



Welcome to [E-XFL.COM](#)

What is "[Embedded - Microcontrollers](#)"?

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

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Obsolete
Core Processor	CPU32
Core Size	32-Bit Single-Core
Speed	16MHz
Connectivity	EBI/EMI, SCI, SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	15
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	132-BQFP Bumpered
Supplier Device Package	132-PQFP (24.13x24.13)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68332acfc16b1

1.3 Pin Assignments

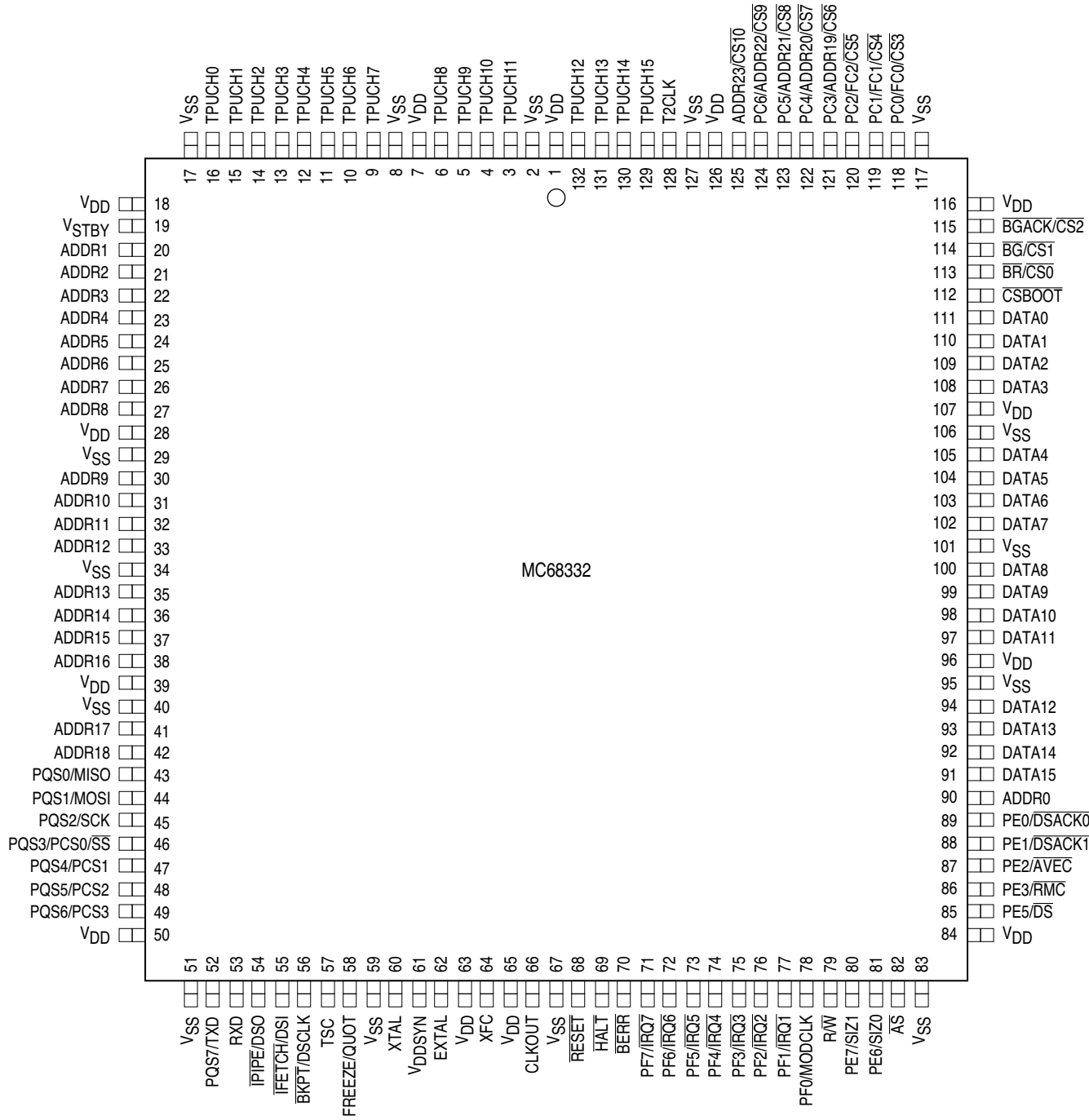


Figure 2 MC68332 132-Pin QFP Pin Assignments

Table 7 SIM Address Map (Continued)

Access	Address	15	8 7	0
S	\$YFFA56	CHIP-SELECT OPTION 2 (CSOR2)		
S	\$YFFA58	CHIP-SELECT BASE 3 (CSBAR3)		
S	\$YFFA5A	CHIP-SELECT OPTION 3 (CSOR3)		
S	\$YFFA5C	CHIP-SELECT BASE 4 (CSBAR4)		
S	\$YFFA5E	CHIP-SELECT OPTION 4 (CSOR4)		
S	\$YFFA60	CHIP-SELECT BASE 5 (CSBAR5)		
S	\$YFFA62	CHIP-SELECT OPTION 5 (CSOR5)		
S	\$YFFA64	CHIP-SELECT BASE 6 (CSBAR6)		
S	\$YFFA66	CHIP-SELECT OPTION 6 (CSOR6)		
S	\$YFFA68	CHIP-SELECT BASE 7 (CSBAR7)		
S	\$YFFA6A	CHIP-SELECT OPTION 7 (CSOR7)		
S	\$YFFA6C	CHIP-SELECT BASE 8 (CSBAR8)		
S	\$YFFA6E	CHIP-SELECT OPTION 8 (CSOR8)		
S	\$YFFA70	CHIP-SELECT BASE 9 (CSBAR9)		
S	\$YFFA72	CHIP-SELECT OPTION 9 (CSOR9)		
S	\$YFFA74	CHIP-SELECT BASE 10 (CSBAR10)		
S	\$YFFA76	CHIP-SELECT OPTION 10 (CSOR10)		
	\$YFFA78	NOT USED		NOT USED
	\$YFFA7A	NOT USED		NOT USED
	\$YFFA7C	NOT USED		NOT USED
	\$YFFA7E	NOT USED		NOT USED

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

3.2 System Configuration and Protection

This functional block provides configuration control for the entire MCU. It also performs interrupt arbitration, bus monitoring, and system test functions. MCU system protection includes a bus monitor, a HALT monitor, a spurious interrupt monitor, and a software watchdog timer. These functions have been made integral to the microcontroller to reduce the number of external components in a complete control system.

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 and periodic interrupt timer counters continue to run.
- 1 = When FREEZE is asserted, the software watchdog and periodic interrupt timer counters are disabled, preventing interrupts during software debug.

FRZBM — Freeze Bus Monitor Enable

- 0 = When FREEZE is asserted, the bus monitor continues to operate.
- 1 = When FREEZE is asserted, the bus monitor is disabled.

SLVEN — Factory Test Mode Enabled

- This bit is a read-only status bit that reflects the state of DATA11 during reset.
- 0 = IMB is not available to an external master.
 - 1 = An external bus master has direct access to the IMB.

SHEN[1:0] — Show Cycle Enable

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

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

SUPV — Supervisor/Unrestricted Data Space

- The SUPV bit places the SIM global registers in either supervisor or user data space.
- 0 = Registers with access controlled by the SUPV bit are accessible from either the user or supervisor privilege level.
 - 1 = Registers with access controlled by the SUPV bit are restricted to supervisor access only.

MM — Module Mapping

- 0 = Internal modules are addressed from \$7FF000 –\$7FFFFFF.
- 1 = Internal modules are addressed from \$FFF000 –\$FFFFFF.

IARB[3:0] — Interrupt Arbitration Field

Each module that can generate interrupt requests has an interrupt arbitration (IARB) field. Arbitration between interrupt requests of the same priority is performed by serial contention between IARB field bit values. Contention must take place whenever an interrupt request is acknowledged, even when there is only a single pending request. An IARB field must have a non-zero value for contention to take place. If an interrupt request from a module with an IARB field value of %0000 is recognized, the CPU processes a spurious interrupt exception. Because the SIM routes external interrupt requests to the CPU, the SIM IARB field value is used for arbitration between internal and external interrupts of the same priority. The reset value of IARB for the SIM is %1111, and the reset IARB value for all other modules is %0000, which prevents SIM interrupts from being discarded during initialization.

3.2.3 Bus Monitor

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

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

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

3.2.4 Halt Monitor

The halt monitor responds to an assertion of \overline{HALT} on the internal bus. A flag in the reset status register (RSR) indicates that the last reset was caused by the halt monitor. The halt monitor reset can be inhibited by the HME bit in the SYPCR.

3.2.5 Spurious Interrupt Monitor

The spurious interrupt monitor issues \overline{BERR} if no interrupt arbitration occurs during an interrupt-acknowledge cycle.

3.2.6 Software Watchdog

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

SWSR —Software Service Register								\$YFFA27	
15	8	7	6	5	4	3	2	1	0
NOT USED								0	0
RESET:									
								0	0

Register shown with read value

Perform a software watchdog service sequence as follows:

- Write \$55 to SWSR.
- Write \$AA to SWSR.

Both writes must occur before time-out in the order listed, but any number of instructions can be executed between the two writes.

The watchdog clock rate is affected by SWP and SWT in SYPCR. When SWT[1:0] are modified, a watchdog service sequence must be performed before the new time-out period takes effect.

The reset value of SWP is affected by the state of the MODCLK pin on the rising edge of reset, as shown in the following table.

MODCLK	SWP
0	1
1	0

3.2.7 Periodic Interrupt Timer

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

PICR — Periodic Interrupt Control Register

\$YFFA22

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

RESET:

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

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

PIRQL[2:0] — Periodic Interrupt Request Level

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

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 SIM 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	0
0	0	0	0	0	0	0	PTP	PITM	

RESET:

0 0 0 0 0 0 0 MODCLK 0 0 0 0 0 0 0 0

The 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

0 = Periodic timer clock not prescaled

1 = Periodic timer clock prescaled by a value of 512

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

PITM[7:0] — Periodic Interrupt Timing Modulus Field

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

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

where

PIT Period = Periodic interrupt timer period

PITM = Periodic interrupt timer register modulus (PITR[7:0])

EXTAL Frequency = Crystal frequency

Prescale = 512 or 1 depending on the state of the PTP bit in the PITR

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.

3.4.11 Misaligned Operands

CPU32 processor architecture uses a basic operand size of 16 bits. An operand is misaligned when it overlaps a word boundary. This is determined by the value of ADDR0. When ADDR0 = 0 (an even address), the address is on a word and byte boundary. When ADDR0 = 1 (an odd address), the address is on a byte boundary only. A byte operand is aligned at any address; a word or long-word operand is misaligned at an odd address. The CPU32 does not support misaligned operand transfers.

The largest amount of data that can be transferred by a single bus cycle is an aligned word. If the MCU transfers a long-word operand via 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.

3.4.12 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.

Table 11 Operand Alignment

Transfer Case	SIZ1	SIZ0	ADDR0	DSACK1	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) ²	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	1	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	1	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 or long-word transfers.

3.5 Chip Selects

Typical microcontrollers require additional hardware to provide external chip-select signals. Twelve independently programmable chip selects provide fast two-cycle access to external memory or peripherals. Address block sizes of 2 Kbytes to 1 Mbyte can be selected.

$\overline{\text{AVEC}}$ — 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 usually 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 $\overline{\text{AVEC}}$ field is set to one, the chip select automatically generates an $\overline{\text{AVEC}}$ in response to the interrupt cycle. Otherwise, the vector must be supplied by the requesting device.

The $\overline{\text{AVEC}}$ bit must not be used in synchronous mode, as autovector response timing can vary because of ECLK synchronization.

3.5.5 Port C Data Register

Bit values in port C determine the state of chip-select pins used for discrete output. When a pin is assigned as a discrete output, the value in this register appears at the output. This is a read/write register. Bit 7 is not used. Writing to this bit has no effect, and it always returns zero when read.

PORTC — Port C Data Register

\$YFFA41

15	8	7	6	5	4	3	2	1	0
NOT USED	0	PC6	PC5	PC4	PC3	PC2	PC1	PC0	
RESET:	0	1	1	1	1	1	1	1	1

3.6 General-Purpose Input/Output

SIM pins can be configured as two general-purpose I/O ports, E and F. The following paragraphs describe registers that control the ports.

PORTE0, PORTE1 —Port E Data Register

\$YFFA11, \$YFFA13

15	8	7	6	5	4	3	2	1	0
NOT USED	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	
RESET:	U	U	U	U	U	U	U	U	U

A write to the port E data register is stored in the internal data latch and, if any port E pin is configured as an output, the value stored for that bit is driven on the pin. A read of the port E 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.

The port E data register is a single register that can be accessed in two locations. When accessed at \$YFFA11, the register is referred to as PORTE0; when accessed at \$YFFA13, the register is referred to as PORTE1. The register can be read or written at any time. It is unaffected by reset.

DDRE — Port E Data Direction Register

\$YFFA15

15	8	7	6	5	4	3	2	1	0
NOT USED	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0	
RESET:	0	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.

PEPAR — Port E Pin Assignment Register

\$YFFA17

15	8	7	6	5	4	3	2	1	0
NOT USED		PEPA7	PEPA6	PEPA5	PEPA4	PEPA3	PEPA2	PEPA1	PEPA0

RESET:

DATA8 DATA8 DATA8 DATA8 DATA8 DATA8 DATA8 DATA8

The bits in this register control the function of each port E pin. Any bit set to one configures the corresponding pin as 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.

Data bus bit 8 controls the state of this register following reset. If DATA8 is set to one during reset, the register is set to \$FF, which defines all port E pins as bus control signals. If DATA8 is cleared to zero during reset, this register is set to \$00, configuring all port E pins as I/O pins.

Any bit cleared to zero defines the corresponding pin to be an I/O pin. Any bit set to one defines the corresponding pin to be a bus control signal.

Table 16 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	$\overline{DSACK0}$

PORTF0, PORTF1 — Port F Data Register

\$YFFA19, \$YFFA1B

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

RESET:

U U U U U U U U

The 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 the port F 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.

The port F data register is a single register that can be accessed in two locations. When accessed at \$YFFA19, the register is referred to as PORTF0; when accessed at \$YFFA1B, the register is referred to as PORTF1. The register can be read or written at any time. It is unaffected by reset.

DDRF — Port F Data Direction Register

\$YFFA1D

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

RESET:

0 0 0 0 0 0 0 0

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.

Table 18 Reset Mode Selection

DATA0	CSBOOT 16-Bit	CSBOOT 8-Bit
DATA1	CS0 CS1 CS2	BR BG BGACK
DATA2	CS3 CS4 CS5	FC0 FC1 FC2
DATA3 DATA4 DATA5 DATA6 DATA7	CS6 CS[7:6] CS[8:6] CS[9:6] CS[10:6]	ADDR19 ADDR[20:19] ADDR[21:19] ADDR[22:19] ADDR[23:19]
DATA8	DSACK0, DSACK1, AVEC, DS, AS, SIZ[1:0]	PORTE
DATA9	IRQ[7:1] MODCLK	PORTF
DATA11	Test Mode Disabled	Test Mode Enabled
MODCLK	VCO = System Clock	EXTAL = System Clock
BKPT	Background Mode Disabled	Background Mode Enabled

3.7.2 Functions of Pins for Other Modules During Reset

Generally, pins associated with modules other than the SIM default to port functions, and input/output ports are set to input state. This is accomplished by disabling pin functions in the appropriate control registers, and by clearing the appropriate port data direction registers. Refer to individual module sections in this manual for more information. The following table is a summary of module pin function out of reset.

Table 19 Module Pin Functions

Module	Pin Mnemonic	Function
CPU32	DSI/IFETCH	DSI/IFETCH
	DSO/IPIPE	DSO/IPIPE
	BKPT/DSCLK	BKPT/DSCLK
GPT	PGP7/IC4/OC5	Discrete Input
	PGP[6:3]/OC[4:1]	Discrete Input
	PGP[2:0]/IC[3:1]	Discrete Input
	PAI	Discrete Input
	PCLK	Discrete Input
	PWMA, PWMB	Discrete Output
QSM	PQS7/TXD	Discrete Input
	PQS[6:4]/PCS[3:1]	Discrete Input
	PQS3/PCS0/SS	Discrete Input
	PQS2/SCK	Discrete Input
	PQS1/MOSI	Discrete Input
	PQS0/MISO	Discrete Input
	RXD	RXD

4 Central Processor Unit

Based on the powerful MC68020, the CPU32 processing module provides enhanced system performance and also uses the extensive software base for the Motorola M68000 family.

4.1 Overview

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. Special instructions, such as table lookup and interpolate and low-power stop, support the specific requirements of controller applications. Also included is the background debugging 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.

4.2 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 non-privileged 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.

4.6 Instruction Set Summary

Table 20 Instruction Set Summary

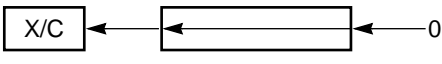
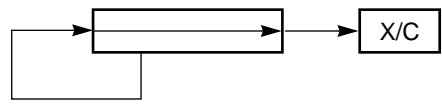
Instruction	Syntax	Operand Size	Operation
ABCD	Dn, Dn – (An), – (An)	8 8	Source ₁₀ + Destination ₁₀ + X ⇒ Destination
ADD	Dn, <ea> <ea>, Dn	8, 16, 32 8, 16, 32	Source + Destination ⇒ Destination
ADDA	<ea>, An	16, 32	Source + Destination ⇒ Destination
ADDI	#<data>, <ea>	8, 16, 32	Immediate data + Destination ⇒ Destination
ADDQ	# <data>, <ea>	8, 16, 32	Immediate data + Destination ⇒ Destination
ADDX	Dn, Dn – (An), – (An)	8, 16, 32 8, 16, 32	Source + Destination + X ⇒ Destination
AND	<ea>, Dn Dn, <ea>	8, 16, 32 8, 16, 32	Source • Destination ⇒ Destination
ANDI	# <data>, <ea>	8, 16, 32	Data • Destination ⇒ Destination
ANDI to CCR	# <data>, CCR	8	Source • CCR ⇒ CCR
ANDI to SR1 ¹	# <data>, SR	16	Source • SR ⇒ SR
ASL	Dn, Dn # <data>, Dn i	8, 16, 32 8, 16, 32 16	
ASR	Dn, Dn # <data>, Dn i	8, 16, 32 8, 16, 32 16	
Bcc	label	8, 16, 32	If condition true, then PC + d ⇒ PC
BCHG	Dn, <ea> # <data>, <ea>	8, 32 8, 32	(bit number) of destination ⇒ Z ⇒ bit of destination
BCLR	Dn, <ea> # <data>, <ea>	8, 32 8, 32	(bit number) of destination 0 ⇒ bit of destination
BGND	none	none	If background mode enabled, then enter background mode, else format/vector ⇒ – (SSP); PC ⇒ – (SSP); SR ⇒ – (SSP); (vector) ⇒ PC
BKPT	# <data>	none	If breakpoint cycle acknowledged, then execute returned operation word, else trap as illegal instruction
BRA	label	8, 16, 32	PC + d ⇒ PC
BSET	Dn, <ea> # <data>, <ea>	8, 32 8, 32	(bit number) of destination ⇒ Z; 1 ⇒ bit of destination
BSR	label	8, 16, 32	SP – 4 ⇒ SP; PC ⇒ (SP); PC + d ⇒ PC
BTST	Dn, <ea> # <data>, <ea>	8, 32 8, 32	(bit number) of destination ⇒ Z
CHK	<ea>, Dn	16, 32	If Dn < 0 or Dn > (ea), then CHK exception
CHK2	<ea>, Rn	8, 16, 32	If Rn < lower bound or Rn > upper bound, then CHK exception
CLR	i	8, 16, 32	0 ⇒ Destination
CMP	<ea>, Dn	8, 16, 32	(Destination – Source), CCR shows results
CPMA	<ea>, An	16, 32	(Destination – Source), CCR shows results
CMPI	# <data>, <ea>	8, 16, 32	(Destination – Data), CCR shows results
CMPM	(An) +, (An) +	8, 16, 32	(Destination – Source), CCR shows results
CMP2	<ea>, Rn	8, 16, 32	Lower bound ≤ Rn ≤ Upper bound, CCR shows result

Table 20 Instruction Set Summary(Continued)

Instruction	Syntax	Operand Size	Operation
SWAP	Dn	16	
TAS	í	8	Destination Tested Condition Codes bit 7 of Destination
TBLS/TBLU	<ea>, Dn Dym : Dyn, Dn	8, 16, 32	$Dyn - Dym \Rightarrow Temp$ $(Temp * Dn [7 : 0]) \Rightarrow Temp$ $(Dym * 256) + Temp \Rightarrow Dn$
TBLSN/TBLUN	<ea>, Dn Dym : Dyn, Dn	8, 16, 32	$Dyn - Dym \Rightarrow Temp$ $(Temp * Dn [7 : 0]) / 256 \Rightarrow Temp$ $Dym + Temp \Rightarrow Dn$
TRAP	#<data>	none	$SSP - 2 \Rightarrow SSP$; format/vector offset $\Rightarrow (SSP)$; $SSP - 4 \Rightarrow SSP$; PC $\Rightarrow (SSP)$; SR $\Rightarrow (SSP)$; vector address $\Rightarrow PC$
TRAPcc	none #<data>	none 16, 32	If cc true, then TRAP exception
TRAPV	none	none	If V set, then overflow TRAP exception
TST	í	8, 16, 32	Source – 0, to set condition codes
UNLK	An	32	$An \Rightarrow SP$; $(SP) \Rightarrow An$, $SP + 4 \Rightarrow SP$

1. Privileged instruction.

6.2 Address Map

The “Access” column in the QSM address map below indicates which registers are accessible only at the supervisor privilege level and which can be assigned to either the supervisor or user privilege level, according to the value of the SUPV bit in the QSMCR.

Table 24 QSM Address Map

Access	Address	15	8	7	0
S	\$YFFC00	QSM MODULE CONFIGURATION (QSMCR)			
S	\$YFFC02	QSM TEST (QTEST)			
S	\$YFFC04	QSM INTERRUPT LEVEL (QILR)		QSM INTERRUPT VECTOR (QIVR)	
S/U	\$YFFC06	NOT USED			
S/U	\$YFFC08	SCI CONTROL 0 (SCCR0)			
S/U	\$YFFC0A	SCI CONTROL 1 (SCCR1)			
S/U	\$YFFC0C	SCI STATUS (SCSR)			
S/U	\$YFFC0E	SCI DATA (SCDR)			
S/U	\$YFFC10	NOT USED			
S/U	\$YFFC12	NOT USED			
S/U	\$YFFC14	NOT USED		PQS DATA (PORTQS)	
S/U	\$YFFC16	PQS PIN ASSIGNMENT (PQSPAR)		PQS DATA DIRECTION (DDRQS)	
S/U	\$YFFC18	SPI CONTROL 0 (SPCR0)			
S/U	\$YFFC1A	SPI CONTROL 1 (SPCR1)			
S/U	\$YFFC1C	SPI CONTROL 2 (SPCR2)			
S/U	\$YFFC1E	SPI CONTROL 3 (SPCR3)		SPI STATUS (SPSR)	
S/U	\$YFFC20– \$YFFCFF	NOT USED			
S/U	\$YFFD00– \$YFFD1F	RECEIVE RAM (RR[0:F])			
S/U	\$YFFD20– \$YFFD3F	TRANSMIT RAM (TR[0:F])			
S/U	\$YFFD40– \$YFFD4F	COMMAND RAM (CR[0:F])			

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

Table 25 QSPAR Pin Assignments

PQSPAR Field	PQSPAR Bit	Pin Function
PQSPA0	0	PQS0
	1	MISO
PQSPA1	0	PQS1
	1	MOSI
PQSPA2	0	PQS2 ¹
	1	SCK
PQSPA3	0	PQS3
	1	PCS0/ \overline{SS}
PQSPA4	0	PQS4
	1	PCS1
PQSPA5	0	PQS5
	1	PCS2
PQSPA6	0	PQS6
	1	PCS3
PQSPA7	0	PQS7 ²
	1	TXD

NOTES:

1. PQS2 is a digital I/O pin unless the SPI is enabled (SPE in SPCR1 set), in which case it becomes SPI serial clock SCK.
2. PQS7 is a digital I/O pin unless the SCI transmitter is enabled (TE in SCCR1 = 1), in which case it becomes SCI serial output TXD.

DDRQS determines whether pins are inputs or outputs. Clearing a bit makes the corresponding pin an input; setting a bit makes the pin an output. DDRQS affects both QSPI function and I/O function.

6.5 QSPI Submodule

The QSPI submodule communicates with external devices through 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 is shown below.

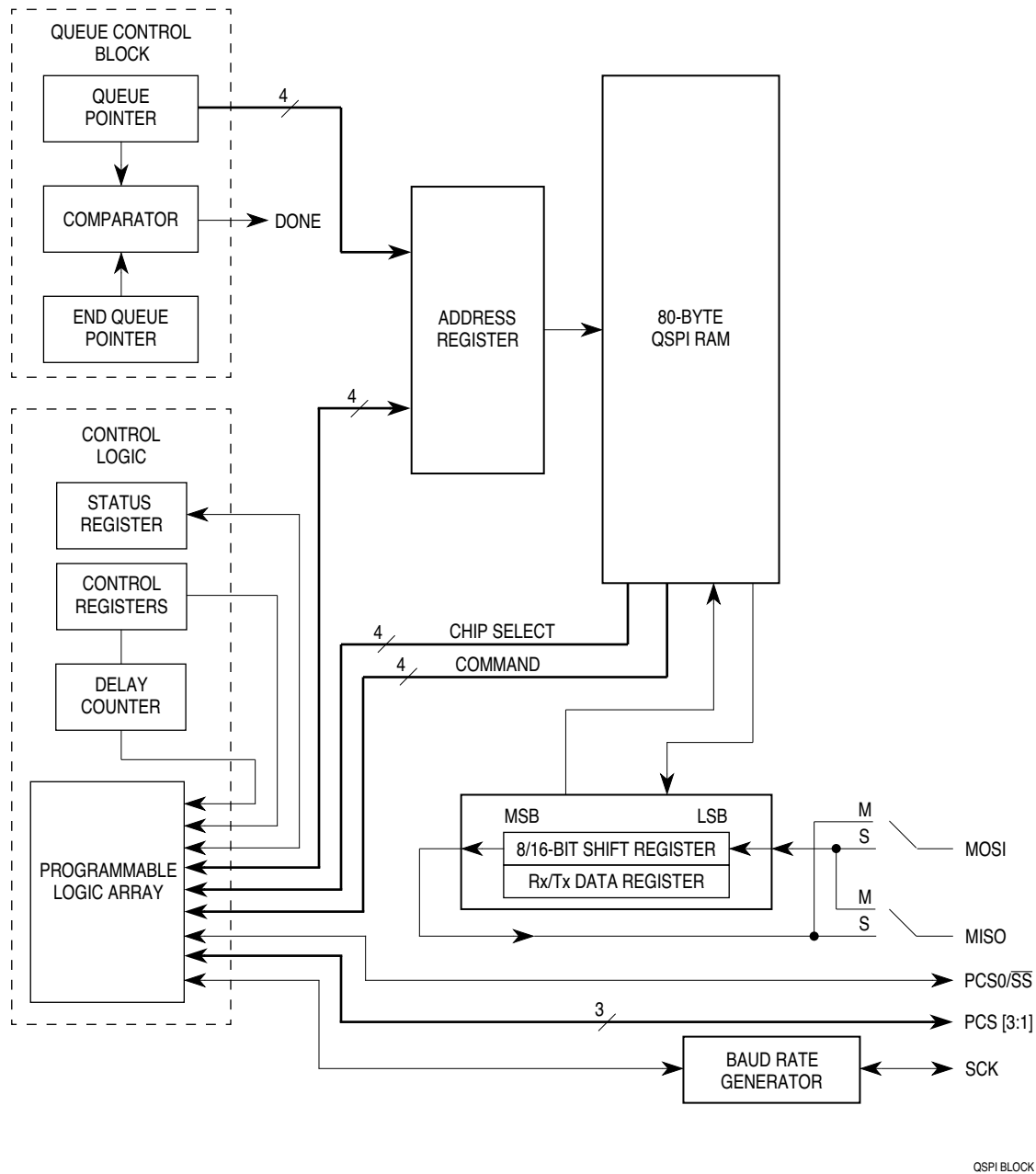


Figure 14 QSPI Block Diagram

6.5.1 QSPI Pins

Seven pins are associated with the QSPI. When not needed for a QSPI application, they can be configured as general-purpose I/O pins. The **PCS0/SS** pin can function as a peripheral chip select output, slave select input, or general-purpose I/O. Refer to the following table for QSPI input and output pins and their functions.

SCK baud rate:

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

or

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

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

Giving SPBR a value of zero or one disables the baud rate generator. SCK is disabled and assumes its inactive state value. No serial transfers occur. At reset, baud rate is initialized to one eighth of the system clock frequency.

SPCR1 — QSPI Control Register 1

\$YFFC1A

15	14	8	7	0
SPE	DSCKL		DTL	

RESET:

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

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

SPE — QSPI Enable

0 = QSPI is disabled. QSPI pins can be used for general-purpose I/O.

1 = QSPI is enabled. Pins allocated by PQSPAR are controlled by the QSPI.

DSCKL — Delay before SCK

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

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

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

When the DSCK value of a queue entry 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 command RAM is set, this field determines the length of delay after serial transfer. The following equation is used to calculate the delay:

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

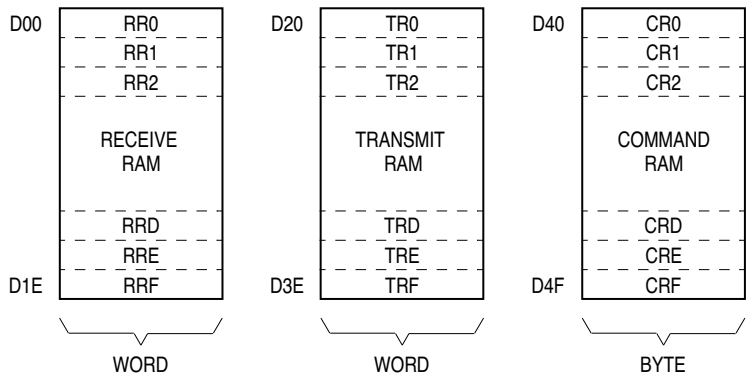
where DTL equals {1, 2, 3,..., 255}.

A zero value for DTL causes a delay-after-transfer value of 8192/System Clock.

If DT equals zero, a standard delay is inserted.

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

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



QSPI RAM MAP

Figure 15 QSPI RAM

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 execute a queue of commands repeatedly without CPU intervention.

RR[0:F] — 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.

TR[0:F] — Transmit Data RAM \$YFFD20

Data that is to be transmitted by the QSPI is stored in this segment. The CPU usually 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.

CR[0:F] — 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} .

7 Standby RAM with TPU Emulation RAM

The TPURAM module contains a 2-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.

7.1 Overview

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. TPURAM responds to both program and data space accesses. Data can be read or written in bytes, words, or long words. The TPURAM is powered by V_{DD} in normal operation. During power-down, the TPURAM contents are maintained by power on standby voltage pin V_{STBY} . Power switching between sources is automatic.

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

An address map of the TPURAM control registers follows. All TPURAM control registers are located in supervisor data space.

Table 28 TPURAM Control Register Address Map

Access	Address	15	8	7	0
S	\$YFFB00	TPURAM MODULE CONFIGURATION REGISTER (TRAMMCR)			
S	\$YFFB02	TPURAM TEST REGISTER (TRAMTST)			
S	\$YFFB04	TPURAM BASE ADDRESS REGISTER (TRAMBAR)			
	\$YFFB06– \$YFFB3F	NOT USED			

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

7.2 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.

7.3 TPURAM Registers

TRAMMCR —TPURAM Module Configuration Register

\$YFFB00

15	14	13	12	11	10	9	8	7	0
STOP	0	0	0	0	0	0	RASP	NOT USED	

RESET:

0 0 0 0 0 0 0 1

TSTOP —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.

8 Summary of Changes

This is a partial revision. Most of the publication remains the same, but the following changes were made to improve it. Typographical errors that do not affect content are not annotated. This document has also been reformatted for use on the web.

Pages 2-3	New Ordering Information included.
Page 6	New block diagram drawn.
Page 7	New 132-pin assignment diagram drawn.
Page 8	New 144-pin assignment diagram drawn.
Page 9	New address map drawn.
Pages 10-14	Added Signal Description section.
Pages 15-47	Expanded and revised SIM section. Made all register diagrams and bit mnemonics consistent. Incorporated new information concerning the system clock, resets, interrupts, and chip-selects circuits.
Page 48-56	Expanded and revised CPU section. Made all register diagrams and bit mnemonics consistent. Revised instruction set summary information.
Page 57-70	Expanded and revised TPU section. Made all register diagrams and bit mnemonics consistent. Revised time functions information to include both MC68332A and MC68332G microcode ROM applications.
Page 71-92	Expanded and revised QSM section. Made all register diagrams and bit mnemonics consistent. Added information concerning SPI and SCI operation.
Page 93-95	Revised Standby RAM with TPU Emulation RAM section. Made all register diagrams and bit mnemonics consistent.

