Wakeup by Address Mark
0 = SCI receiver awakened by idle-line detection
1 = SCI receiver awakened by address mark (last bit set)

TIE — Transmit Interrupt Enable
0 = SCI TDRE interrupts inhibited
1 = SCI TDRE interrupts enabled

TCIE — Transmit Complete Interrupt Enable
0 = SCI TC interrupts inhibited
1 = SCI TC interrupts enabled

RIE — Receiver Interrupt Enable
0 = SCI RDRF and OR interrupts inhibited
1 = SCI RDRF and OR interrupts enabled

ILIE — Idle-Line Interrupt Enable
0 = SCI IDLE interrupts inhibited
1 = SCI IDLE interrupts enabled

TE — Transmitter Enable
0 = SCI transmitter disabled (TXD pin may be used as I/O)
1 = SCI transmitter enabled (TXD pin dedicated to SCI transmitter)
The transmitter retains control of the TXD pin until completion of any character transfer that was in progress when TE is cleared.

RE — Receiver Enable
0 = SCI receiver disabled (status bits inhibited)
1 = SCI receiver enabled

RWU — Receiver Wakeup
0 = Normal receiver operation (received data recognized)
1 = Wakeup mode enabled (received data ignored until awakened)
Setting RWU enables the wakeup function, which allows the SCI to ignore received data until awakened by either an idle line or address mark (as determined by WAKE). When in wakeup mode,

the receiver status flags are not set, and interrupts are inhibited. This bit is cleared automatically (returned to normal mode) when the receiver is awakened.

SBK — Send Break

0 = Normal operation

1 = Break frame(s) transmitted after completion of current frame

SBK provides the ability to transmit a break code from the SCI. If the SCI is transmitting when SBK is set, it will transmit continuous frames of zeros after it completes the current frame, until SBK is cleared. If SBK is toggled (one to zero in less than one frame interval), the transmitter sends only one or two break frames before reverting to idle line or commencing to send data.

SCSR — SCI Status Register **$YFFC0C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED							TDRE	TC	RDRF	RAF	IDLE	OR	NF	FE	PF
RESET:															
0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

The SCSR contains flags that show SCI operational conditions. These flags can be cleared either by hardware or by a special acknowledgement sequence. The sequence consists of a read of the SCSR with flags set, followed by a read of the SCDR (or a write of the SCDR in the case of TDRE and TC). A long-word read can consecutively access both the SCSR and the SCDR. This action clears receive status flag bits that were set at the time of the read, but does not clear TDRE or TC flags.

If an internal SCI signal for setting a status bit comes after the CPU has read the asserted status bits, but before the CPU has written or read the SCDR, the newly set status bit is not cleared — the SCSR must be read again with the bit set, and the SCDR must be written or read before the status bit is cleared.

Reading either byte of the SCSR causes all 16 bits to be accessed, and any status bit already set in either byte will be cleared on a subsequent read or write of the SCDR.

TDRE — Transmit Data Register Empty Flag

0 = The TDR still contains data to be sent to the transmit serial shifter.

1 = A new character may now be written to the TDR.

TDRE is set when the byte in the TDR is transferred to the transmit serial shifter. If TDRE is zero, transfer has not occurred and a write to the TDR will overwrite the previous value. New data is not transmitted if the TDR is written without first clearing TDRE.

TC — Transmit Complete Flag

0 = SCI transmitter is busy.

1 = SCI transmitter is idle.

TC is set when the transmitter finishes shifting out all data, queued preambles (mark/idle line), or queued breaks (logic zero). The flag may be cleared by reading the SCSR when TC is set and then by writing the transmit data register (TDR).

RDRF — Receive Data Register Full Flag

0 = Register RDR is empty or contains previously read data.

1 = Register RDR contains new data.

RDRF is set when the content of the receive serial shifter is transferred to the RDR. If one or more errors are detected in the received word, the appropriate flags (NF, FE, and PF) are set within the same clock cycle.

RAF — Receiver Active Flag

0 = SCI receiver is idle.

1 = SCI receiver is busy.

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

IDLE — Idle-Line Detected Flag

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

1 = SCI receiver detected an idle-line condition.

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

OR — Overrun Error Flag

0 = RDRF is cleared before new data arrives.

1 = RDRF is not cleared before new data arrives.

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

NF — Noise Error Flag

0 = No noise detected on the received data.

1 = Noise occurred on the received data.

NF is set when the SCI receiver detects noise on a valid start bit, on any data bit, or on a stop bit. It is not set by noise on the idle line or on invalid start bits. Each bit is sampled three times. If all three samples are not the same logic level, the majority value is used for the received data value, and NF is set. NF is not set until an entire frame is received and RDRF is set.

FE — Framing Error Flag

0 = No framing error on the received data.

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

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

PF — Parity Error Flag

0 = No parity error on the received data.

1 = Parity error occurred on the received data.

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

SCDR — SCI Data Register **$YFFC0E**

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

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

7 Analog-to-Digital Converter Module (ADC)

The ADC is a unipolar, successive-approximation converter with eight modes of operation. It has selectable 8- or 10-bit resolution. Accuracy is ±1 count (1 LSB) in 8-bit mode and ±4 counts (2 LSB) in 10 bit mode. Monotonicity is guaranteed in both modes. The ADC can perform an 8-bit single conversion (4-clock sample) in 10 microseconds and a 10-bit single conversion in 11 microseconds.

The ADC consists of an analog and a digital subsystem. A block diagram of the converter appears on the following page.

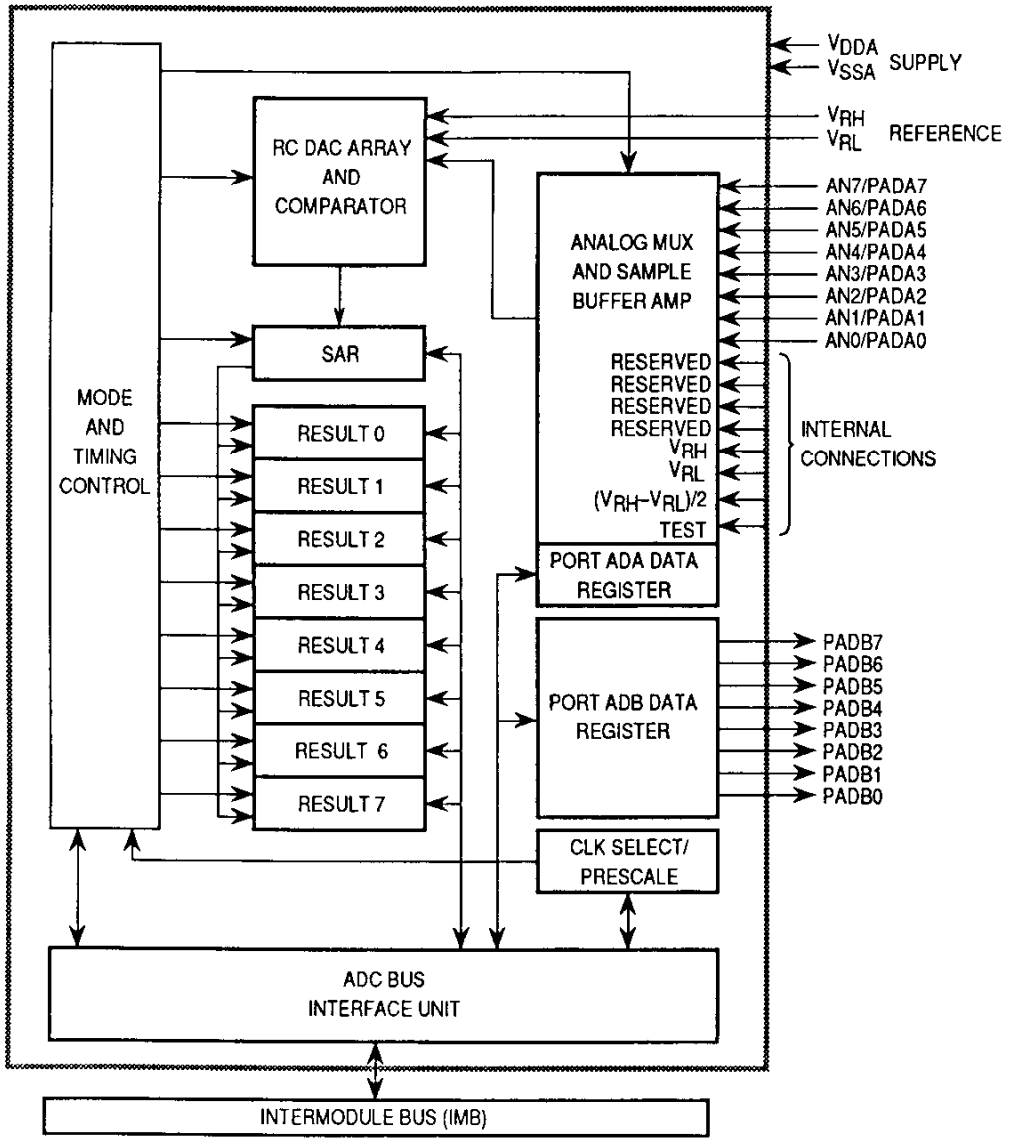
7.1 Analog Subsystem

The analog front end consists of a multiplexer, a buffer amplifier, a resistor-capacitor array, and a high-gain comparator. The multiplexer selects one of eight internal or eight external signal sources for conversion. The resistor capacitor (RC) array performs two functions: it acts as a sample/hold circuit, and it provides the digital-to-analog comparison output necessary for successive approximation conversion. The comparator indicates whether each successive output of the RC array is higher or lower than the sampled input.

7.2 Digital Control Subsystem

The digital control section includes conversion sequence control logic, channel and reference select logic, a successive approximation register, eight result registers, a port data register, and control/status registers. It controls the multiplexer and the output of the RC array during the sample and conversion periods, stores the results of comparison in the successive-approximation register, and transfers the result to a result register.

The ADC bus interface unit (ABIU) contains logic necessary to interface the ADC to the intermodule bus. The ADC is designed to act as a slave device on the bus. The interface must respond with appropriate bus cycle termination signals and supply appropriate interface timing to the other submodules.



Analog-to-Digital Converter Block Diagram

7.3 ADC Register Map

Address	Control Register
$YFF700	ADC Module Configuration Register (ADCMCR)
$YFF702	Module Test Register (ADCTEST)
$YFF704	(Reserved)
$YFF706	Port Data Register (PDR)
$YFF708	(Reserved)
$YFF70A	ADC Control Register 0 (ADCTL0)
$YFF70C	ADC Control Register 1 (ADCTL1)
$YFF70E	ADC Status Register (ADSTAT)
Address	**Right-Justified Unsigned Result Registers**
$YFF710	ADC Result Register 0 (RSLT0)
$YFF712	ADC Result Register 1 (RSLT1)
$YFF714	ADC Result Register 2 (RSLT2)
$YFF716	ADC Result Register 3 (RSLT3)
$YFF718	ADC Result Register 4 (RSLT4)
$YFF71A	ADC Result Register 5 (RSLT5)
$YFF71C	ADC Result Register 6 (RSLT6)
$YFF71E	ADC Result Register 7 (RSLT7)
Address	**Left-Justified Signed Result Registers**
$YFF720	ADC Result Register 0 (RSLT0)
$YFF722	ADC Result Register 1 (RSLT1)
$YFF724	ADC Result Register 2 (RSLT2)
$YFF726	ADC Result Register 3 (RSLT3)
$YFF728	ADC Result Register 4 (RSLT4)
$YFF72A	ADC Result Register 5 (RSLT5)
$YFF72C	ADC Result Register 6 (RSLT6)
$YFF72E	ADC Result Register 7 (RSLT7)
Address	**Left-Justified Unsigned Result Registers**
$YFF730	ADC Result Register 0 (RSLT0)
$YFF732	ADC Result Register 1 (RSLT1)
$YFF734	ADC Result Register 2 (RSLT2)
$YFF736	ADC Result Register 3 (RSLT3)
$YFF738	ADC Result Register 4 (RSLT4)
$YFF73A	ADC Result Register 5 (RSLT5)
$YFF73C	ADC Result Register 6 (RSLT6)
$YFF73E	ADC Result Register 7 (RSLT7)

Y = M111, where M is the MODMAP bit in the SCIMCR.

7.4 ADC Registers

ADCMCR — ADC Module Configuration Register **$YFF700**

15	14	13	12	8	7	6	0
STOP	FRZ		NOT USED		SUPV	NOT USED	
RESET:							
1	0	0			0		

The ADCMCR is used to initialize the ADC.

STOP — STOP Mode

0 = Normal operation

1 = Low-power operation

STOP places the ADC in low-power state by disabling the ADC clock and powering down the analog circuitry. Setting STOP aborts any conversion in progress. STOP is set to logic level one at reset, and may be cleared to logic level zero by the CPU.

Clearing STOP enables normal ADC operation. However, because analog circuitry bias current has been turned off, there is a period of recovery before output stabilization.

FRZ[1:0] — Freeze 1

The FRZ field is used to determine ADC response to assertion of the FREEZE signal. The following table shows possible responses.

FRZ	Response
00	Ignore FREEZE
01	Reserved
10	Finish conversion, then freeze
11	Freeze immediately

SUPV — Supervisor/Unrestricted

0 = Unrestricted access

1 = Supervisor access

SUPV defines access to assignable ADC registers.

ADTEST — ADC Test Register **$YFF702**

ADTEST is used during factory test of the ADC.

PDR — Port Data Register **$YFF706**

15	8	7	0
PORT B DATA		PORT A DATA	
RESET:			
0 0 0 0 0 0 0 0		Reflects state of the input pins	

Port A is an input port that shares pins with the A/D converter inputs. Port B is a dedicated output port.

PDR[7:0] — Port A Data

A read of PDR[7:0] will return the logic level of the port A pins. If the input is outside the defined voltage range, the result of the read will be indeterminate. Use of a port A pin for digital input does not preclude use as an analog input.

PDR[15:8] — Port B Data

PDR[15:8] is a dedicated output port.

ADCTL0 — A/D Control Register 0 **$YFF70A**

15							8	7	6	5	4	3	2	1	0
NOT USED								RES10	STS		PRS				
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

ADCTL0 is used to select ADC clock source and to set up prescaling. Writes to it have immediate effect.

RES10 — 10-Bit Resolution

0 = 8-bit resolution

1 = 10-bit resolution

Conversion results are appropriately aligned in result registers to reflect conversion status.

STS[1:0] — Sample Time Select

Total conversion time depends on initial sample time, transfer time, final sample time, and resolution time. Initial sample time is fixed at two clocks. Transfer time is fixed at two clocks. Resolution time is fixed at 10 ADC clock cycles for an 8-bit conversion and 12 ADC clock cycles for a 10-bit conversion. Final sample time depends on the STS field, as shown below.

STS[1:0]	Sample Time
00	2 A/D Clock Periods
01	4 A/D Clock Periods
10	8 A/D Clock Periods
11	16 A/D Clock Periods

PRS[4:0] — Prescaler Rate Selection Field

ADC clock is generated from system clock using a modulo counter and a divide-by-two circuit. The binary value of this field is the counter modulus. System clock is divided by the PRS value plus one, then sent to the divide-by-two circuit, as shown in the following table. Maximum ADC clock rate is 2 MHz. Reset value of PRS is a divisor value of 8, resulting in a nominal 2-MHz ADC clock.

PRS[4:0]	Divisor Value
00000	4
00001	4
00010	6
...	...
11101	60
11110	62
11111	64

ADCTL1 — A/D Control Register 1 **$YFF70C**

15							8	7	6	5	4	3	2	1	0
NOT USED									SCAN	MULT	S8CM	CD	CC	CB	CA
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCTL1 is used to initiate an A/D conversion and to select conversion modes and a conversion channel. It can be written or read at any time. A write to ADCTL1 initiates a conversion sequence. If a conversion sequence is already in progress, a write to ADCTL1 aborts it and resets the SCF and CCF flags in the ADC status register.

SCAN — Scan Mode Selection Bit

0 = Single conversion sequence

1 = Continuous conversion

Length of conversion sequence(s) is determined by S8CM.

MULT — Multichannel Conversion Bit

0 = Conversion sequence(s) run on single channel (channel selected via [CD:CA])

1 = Sequential conversion of a block of four or eight channels (block selected via [CD:CA])

Length of conversion sequence(s) is determined by S8CM.

S8CM — Select Eight-Conversion Sequence Mode

0 = Four-conversion sequence

1 = Eight-conversion sequence

This bit determines the number of conversions in a conversion sequence.

[CD:CA] — Channel Selection Field

The bits in this field are used to select an input or block of inputs for A/D conversion.

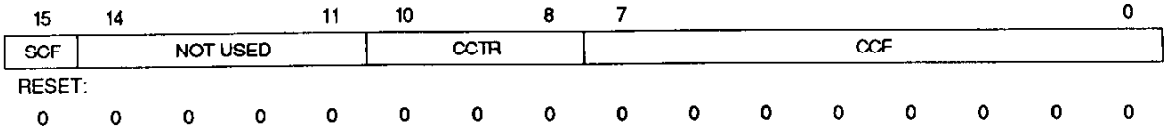
The following table summarizes the operation of S8CM and [CD:CA] when MULT is cleared (single channel mode). Number of conversions per channel is determined by SCAN.

S8CM	CD	CC	CB	CA	Input	Result Register
0	0	0	0	0	AN0	RSLT0 – RSLT3
0	0	0	0	1	AN1	RSLT0 – RSLT3
0	0	0	1	0	AN2	RSLT0 – RSLT3
0	0	0	1	1	AN3	RSLT0 – RSLT3
0	0	1	0	0	AN4	RSLT0 – RSLT3
0	0	1	0	1	AN5	RSLT0 – RSLT3
0	0	1	1	0	AN6	RSLT0 – RSLT3
0	0	1	1	1	AN7	RSLT0 – RSLT3
0	1	0	0	0	RESERVED	RSLT0 – RSLT3
0	1	0	0	1	RESERVED	RSLT0 – RSLT3
0	1	0	1	0	RESERVED	RSLT0 – RSLT3
0	1	0	1	1	RESERVED	RSLT0 – RSLT3
0	1	1	0	0	V_{RH}	RSLT0 – RSLT3
0	1	1	0	1	V_{RL}	RSLT0 – RSLT3
0	1	1	1	0	$(V_{RH} - V_{RL}) / 2$	RSLT0 – RSLT3
0	1	1	1	1	TEST/RESERVED	RSLT0 – RSLT3
1	0	0	0	0	AN0	RSLT0 – RSLT7
1	0	0	0	1	AN1	RSLT0 – RSLT7
1	0	0	1	0	AN2	RSLT0 – RSLT7
1	0	0	1	1	AN3	RSLT0 – RSLT7
1	0	1	0	0	AN4	RSLT0 – RSLT7
1	0	1	0	1	AN5	RSLT0 – RSLT7
1	0	1	1	0	AN6	RSLT0 – RSLT7
1	0	1	1	1	AN7	RSLT0 – RSLT7
1	1	0	0	0	RESERVED	RSLT0 – RSLT7
1	1	0	0	1	RESERVED	RSLT0 – RSLT7
1	1	0	1	0	RESERVED	RSLT0 – RSLT7
1	1	0	1	1	RESERVED	RSLT0 – RSLT7
1	1	1	0	0	V_{RH}	RSLT0 – RSLT7
1	1	1	0	1	V_{RL}	RSLT0 – RSLT7
1	1	1	1	0	$(V_{RH} - V_{RL}) / 2$	RSLT0 – RSLT7
1	1	1	1	1	TEST/RESERVED	RSLT0 – RSLT7

The following table summarizes the operation of S8CM and [CD:CA] when MULT is set (multichannel mode). Number of conversions per channel is determined by SCAN. Channel numbers are given in order of conversion.

S8CM	CD	CC	CB	CA	Input	Result Register
0	0	0	X	X	AN0	RSLT0
					AN1	RSLT1
					AN2	RSLT2
					AN3	RSLT3
0	0	1	X	X	AN4	RSLT0
					AN5	RSLT1
					AN6	RSLT2
					AN7	RSLT3
0	1	0	X	X	RESERVED	RSLT0
					RESERVED	RSLT1
					RESERVED	RSLT2
					RESERVED	RSLT3
0	1	1	X	X	V_{RH}	RSLT0
					V_{RL}	RSLT1
					$(V_{RH} - V_{RL}) / 2$	RSLT2
					TEST/RESERVED	RSLT3
1	0	X	X	X	AN0	RSLT0
					AN1	RSLT1
					AN2	RSLT2
					AN3	RSLT3
					AN4	RSLT4
					AN5	RSLT5
					AN6	RSLT6
					AN7	RSLT7
1	1	X	X	X	RESERVED	RSLT0
					RESERVED	RSLT1
					RESERVED	RSLT2
					RESERVED	RSLT3
					V_{RH}	RSLT4
					V_{RL}	RSLT5
					$(V_{RH} - V_{RL}) / 2$	RSLT6
					TEST/RESERVED	RSLT7

ADSTAT — ADC Status Register **$YFF70E**



ADSTAT contains information related to the status of a conversion sequence.

SCF — Sequence Complete Flag
0 = Sequence not complete
1 = Sequence complete
SCF is set at the end of the conversion sequence when SCAN is cleared, and at the end of the first conversion sequence when SCAN is set. SCF is cleared when ADCTL1 is written and a new conversion sequence begins.

CCTR[2:0] — Conversion Counter Field
This field reflects the contents of the conversion counter pointer in either four or eight count conversion sequence. The value corresponds to the number of the next result register to be written, and thus indicates which channel is being converted.

CCF[7:0] — Conversion Complete Field
Each bit in this field corresponds to an A/D result register (CCF7 to RSLT7, etc.). A bit is set when conversion for the corresponding channel is complete, and remains set until the result register is read.

RSLT0–RSLT7 — A/D Result Registers **$YFF710–$YFF73E**

The result registers are used to store data after conversion is complete. Each register can be read from three different addresses in the register block. Data format depends on the address from which it is read.

Unsigned Right-Justified Format **$YFF710–$YFF71F**

Conversion result is unsigned right-justified data. Bits [9:0] are used for 10-bit resolution. For 8-bit conversions, bits [7:0] contain data and bits [9:8] are zero. Bits [15:10] always return zero when read.

Signed Left-Justified Format **$YFF720–$YFF72F**

Conversion result is signed left-justified data. Bits [15:6] are used for 10-bit resolution. For 8-bit conversions, bits [15:8] contain data and bits [7:6] are zero. Although the ADC is unipolar, it is assumed that the zero point is halfway between low and high reference when this format is used — for positive input, bit 15 = 0, for negative input, bit 15 = 1. Bits [5:0] always return zero when read.

Unsigned Left-Justified Format **$YFF730–$YFF73F**

Conversion result is unsigned left-justified data. Bits [15:6] are used for 10-bit resolution. For 8-bit conversions, bits [15:8] contain data and bits [7:6] are zero. Bits [5:0] always return zero when read.

8 Flash EEPROM

The MC68F333 MCU contains two flash electrically erasable programmable read-only memory (EEPROM) modules: a 16-Kbyte module and a 48-Kbyte module. These modules serve as nonvolatile, fast-access, electrically erasable and programmable ROM-emulation memory. The modules can contain program code (e.g., operating system kernels and standard subroutines) which must execute at high speed or is frequently executed, or static data which is read frequently. Flash EEPROM can be read as either bytes or words. It is capable of responding to back-to-back IMB accesses to provide two bus cycle (four system clock) access for aligned long words. It can also be programmed to insert up to three wait states to accommodate migration from slower external development memory to onboard flash EEPROM without the need for retiming the system.

The 16-Kbyte flash EEPROM array can be placed on any 16K boundary, and the 48-Kbyte array can be placed in the lower 48 Kbytes of any 64-Kbyte boundary, via writable array base address registers. They can be individually configured to reside in supervisor or unrestricted address space. They can also be programmed to reside in either program space or data space. The flash EEPROM arrays can be configured to appear as a single contiguous memory block.

Either of the flash EEPROM modules can be configured to generate bootstrap information on system reset. Bootstrap information consists of the initial program counter and stack pointer values for the CPU32.

Pulling data pins DATA15 and DATA14 low during reset disables the 16- and 48-Kbyte flash EEPROM modules, respectively, and places them in stop mode.

The flash EEPROM and its control bits are erasable and programmable under software control. Program/erase voltage must be supplied via external V_{FP} pins. Programming is by byte or aligned word. Multiple word programming is supported in test mode only. The flash EEPROM modules support bulk erase only. The module has a minimum program-erase life of 100 cycles.

The flash EEPROM module has hardware interlocks which protect stored data from corruption by accidental enabling of the program/erase voltage to the flash EEPROM array. With the hardware interlocks, inadvertent programming or erasure is highy unlikely.

8.1 Flash EEPROM Register Map

The register map of the flash EEPROM modules follows. All control block registers reside in supervisor space.

Flash EEPROM Register Map

Address	Register	Module
$YFF800	Flash EEPROM Module Configuration Register (FEE1MCR)	16-Kbyte Flash EEPROM
$YFF802	Flash EEPROM Test Register (FEE1TST)	
$YFF804	Flash EEPROM Base Address High Register (FEE1BAH)	
$YFF806	Flash EEPROM Base Address Low Register (FEE1BAL)	
$YFF808	Flash EEPROM Control Register (FEE1CTL)	
$YFF80A	Reserved	
$YFF80C	Reserved	
$YFF80E	Reserved	
$YFF810	Flash EEPROM Bootstrap Word 0 (FEE1BS0)	
$YFF812	Flash EEPROM Bootstrap Word 1 (FEE1BS1)	
$YFF814	Flash EEPROM Bootstrap Word 2 (FEE1BS2)	
$YFF816	Flash EEPROM Bootstrap Word 3 (FEE1BS3)	
$YFF818	Reserved	
$YFF81A	Reserved	
$YFF81C	Reserved	
$YFF81E	Reserved	
$YFF820	Flash EEPROM Module Configuration Register (FEE2MCR)	48-Kbyte Flash EEPROM
$YFF822	Flash EEPROM Test Register (FEE2TST)	
$YFF824	Flash EEPROM Base Address High Register (FEE2BAH)	
$YFF826	Flash EEPROM Base Address Low Register (FEE2BAL)	
$YFF828	Flash EEPROM Control Register (FEE2CTL)	
$YFF82A	Reserved	
$YFF82C	Reserved	
$YFF82E	Reserved	
$YFF830	Flash EEPROM Bootstrap Word 0 (FEE2BS0)	
$YFF832	Flash EEPROM Bootstrap Word 1 (FEE2BS1)	
$YFF834	Flash EEPROM Bootstrap Word 2 (FEE2BS2)	
$YFF836	Flash EEPROM Bootstrap Word 3 (FEE2BS3)	
$YFF838	Reserved	
$YFF83A	Reserved	
$YFF83C	Reserved	
$YFF83E	Reserved	

8.2 Flash EEPROM Control Block

Each flash EEPROM module has a 32-byte control block with five registers to control flash EEPROM module operation: Flash EEPROM module configuration register (FEEMCR), test register (FEETST), array base address registers (FEEBAH and FEEBAL), and the flash EEPROM control register (FEECTL). Configuration information is specified and programmed independently from the contents of the flash EEPROM array. Four additional registers contain bootstrap information if the flash EEPROM is to be used as the bootstrap memory for the system. Control registers are located in supervisor data space.

The control register blocks for the 16-Kbyte and 48-Kbyte flash EEPROM modules start at locations $YFF800 and $YFF820, respectively. The following register descriptions apply to the corresponding register in either control block. References to FEEMCR, for example, apply to both FEE1MCR (in the 16-Kbyte module) and FEE2MCR (in the 48-Kbyte module.)

The registers in the control block have associated flash EEPROM shadow registers which are physically located in a spare flash EEPROM row. During master reset, some of the control registers or fields within them are loaded with default reset information from the shadow registers. For these fields, a reset value of "SB" (shadow bit) is indicated in the register diagram.

The shadow registers are programmed or erased in the same manner as a location in the flash EEPROM array, using the addresses of their corresponding control/configuration registers. When a shadow register is being programmed, the data is not written to the corresponding control register. Data in the shadow register is not copied into the control register until the next master reset.

The contents of the flash EEPROM shadow registers are erased whenever the flash EEPROM array is erased.

After reset, registers in the control block which contain writable bits can be modified. Writes to these registers affect only the writable bits in the registers and not the value that gets programmed into the shadow location. Certain registers are writable only when the LOCK bit in the FEEMCR is disabled, when the STOP bit in the FEEMCR is set, or when the MCU is in supervisor mode. These restrictions are noted in the individual register descriptions.

8.3 Flash EEPROM Array

The base address registers specify the base address of the flash EEPROM array. The user programs the default reset base address. The base address of the 16-Kbyte array must be on a 16-Kbyte boundary; the base array of the 48-Kbyte array must be on a 64-Kbyte boundary. The base address must be set so that the array does not overlap the flash EEPROM control block in the data space memory map. If the array does overlap the control block, accesses to the 32 bytes in the array that overlap are ignored, allowing the flash EEPROM control blocks to remain accessible. If the array overlaps the control block of another module, the results are undetermined.

Accesses to unimplemented locations in the address block are ignored, allowing another internal module or an external device to respond.

8.4 Flash EEPROM Registers

In the following register diagrams, the reset value SB means the bit assumes the value of its associated shadow bit.

FEE1MCR — Flash EEPROM Module Configuration Register **$YFF800**
FEE2MCR **$YFF820**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STOP	FRZ	0	BOOT	LOCK	0	ASPC		WAIT		0	0	0	0	0	0
RESET:															
SB	0	0	SB	SB	0	SB		SB		0	0	0	0	0	0

This register is writable only when the control block is not write-locked (i.e., when LOCK = 0). All fields and bits are set with the values in the shadow register after reset.

STOP — Stop Mode Control

0 = Normal operation

1 = Low-power stop operation

STOP can be set with an external STOP configuration signal or by reset if the STOP shadow bit is set. The array can be reenabled by clearing STOP after reset. If STOP is set during programming or erasing, program/erase voltage is automatically turned off. However, the ENPE control bit remains set. When STOP is cleared, the program/erase voltage is automatically turned back on if ENPE is set.

FRZ — Freeze Mode Control

0 = Disable program/erase voltage while FREEZE is asserted

1 = Allow ENPE bit to turn on the program/erase voltage while FREEZE signal is asserted

BOOT — Boot Control

0 = Flash EEPROM module responds to the bootstrap addresses after reset

1 = Flash EEPROM module does not respond to the bootstrap addresses after reset

On reset, the state of the BOOT bit is forced to the default reset value stored in the shadow flash EEPROM. If BOOT = 0 and STOP = 0, the array responds only to supervisor program space addresses $00000000 to $00000006. After address $00000006 is read, the flash EEPROM module responds only to control block or array addresses.

LOCK — Lock Registers

0 = Write-locking disabled

1 = Write-locked registers protected

If the default reset state of the LOCK is zero, it can be set once after master reset to allow protection of the registers after initialization. Once the LOCK bit is set by software, it cannot be cleared again until after a master reset.

ASPC — Flash EEPROM Array Space

00 = Unrestricted program and data space

01 = Unrestricted program space

10 = Supervisor program and data space

11 = Supervisor program space

ASPC assigns the flash EEPROM array to supervisor or user space, and to program or data space. The field can be written to only if LOCK = 0 and STOP = 1.

WAIT — Wait States

The WAIT field specifies the number of wait states inserted during accesses to the flash EEPROM module. A wait state has the duration of one system clock cycle. This field affects both control block and array access.

WAIT[1:0]	Wait States	Clocks/Transfer
00	1	3
01	2	4
10	3	5
11	0	2

The value of the WAIT field is compatible with the lower two bits of the DSACK field in the SCIM chip select option registers.

FEE1TST — Flash EEPROM Test Register **$YFF802**
FEE2TST **$YFF822**

This register is used for factory test purposes only.

FEE1BAH — Flash EEPROM Base Address High Register **$YFF804**
FEE2BAH **$YFF824**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Flash EEPROM Base Address (high-order bits)															
RESET:															
SB															

FEEBAH contains the 16 high-order bits of the flash EEPROM array base address. After reset, if LOCK = 0 and STOP = 1, software can write to FEEBAH and FEEBAL to relocate the flash EEPROM array.

FEE1BAL — Flash EEPROM Base Address Low Register **$YFF806**
FEE2BAL **$YFF826**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Base Address (low-order bits)		0	0	0	0	0	0	0	0	0	0	0	0	0	0
RESET:															
SB		0	0	0	0	0	0	0	0	0	0	0	0	0	0

FEE1BAL contains the two low-order bits of the 16-Kbyte flash EEPROM array base address. After reset, if LOCK = 0 and STOP = 1, software can write to FEEBAH and FEEBAL to relocate the flash EEPROM array. All bits of FEE2BAL (in the 48-Kbyte module) are always zero.

FEE1CTL — Flash EEPROM Control Register **$YFF808**
FEE2CTL **$YFF828**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	VFPE	ERAS	LAT	ENPE
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FEECTL contains the bits needed to control the programming and erasure of the flash EEPROM. This register is accessible in supervisor mode only.

VFPE — Verify Program/Erase

0 = Normal read cycles

1 = Invoke program verify circuit

This bit invokes a special program verify circuit. During programming sequences (ERAS = 0), VFPE is used in conjunction with the LAT bit to determine when programming of a location is complete. If VFPE and LAT are both set, a bit-wise exclusive-OR of the latched data occurs with the data currently in the location being programmed, when any valid flash EEPROM location is read. If the location is completely programmed, a value of zero is read. Any other value obtained from the read indicates that the location is not fully programmed. When VFPE is negated, normal reads of valid flash EEPROM locations occur.

ERAS — Erase Control

0 = Flash EEPROM configured for programming

1 = Flash EEPROM configured for erasure

Asserting ERAS causes all locations in the array and all flash EEPROM control bits in the control block to be configured for erasure at the same time.

When the LAT bit is set, ERAS also determines whether a read returns the value of the addressed location (ERAS = 1) or the location being programmed (ERAS = 0).

The value of ERAS cannot be changed if the program/erase voltage is turned on (ENPE = 1).

LAT — Latch Control

0 = Programming latches disabled

1 = Programming latches enabled

When LAT = 0, the flash EEPROM address and data buses are connected to the IMB address and data buses and the flash EEPROM is configured for normal reads. When LAT is asserted, the flash EEPROM address and data buses are connected to parallel internal latches and the flash EEPROM array is configured for programming or erasure. If LAT = 1 and the next write is to a valid flash EEPROM location, the programming latches latch the address and data. (Notice that the write immediately following the setting of the LATCH bit must be a write to the location being programmed and not a write to set or clear a bit in a control register. Otherwise the control register shadow bits are programmed.)

The value of LAT cannot be changed if the program/erase voltage is turned on (ENPE = 1).

ENPE — Enable Programming/Erase

0 = Disable program/erase voltage

1 = Apply program/erase voltage to flash EEPROM

ENPE can be asserted only after LAT has been asserted and a write to the data and address latches has occurred. (ENPE remains negated if these conditions are not met.) With ENPE asserted, the LAT, VFPE, and ERAS bits cannot be changed, and attempts to read a flash EEPROM array location in the flash EEPROM module are ignored.

FEE1BS[3:0] — Flash EEPROM Bootstrap Words $YFF810 – $YFF817
FEE2BS[3:0] $YFF830 – $YFF837

These words may contain bootstrap information for the processor. If BOOT = 1 in the FEEMCR, these words are mapped to the following addresses:

Word	Address
FEEBS0	$00000000
FEEBS1	$00000002
FEEBS2	$00000004
FEEBS3	$00000006

When BOOT = 1, these words respond only to supervisor program space accesses. When BOOT = 0, they respond only to supervisor data space accesses. FEEBS[3:0] can be read as registers at any time, but they respond to addresses $00 to $06 only when BOOT = 1.

8.5 Flash EEPROM Operation

The following paragraphs describe the operation of the flash EEPROM module during reset, system boot, normal operation, and while it is being programmed or erased.

8.5.1 Reset Operation

Reset initializes all register bits to their reset values. Some of these reset values are programmable by the user and are contained in flash EEPROM shadow registers. If the state of the STOP shadow bit is zero, the STOP bit in the FEEMCR is cleared during reset and the array responds normally to the bootstrap address range and the flash EEPROM array base address. If the STOP shadow bit is one, the STOP bit in the FEEMCR is set and the flash EEPROM array is disabled until the STOP bit is cleared by software. It will not respond to the bootstrap address range or the flash EEPROM array base address in FEEBAH and FEEBAL, allowing an external device to respond to the flash EEPROM array's address space or bootstrap information. Since the erased state of the shadow bits is one, erased flash EEPROM modules come out of reset in STOP mode.

8.5.2 Bootstrap Operation

After reset, the CPU begins bootstrap operation by fetching initial values for its internal registers from special bootstrap word addresses $00000000 through $00000006. If BOOT = 0 and STOP = 0 in the FEEMCR, the flash EEPROM module is configured to recognize these addresses after a reset and provide this information from bootstrap registers in the flash EEPROM control block. The information in these registers is programmed by the user.

8.5.3 Normal Operation

The flash EEPROM module allows a byte or aligned-word read or write in one bus cycle. Long-word reads or writes require an additional bus cycle.

The module checks function codes to verify access privileges. All control block addresses must be in supervisor data space. Array accesses are defined by the state of ASPC in FEEMCR. Access time is governed by the WAIT field in FEEMCR. When the flash EEPROM module is

configured for normal operation, the array responds to read accesses only; write operations are ignored.

Accesses to any address in the address block defined by FEEBAH and FEEBAL which does not fall within the array are ignored, allowing external devices to adjoin flash EEPROM arrays which do not entirely fill the entire address space specified by FEEBAH, FEEBAL.

8.5.4 Program/Erase Operation

Erasing a flash EEPROM bit returns it to its unprogrammed state of one. Programming causes the bit to change from one to zero. The flash EEPROM module can be programmed and erased a maximum of 100 cycles. Programming or erasing the flash EEPROM consists of a series of control register writes and a write to a set of programming latches. The same programming procedure is used to program both array locations and control registers which contain flash EEPROM bits. Programming is restricted to a single byte or aligned word at a time. The entire module (the flash EEPROM array and the shadow register bits) is erased at the same time.

Specifications for programming and erasing the flash EEPROM modules are shown below on the next page.

8.5.4.1 Intelligent Programming and Erasing

The instructions below are for intelligent programming and erasing of the flash EEPROM. This involves verifying the flash EEPROM array as it is being programmed or erased to ensure accurate results and provide the longest possible life expectancy for the module. The user must stop the programming or erase sequence at periods of t_{ppulse} or t_{epulse} to determine whether the sequence was successful. These values must be recalculated after each pulse for optimum performance. After the location reaches the proper value, it must continue to be programmed ($t_{pmargin}$) or erased ($t_{emargin}$) for a short period to ensure that it remains programmed or erased.

Flash EEPROM Specifications

Symbol	Meaning	Value
V_{dd}	Operating voltage Normal operation Program/erase	 +5.0 v 10% +5.0 v 10%
f	Operating frequency	DC to 16.78 MHz
T	Operating temperature range Normal Program/erase	 –40 to +85 C 0 to +85 C
V_{fp}	External program/erase voltage Normal Program/erase	 V_{dd} 12.0 v 5%
t_{vprog}	Program recovery time	10 µs
$t_{progmax}$	Maximum allowable programming time	2 ms
t_{pmin}	Minimum program pulse	10 µs
t_{ppulse} (new)	New value of programming pulse	t_{ppulse} (old) + t_{pdelta}
t_{pdelta} (recommended)	Amount to increment t_{ppulse}	5 µs
$t_{pmargin}$	Final programming pulse after location reaches programmed value	See below*
t_{verase}	Erase recovery time	10 ms
$t_{erasemax}$	Maximum allowable erase time	25 s
t_{emin}	Minimum erase pulse	200 ms
t_{epulse} (new)	New value of erasing pulse	t_{epulse} (old) + t_{edelta}
t_{edelta} (recommended)	Amount to increment t_{epulse}	300 ms
$t_{emargin}$	Final erasing pulse after array reaches erased value	25% • total erase time

*If total programming time exceeds 100 µs, then $t_{pmargin}$ = 50% • total programming time. Otherwise, $t_{pmargin}$ = t_{ppulse} (final) + t_{pdelta}

8.5.4.2 Programming Sequence

Intelligent programming of the flash EEPROM consists of these steps:

1. Turn on V_{fp} (apply V_{fp} to V_{FPE16K} or V_{FPE48K} pin).
2. Clear ERAS and set LAT and VFPE bits in FEECTL to set program mode, enable programming address and data latches, and invoke special verification read circuitry. Set initial value of t_{ppulse} to t_{pmin}.
3. Write new data to the desired address. This causes the address and data of the location to be programmed to be latched in the programming latches.
4. Set ENPE to apply programming voltage.
5. Delay long enough for one programming pulse to occur (t_{ppulse}).
6. Clear ENPE to remove programming voltage.
7. Delay while high voltage is turning off (t_{vprog}).

8. Read the EEPROM location just programmed. If the value read is all zeros, proceed to step 9. If not, calculate a new value for t_{ppulse} and repeat steps 4 through 7 until either the location is verified or the total programming time ($t_{progmax}$) has been exceeded. If $t_{progmax}$ has been exceeded, the location may be bad and should not be used.
9. If the flash EEPROM location is programmed, calculate $t_{pmargin}$ and repeat steps 4 through 7. If the flash EEPROM location does not remain programmed, the flash EEPROM location is bad.
10. Clear VFPE and LAT.
11. If there are more locations to program, repeat steps 2 through 10.
12. Turn off V_{fp} (reduce voltage on V_{FPE16K} or V_{FPE48K} pin to V_{dd}).
13. Read the entire array to verify that all locations are correct. If any locations are incorrect, the flash EEPROM module is bad.

8.5.4.3 Erasing Sequence

Intelligent erasing of the flash EEPROM consists of these steps:

1. Turn on V_{fp} (apply V_{fp} to V_{FPE16K} or V_{FPE48K} pin).
2. Set LAT, VFPE, and ERAS bits to configure flash EEPROM for erasure. Set initial value to t_{epulse} to t_{emin}.
3. Write to any valid address in the control block or array. This allows the erase voltage to be turned on. The data written and the address written to are of no consequence.
4. Set ENPE to apply programming voltage.
5. Delay long enough for one erase pulse to occur (t_{epulse}).
6. Clear ENPE to remove programming voltage.
7. Delay while high voltage is turning off (t_{vprog}).
8. Read the entire array and control block to ensure that flash EEPROM is erased.
9. If all of the flash EEPROM locations are not erased, calculate a new value for t_{epulse} and repeat steps 4 through 8 until either the remaining locations are erased or the maximum erase time (t_{erase}) has been exceeded. If t_{erase} has been exceeded, the location may be bad and should not be used.
10. If all flash EEPROM locations are erased, calculate $t_{emargin}$ and repeat steps 4 through 8. If all flash EEPROM locations do not remain erased, the flash EEPROM module may be bad.
11. Clear LAT, ERAS, and VFPE to allow normal access to the flash EEPROM.
12. Turn off V_{fp} (reduce voltage on V_{FPE16K} or V_{FPE48K} pin to V_{dd}).

9 Standby RAM Module (SRAM)

SRAM contains a 512-byte array of fast (two bus cycle) static RAM, which is especially useful for system stacks and variable storage. The SRAM can be mapped to any 512-byte boundary in the address map, but must not overlap the module control registers. (Overlap makes the registers inaccessible.) Data can be read or written in bytes, word, or long words. SRAM is powered by V_{DD} in normal operation. During power-down, the RAM contents are maintained by power on the standby voltage pin $V_{STBYRAM}$. Power switching between sources is automatic.

9.1 SRAM Registers

There are four SRAM control registers: the RAM module configuration register (SRAMMCR), the RAM test register (SRAMTST), and the RAM array base address registers (SRAMBAH and SRAMBAL). There is an 8-byte minimum register block size for the module. Unimplemented register addresses are read as zeros, and writes have no effect.

Access to the SRAM array is controlled by the RASP field in the SRAMMCR. SRAM responds to both program and data space accesses. This allows code to be executed from RAM and permits use of program counter relative addressing mode for operand fetches from the array.

SRAMMCR — SRAM Module Configuration Register **$YFFB00**

15				11		9	8	7	6	5	4	3	2	1	0
STOP	0	0	0	RLCK	0	RASP		NOT USED							
RESET:															
1	0	0	0	0	0	1	1								

SRAMMCR is used to determine whether the RAM is in STOP mode or normal mode. It is also used to determine in which space the array resides, and controls access to the base array registers. Reads of unimplemented bits always return zeros. Writes do not affect unimplemented bits.

STOP — Stop Control

0 = RAM array operates normally.

1 = RAM array enters low-power stop mode.

This bit controls whether the RAM array is in stop mode or normal operation. Reset state is one, causing the module to enter low-power stop mode. In stop mode, the array retains its contents, but cannot be read or written by the CPU.

RLCK — RAM Base Address Lock

0 = SRAM base address registers are writable from IMB.

1 = SRAM base address registers are locked.

RLCK defaults to zero on reset; it is one-time writable to one.

RASP[1:0] — RAM Array Space Field

This field locates the SRAM array in supervisor or user space and in program or data space.

RASP	Space
00	Unrestricted Program and Data
01	Unrestricted Program
10	Supervisor Program and Data
11	Supervisor Program

SRAMTST — SRAM Test Register **$YFFB02**

SRAMTST is used during factory test of the module. Accesses to SRAMTST must be made while the MCU is in test mode.

SRAMBAH — SRAM Array Base Address Register High **$YFFB04**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 31	ADDR 30	ADDR 29	ADDR 28	ADDR 27	ADDR 26	ADDR 25	ADDR 24	ADDR 23	ADDR 22	ADDR 21	ADDR 20	ADDR 19	ADDR 18	ADDR 17	ADDR 16
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRAMBAL — SRAM Array Base Address Register Low **$YFFB06**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 15	ADDR 14	ADDR 13	ADDR 12	ADDR 11	ADDR 10	ADDR9	ADDR8	ADDR7	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SRAMBAH and SRAMBAL are used to specify an SRAM base address in the system memory map. They can only be written when RLCK in SRAMMCR is cleared and STOP is set. This prevents accidental remapping of the array.

9.2 SRAM Operation

There are five operating modes, as follows.

a. The RAM module is in normal mode when powered by V_{DD}. The array can be accessed by byte, word, or long word. A byte or aligned word (high-order byte is at an even address) access only takes one bus cycle or two system clocks (also referred to as –1 wait states). A long word or misaligned word access requires two bus cycles.

b. Standby mode is intended to preserve RAM contents when V_{DD} is removed. SRAM contents are maintained by V_{STBY}. Circuitry within the SRAM module switches to the higher of V_{DD} or V_{STBY} with no loss of data. When SRAM is powered by V_{STBY}, access to the array is not guaranteed. If standby operation is not desired, connect the V_{STBY} pin to V_{SS}.

c. Reset mode allows the CPU to complete the current bus cycle before resetting. When a synchronous reset occurs while a byte or word SRAM access is in progress, the access will be completed. If reset occurs during the first word access of a long-word operation, only the first word access will be completed. If reset occurs during the second word access of a long word operation, the entire access will be completed. Data being read from or written to the RAM may be corrupted by an asynchronous reset.

d. The RAM module has special test functions that are used during factory test of the MCU.

e. Writing the STOP bit of RAMMCR causes the SRAM module to enter stop mode. The RAM array is disabled (which allows external logic to decode SRAM addresses, if necessary), but all data is retained. If V_{DD} falls below V_{STBY}, internal circuitry switches to V_{STBY}, as in standby mode. Stop mode is exited by clearing the STOP bit.

10 Standby Ram with TPU Emulation (TPURAM)

The TPURAM module contains a 3.5 Kbyte array of fast (two bus cycle) static RAM, which is especially useful for system stacks and variable storage. Alternately, it can be used by the TPU as emulation RAM for new timer algorithms. The TPURAM can be mapped to any 4-Kbyte boundary in the address map, but must not overlap the module control registers. (Overlap makes the registers inaccessible.) Data can be read or written in bytes, word, or long words. The RAM is powered by V_{DD} in normal operation. During power-down, the RAM contents are maintained by power on the standby voltage pin $V_{STBYRAM}$. Power switching between sources is automatic.

10.1 TPURAM Register Block

There are three TPURAM control registers: the RAM module configuration register (TRAMMCR), the RAM test register (TRAMTST), and the RAM array base address registers (TRAMBAR).

There is an 8-byte minimum register block size for the module. Unimplemented register addresses are read as zeros, and writes have no effect.

10.2 TPURAM Registers

Access to the TPURAM array is controlled by the RASP field in the TRAMMCR. TPURAM responds to both program and data space accesses. This allows code to be executed from RAM and permits use of program counter relative addressing mode for operand fetches from the array.

TRAMMCR — RAM Module Configuration Register **$YFFB00**

15				11		9	8	7	6	5	4	3	2	1	0
STOP	0	0	0	0	0	0	RASP	NOT USED							
RESET:															
0	0	0	0	0	0	0	1								

STOP — Stop Control

0 = RAM array operates normally.

1 = RAM array enters low-power stop mode.

This bit controls whether the RAM array is in stop mode or normal operation. Reset state is zero, for normal operation. In stop mode, the array retains its contents, but cannot be read or written by the CPU.

RASP[1:0] — RAM Array Space Field

0 = TPURAM array is placed in unrestricted space

1 = TPURAM array is placed in supervisor space.

TRAMTST — RAM Test Register **$YFFB02**

TRAMTST is used for factory test of the TPURAM module.

TRAMBAR — TPURAM Base Address and Status Register **$YFFB04**

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	0	0	RAMDS
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADDR[23:11] — RAM Array Base Address

These bits specify address lines ADDR23–ADDR11 of the base address of the RAM array when enabled.

RAMDS — RAM Array disabled

0 = RAM array is enabled

1 = RAM array is disabled

The RAM array is disabled by internal logic after a master reset. Writing a valid base address to the RAM array base address field (bits [15:3]) automatically clears RAMDS, enabling the RAM array.

10.3 TPURAM Operation

There are six operating modes, as follows.

a. The RAM module is in normal mode when powered by V_{DD}. The array can be accessed by byte, word, or long word. A byte or aligned word (high-order byte is at an even address) access only takes one bus cycle or two system clocks. A long word or misaligned word access requires two bus cycles.

b. Standby mode is intended to preserve RAM contents when V_{DD} is removed. SRAM contents are maintained by V_{STBY}. Circuitry within the SRAM module switches to the higher of V_{DD} or V_{STBY} with no loss of data. When SRAM is powered by V_{STBY}, access to the array is not guaranteed. If standby operation is not desired, connect the V_{STBY} pin to V_{SS}.

c. Reset mode allows the CPU to complete the current bus cycle before resetting. When a synchronous reset occurs while a byte or word SRAM access is in progress, the access will be completed. If reset occurs during the first word access of a long-word operation, only the first word access will be completed. If reset occurs during the second word access of a long word operation, the entire access will be completed. Data being read from or written to the RAM may be corrupted by asynchronous reset.

d. The RAM module has special test functions that are used during factory test of the MCU.

e. Writing the STOP bit of RAMMCR causes the SRAM module to enter stop mode. The RAM array is disabled (which allows external logic to decode SRAM addresses, if necessary), but all data is retained. If V_{DD} falls below V_{STBY}, internal circuitry switches to V_{STBY}, as in standby mode. Stop mode is exited by clearing the STOP bit.

f. The TPURAM array may be used as the microcode control store for the TPU module. This mode of operation is selected from within the TPU. Refer to development support in the TPU manual for more information.

The TPU is connected to the RAM via a dedicated bus. While in emulation mode, the access timing of the TPURAM module matches the timing of the TPU microinstruction ROM to ensure accurate emulation. Normal accesses via the IMB are inhibited and the control registers have no effect, allowing external RAM to emulate the TPURAM at the same addresses.

Literature Distribution Centers:
USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036.
EUROPE: Motorola Ltd.; European Literature Centre; 88 Tanners Drive, Blakelands, Milton Keynes, MK14 5BP, England.
JAPAN: Nippon Motorola Ltd.; 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141, Japan.
ASIA PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong.

MOTOROLA

A31036 PRINTED IN USA 2/92 IMPERIAL LITHO 84100 15,000 MCU YCACAA

MC68F333TS/D