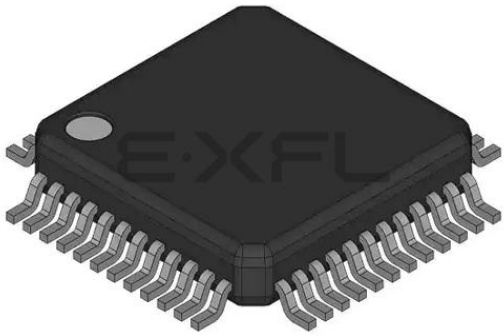


Welcome to E-XFL.COM

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	Active
Core Processor	HC08
Core Size	8-Bit
Speed	8MHz
Connectivity	I²C, IRSCI, SCI, SPI
Peripherals	LED, LVD, POR, PWM
Number of I/O	32
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc908ap64cfaer

Table of Contents

Table 2-1. Vector Addresses (Continued)

Priority	INT Flag	Address	Vector
↓	IF8	$FFEC	TIM2 Channel 1 Vector (High)
		$FFED	TIM2 Channel 1 Vector (Low)
	IF7	$FFEE	TIM2 Channel 0 Vector (High)
		$FFEF	TIM2 Channel 0 Vector (Low)
	IF6	$FFF0	TIM1 Overflow Vector (High)
		$FFF1	TIM1 Overflow Vector (Low)
	IF5	$FFF2	TIM1 Channel 1 Vector (High)
		$FFF3	TIM1 Channel 1 Vector (Low)
	IF4	$FFF4	TIM1 Channel 0 Vector (High)
		$FFF5	TIM1 Channel 0 Vector (Low)
	IF3	$FFF6	PLL Vector (High)
		$FFF7	PLL Vector (Low)
	IF2	$FFF8	$\overline{\text{IRQ2}}$ Vector (High)
		$FFF9	$\overline{\text{IRQ2}}$ Vector (Low)
	IF1	$FFFA	$\overline{\text{IRQ1}}$ Vector (High)
		$FFFB	$\overline{\text{IRQ1}}$ Vector (Low)
	—	$FFFC	SWI Vector (High)
		$FFFD	SWI Vector (Low)
	—	$FFFE	Reset Vector (High)
Highest		$FFFF	Reset Vector (Low)

2.4 Random-Access Memory (RAM)

The following table shows the RAM size and address range:

Device	RAM Size (Bytes)	Memory Address Range
MC68HC908AP64	2,048	$0060–$085F
MC68HC908AP32		
MC68HC908AP16	1,024	$0060–$045F
MC68HC908AP8		

The location of the stack RAM is programmable. The 16-bit stack pointer allows the stack to be anywhere in the 64k-byte memory space.

NOTE

For correct operation, the stack pointer must point only to RAM locations.

Within page zero are 160 bytes of RAM. Because the location of the stack RAM is programmable, all page zero RAM locations can be used for I/O control and user data or code. When the stack pointer is moved from its reset location at $00FF, direct addressing mode instructions can access efficiently all page zero RAM locations. Page zero RAM, therefore, provides ideal locations for frequently accessed global variables.

Before processing an interrupt, the CPU uses five bytes of the stack to save the contents of the CPU registers.

NOTE

For M6805 compatibility, the H register is not stacked.

During a subroutine call, the CPU uses two bytes of the stack to store the return address. The stack pointer decrements during pushes and increments during pulls.

NOTE

Be careful when using nested subroutines. The CPU may overwrite data in the RAM during a subroutine or during the interrupt stacking operation.

2.5 FLASH Memory

This sub-section describes the operation of the embedded FLASH memory. This memory can be read, programmed, and erased from a single external supply. The program and erase operations are enabled through the use of an internal charge pump. The following table shows the FLASH memory size and address range:

Device	FLASH Size (Bytes)	Memory Address Range
MC68HC908AP64	62,368	$0860–$FBFF
MC68HC908AP32	32,768	$0860–$885F
MC68HC908AP16	16,384	$0860–$485F
MC68HC908AP8	8,192	$0860–$285F

2.5.1 Functional Description

The FLASH memory consists of an array for user memory plus a block of 48 bytes for user interrupt vectors and one byte for the mask option register. *An erased bit reads as logic 1 and a programmed bit reads as a logic 0.* The FLASH memory page size is defined as 512 bytes, and is the minimum size that can be erased in a page erase operation. Program and erase operations are facilitated through control bits in FLASH control register (FLCR). The address ranges for the FLASH memory are:

- $0860–$FBFF; user memory, 62,368 / 32,768 / 16,384 / 8,192 bytes
- $FFD0–$FFFF; user interrupt vectors, 48 bytes
- $FFCF; mask option register

Programming tools are available from Freescale. Contact your local Freescale representative for more information.

NOTE

A security feature prevents viewing of the FLASH contents.[1]

Chapter 4
Central Processor Unit (CPU)

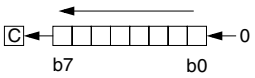
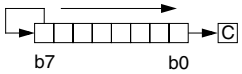
4.1 Introduction

The M68HC08 CPU (central processor unit) is an enhanced and fully object-code-compatible version of the M68HC05 CPU. The *CPU08 Reference Manual* (Freescale document order number CPU08RM/AD) contains a description of the CPU instruction set, addressing modes, and architecture.

4.2 Features

- Object code fully upward-compatible with M68HC05 Family
- 16-bit stack pointer with stack manipulation instructions
- 16-Bit index register with x-register manipulation instructions
- 8-MHz CPU internal bus frequency
- 64-Kbyte program/data memory space
- 16 addressing modes
- Memory-to-memory data moves without using accumulator
- Fast 8-bit by 8-bit multiply and 16-bit by 8-bit divide instructions
- Enhanced binary-coded decimal (BCD) data handling
- Modular architecture with expandable internal bus definition for extension of addressing range beyond 64 Kbytes
- Low-power stop and wait modes

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
AND #*opr* AND *opr* AND *opr* AND *opr*,X AND *opr*,X AND ,X AND *opr*,SP AND *opr*,SP	Logical AND	A ← (A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A4 B4 C4 D4 E4 F4 9EE4 9ED4	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
ASL *opr* ASLA ASLX ASL *opr*,X ASL ,X ASL *opr*,SP	Arithmetic Shift Left (Same as LSL)		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	4 1 1 4 3 5
ASR *opr* ASRA ASRX ASR *opr*,X ASR *opr*,X ASR *opr*,SP	Arithmetic Shift Right		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	37 47 57 67 77 9E67	dd ff ff	4 1 1 4 3 5
BCC *rel*	Branch if Carry Bit Clear	PC ← (PC) + 2 + rel ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BCLR *n*, *opr*	Clear Bit n in M	Mn ← 0	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 4
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE *opr*	Branch if Greater Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGT *opr*	Branch if Greater Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) ǀ (N ⊕ V)=0	–	–	–	–	–	–	REL	92	rr	3
BHCC *rel*	Branch if Half Carry Bit Clear	PC ← (PC) + 2 + *rel* ? (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS *rel*	Branch if Half Carry Bit Set	PC ← (PC) + 2 + *rel* ? (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? (C) ǀ (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS *rel*	Branch if Higher or Same (Same as BCC)	PC ← (PC) + 2 + *rel* ? (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH *rel*	Branch if $\overline{\text{IRQ}}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL *rel*	Branch if $\overline{\text{IRQ}}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{\text{IRQ}}$ = 0	–	–	–	–	–	–	REL	2E	rr	3

Enter monitor mode with pin configuration shown in Figure 8-1 by pulling $\overline{RST}$ low and then high. The rising edge of $\overline{RST}$ latches monitor mode. Once monitor mode is latched, the values on the specified pins can change.

Once out of reset, the MCU waits for the host to send eight security bytes. (See 8.4 Security.) After the security bytes, the MCU sends a break signal (10 consecutive logic 0's) to the host, indicating that it is ready to receive a command.

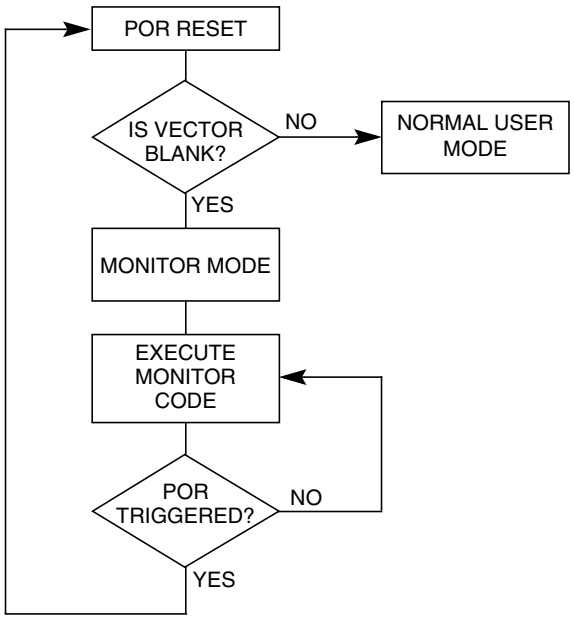


Figure 8-2. Low-Voltage Monitor Mode Entry Flowchart

In monitor mode, the MCU uses different vectors for reset, SWI (software interrupt), and break interrupt than those for user mode. The alternate vectors are in the $FE page instead of the $FF page and allow code execution from the internal monitor firmware instead of user code.

NOTE

Exiting monitor mode after it has been initiated by having a blank reset vector requires a power-on reset (POR). Pulling $\overline{RST}$ low will not exit monitor mode in this situation.

Table 8-2 summarizes the differences between user mode and monitor mode vectors.

Table 8-2. Mode Differences (Vectors)

Modes	Functions					
	Reset Vector High	Reset Vector Low	Break Vector High	Break Vector Low	SWI Vector High	SWI Vector Low
User	$FFFE	$FFFF	$FFFC	$FFFD	$FFFC	$FFFD
Monitor	$FEFE	$FEFF	$FEFC	$FEFD	$FEFC	$FEFD

8.5.3 LDRNGE

LDRNGE is used to load the data array in RAM with data from a range of FLASH locations.

Table 8-13. LDRNGE Routine

Routine Name	LDRNGE
Routine Description	Loads data from a range of locations
Calling Address	$FC00
Stack Used	7 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Starting address (ADDRH) Starting address (ADDRL) Data 1 : Data N

The start location of FLASH from where data is retrieved is specified by the address ADDRH:ADDRL and the number of bytes from this location is specified by DATASIZE. The maximum number of bytes that can be retrieved in one routine call is 255 bytes. The data retrieved from FLASH is loaded into the data array in RAM. Previous data in the data array will be overwritten. User can use this routine to retrieve data from FLASH that was previously programmed.

The coding example below is to retrieve 64 bytes of data starting from $EE00 in FLASH. The Initialization subroutine is the same as the coding example for PRGRNGE (see 8.5.1 PRGRNGE).

```
LDRNGE          EQU     $FC00
MAIN:
        BSR     INITIALIZATION
        :
        :
        LDHX    #FILE_PTR
        JSR     LDRNGE
        :
```

TOF — TIM Overflow Flag Bit

This read/write flag is set when the TIM counter reaches the modulo value programmed in the TIM counter modulo registers. Clear TOF by reading the TIM status and control register when TOF is set and then writing a logic 0 to TOF. If another TIM overflow occurs before the clearing sequence is complete, then writing logic 0 to TOF has no effect. Therefore, a TOF interrupt request cannot be lost due to inadvertent clearing of TOF. Reset clears the TOF bit. Writing a logic 1 to TOF has no effect.

1 = TIM counter has reached modulo value
0 = TIM counter has not reached modulo value

TOIE — TIM Overflow Interrupt Enable Bit

This read/write bit enables TIM overflow interrupts when the TOF bit becomes set. Reset clears the TOIE bit.

1 = TIM overflow interrupts enabled
0 = TIM overflow interrupts disabled

TSTOP — TIM Stop Bit

This read/write bit stops the TIM counter. Counting resumes when TSTOP is cleared. Reset sets the TSTOP bit, stopping the TIM counter until software clears the TSTOP bit.

1 = TIM counter stopped
0 = TIM counter active

NOTE

Do not set the TSTOP bit before entering wait mode if the TIM is required to exit wait mode.

TRST — TIM Reset Bit

Setting this write-only bit resets the TIM counter and the TIM prescaler. Setting TRST has no effect on any other registers. Counting resumes from $0000. TRST is cleared automatically after the TIM counter is reset and always reads as logic 0. Reset clears the TRST bit.

1 = Prescaler and TIM counter cleared
0 = No effect

NOTE

Setting the TSTOP and TRST bits simultaneously stops the TIM counter at a value of $0000.

PS[2:0] — Prescaler Select Bits

These read/write bits select one of the seven prescaler outputs as the input to the TIM counter as Table 9-2 shows. Reset clears the PS[2:0] bits.

Table 9-2. Prescaler Selection

PS2	PS1	PS0	TIM Clock Source
0	0	0	Internal bus clock ÷ 1
0	0	1	Internal bus clock ÷ 2
0	1	0	Internal bus clock ÷ 4
0	1	1	Internal bus clock ÷ 8
1	0	0	Internal bus clock ÷ 16
1	0	1	Internal bus clock ÷ 32
1	1	0	Internal bus clock ÷ 64
1	1	1	Not available

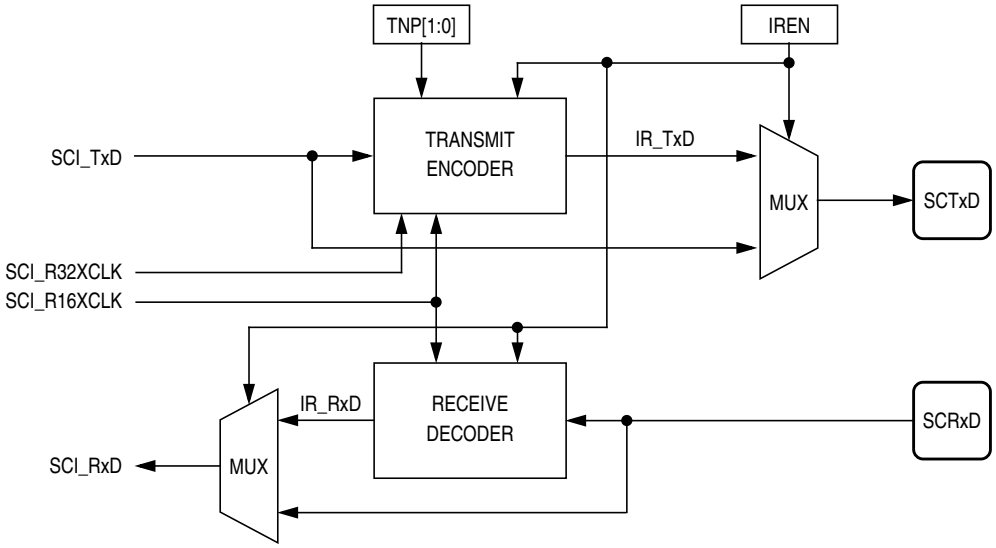


Figure 12-3. Infrared Sub-Module Diagram

12.4.1 Infrared Transmit Encoder

The infrared transmit encoder converts the "0" bits in the serial data stream from the SCI module to narrow "low" pulses, to the TxD pin. The narrow pulse is sent with a duration of 1/32, 1/16, or 3/16 of a data bit width**.** When two consecutive zeros are sent, the two consecutive narrow pulses will be separated by a time equal to a data bit width.

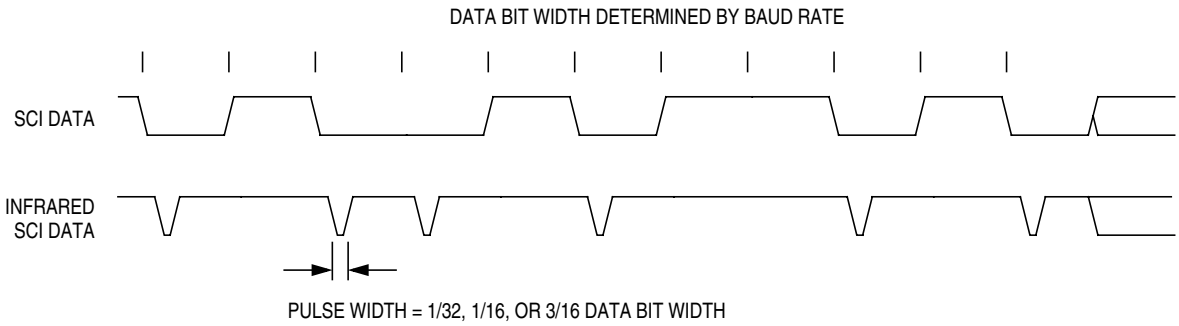


Figure 12-4. Infrared SCI Data Example

12.4.2 Infrared Receive Decoder

The infrared receive decoder converts low narrow pulses from the RxD pin to standard SCI data bits. The reference clock, SCI_R16XCLK, clocks a four bit internal counter which counts from 0 to 15. An incoming pulse starts the internal counter and a "0" is sent out to the IR_RxD output. Subsequent incoming pulses are ignored when the counter count is between 0 and 7; IR_RxD remains "0". Once the counter passes 7, an incoming pulse will reset the counter; IR_RxD remains "0". When the counter reaches 15, the IR_RxD output returns to "1", the counter stops and waits for further pulses. A pulse is interpreted as jitter if it arrives shortly after the counter reaches 15; IR_RxD remains "1".

The SCI allows full-duplex, asynchronous, NRZ serial communication between the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud rate generator. During normal operation, the CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

NOTE

For SCI operations, the IR sub-module is transparent to the SCI module. Data at going out of the SCI transmitter and data going into the SCI receiver is always in SCI format. It makes no difference to the SCI module whether the IR sub-module is enabled or disabled.

NOTE

This SCI module is a standard HC08 SCI module with the following modifications:

- *A control bit, CKS, is added to the SCI baud rate control register to select between two input clocks for baud rate clock generation*
- *The TXINV bit is removed from the SCI control register 1*

12.5.1 Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in Figure 12-6.

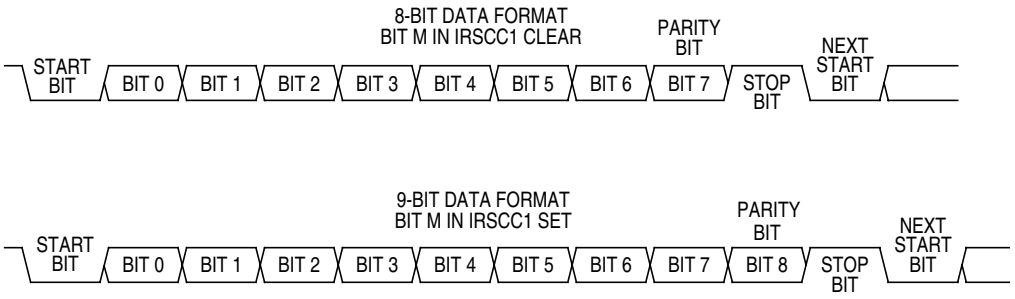


Figure 12-6. SCI Data Formats

12.5.2 Transmitter

Figure 12-7 shows the structure of the SCI transmitter.

The baud rate clock source for the SCI can be selected by the CKS bit, in the SCI baud rate register (see 12.9.7 IRSCI Baud Rate Register).

12.5.2.1 Character Length

The transmitter can accommodate either 8-bit or 9-bit data. The state of the M bit in IRSCI control register 1 (IRSCC1) determines character length. When transmitting 9-bit data, bit T8 in IRSCI control register 3 (IRSCC3) is the ninth bit (bit 8).

$$\left|\frac{170 - 176}{170}\right| \times 100 = 3.53\%$$

12.5.3.6 Receiver Wakeup

So that the MCU can ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in IRSCC2 puts the receiver into a standby state during which receiver interrupts are disabled.

Depending on the state of the WAKE bit in IRSCC1, either of two conditions on the RxD pin can bring the receiver out of the standby state:

- Address mark — An address mark is a logic 1 in the most significant bit position of a received character. When the WAKE bit is set, an address mark wakes the receiver from the standby state by clearing the RWU bit. The address mark also sets the SCI receiver full bit, SCRF. Software can then compare the character containing the address mark to the user-defined address of the receiver. If they are the same, the receiver remains awake and processes the characters that follow. If they are not the same, software can set the RWU bit and put the receiver back into the standby state.
- Idle input line condition — When the WAKE bit is clear, an idle character on the RxD pin wakes the receiver from the standby state by clearing the RWU bit. The idle character that wakes the receiver does not set the receiver idle bit, IDLE, or the SCI receiver full bit, SCRF. The idle line type bit, ILTY, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit.

NOTE

Clearing the WAKE bit after the RxD pin has been idle may cause the receiver to wake up immediately.

12.5.3.7 Receiver Interrupts

The following sources can generate CPU interrupt requests from the SCI receiver:

- SCI receiver full (SCRF) — The SCRF bit in IRSCS1 indicates that the receive shift register has transferred a character to the IRSCDR. SCRF can generate a receiver interrupt request. Setting the SCI receive interrupt enable bit, SCRIE, in IRSCC2 enables the SCRF bit to generate receiver CPU interrupts.
- Idle input (IDLE) — The IDLE bit in IRSCS1 indicates that 10 or 11 consecutive logic 1s shifted in from the RxD pin. The idle line interrupt enable bit, ILIE, in IRSCC2 enables the IDLE bit to generate CPU interrupt requests.

12.5.3.8 Error Interrupts

The following receiver error flags in IRSCS1 can generate CPU interrupt requests:

- Receiver overrun (OR) — The OR bit indicates that the receive shift register shifted in a new character before the previous character was read from the IRSCDR. The previous character remains in the IRSCDR, and the new character is lost. The overrun interrupt enable bit, ORIE, in IRSCC3 enables OR to generate SCI error CPU interrupt requests.
- Noise flag (NF) — The NF bit is set when the SCI detects noise on incoming data or break characters, including start, data, and stop bits. The noise error interrupt enable bit, NEIE, in IRSCC3 enables NF to generate SCI error CPU interrupt requests.

RPF — Reception in Progress Flag Bit

This read-only bit is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RPF does not generate an interrupt request. RPF is reset after the receiver detects false start bits (usually from noise or a baud rate mismatch) or when the receiver detects an idle character. Polling RPF before disabling the SCI module or entering stop mode can show whether a reception is in progress.

1 = Reception in progress
0 = No reception in progress

12.9.6 IRSCI Data Register

The IRSCI data register is the buffer between the internal data bus and the receive and transmit shift registers. Reset has no effect on data in the IRSCI data register.

Address: $0045

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	R7	R6	R5	R4	R3	R2	R1	R0
Write:	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	Unaffected by reset							

Figure 12-18. IRSCI Data Register (IRSCDR)

R7/T7–R0/T0 — Receive/Transmit Data Bits

Reading the IRSCDR accesses the read-only received data bits, R7–R0. Writing to the IRSCDR writes the data to be transmitted, T7–T0. Reset has no effect on the IRSCDR.

NOTE

Do not use read/modify/write instructions on the IRSCI data register.

12.9.7 IRSCI Baud Rate Register

The baud rate register selects the baud rate for both the receiver and the transmitter.

Address: $0046

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CKS	0	SCP1	SCP0	R	SCR2	SCR1	SCR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved

Figure 12-19. IRSCI Baud Rate Register (IRSCBR)

CKS — Baud Clock Input Select

This read/write bit selects the source clock for the baud rate generator. Reset clears the CKS bit, selecting CGMXCLK.

1 = Bus clock drives the baud rate generator
0 = CGMXCLK drives the baud rate generator

SCP1 and SCP0 — SCI Baud Rate Prescaler Bits

These read/write bits select the baud rate prescaler divisor as shown in Table 12-7. Reset clears SCP1 and SCP0.

ERRIE — Error Interrupt Enable Bit

This read/write bit enables the MODF and OVRF bits to generate CPU interrupt requests. Reset clears the ERRIE bit.

1 = MODF and OVRF can generate CPU interrupt requests
0 = MODF and OVRF cannot generate CPU interrupt requests

OVRF — Overflow Bit

This clearable, read-only flag is set if software does not read the byte in the receive data register before the next full byte enters the shift register. In an overflow condition, the byte already in the receive data register is unaffected, and the byte that shifted in last is lost. Clear the OVRF bit by reading the SPI status and control register with OVRF set and then reading the receive data register. Reset clears the OVRF bit.

1 = Overflow
0 = No overflow

MODF — Mode Fault Bit

This clearable, read-only flag is set in a slave SPI if the $\overline{SS}$ pin goes high during a transmission with the MODFEN bit set. In a master SPI, the MODF flag is set if the $\overline{SS}$ pin goes low at any time with the MODFEN bit set. Clear the MODF bit by reading the SPI status and control register (SPSCR) with MODF set and then writing to the SPI control register (SPCR). Reset clears the MODF bit.

1 = $\overline{SS}$ pin at inappropriate logic level
0 = $\overline{SS}$ pin at appropriate logic level

SPTE — SPI Transmitter Empty Bit

This clearable, read-only flag is set each time the transmit data register transfers a byte into the shift register. SPTE generates an SPTE CPU interrupt request if the SPTIE bit in the SPI control register is set also.

NOTE

Do not write to the SPI data register unless the SPTE bit is high.

During an SPTE CPU interrupt, the CPU clears the SPTE bit by writing to the transmit data register. Reset sets the SPTE bit.

1 = Transmit data register empty
0 = Transmit data register not empty

MODFEN — Mode Fault Enable Bit

This read/write bit, when set to 1, allows the MODF flag to be set. If the MODF flag is set, clearing the MODFEN does not clear the MODF flag. If the SPI is enabled as a master and the MODFEN bit is low, then the $\overline{SS}$ pin is available as a general-purpose I/O.

If the MODFEN bit is set, then this pin is not available as a general-purpose I/O. When the SPI is enabled as a slave, the $\overline{SS}$ pin is not available as a general-purpose I/O regardless of the value of MODFEN. (See 13.12.4 SS (Slave Select).)

If the MODFEN bit is low, the level of the $\overline{SS}$ pin does not affect the operation of an enabled SPI configured as a master. For an enabled SPI configured as a slave, having MODFEN low only prevents the MODF flag from being set. It does not affect any other part of SPI operation. (See 13.7.2 Mode Fault Error.)

SPR1 and SPR0 — SPI Baud Rate Select Bits

In master mode, these read/write bits select one of four baud rates as shown in Table 13-4. SPR1 and SPR0 have no effect in slave mode. Reset clears SPR1 and SPR0.

Only the slave with a matched address will respond by sending back an acknowledge bit by pulling SDA low on the 9th clock cycle.
(See Figure 14-2.)

14.5.3 Data Transfer

Once a successful slave addressing is achieved, the data transfer can proceed byte by byte in the direction specified by the R/W-bit sent by the calling master.

Each data byte is 8 bits. Data can be changed only when SCL is low and must be held stable when SCL is high as shown in Figure 14-2. The MSB is transmitted first and each byte has to be followed by an acknowledge bit. This is signalled by the receiving device by pulling the SDA low on the 9th clock cycle. Therefore, one complete data byte transfer requires 9 clock cycles.

If the slave receiver does not acknowledge the master, the SDA line should be left high by the slave. The master can then generate a STOP signal to abort the data transfer or a START signal (repeated START) to commence a new transfer.

If the master receiver does not acknowledge the slave transmitter after a byte has been transmitted, it means an “end of data” to the slave. The slave should release the SDA line for the master to generate a STOP or START signal.

14.5.4 Repeated START Signal

As shown in Figure 14-2, a repeated START signal is used to generate START signal without first generating a STOP to terminate the communication. This is used by the master to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

14.5.5 STOP Signal

The master can terminate the communication by generating a STOP signal to free the bus. However, the master may generate a START signal followed by a calling command without first generating a STOP signal. This is called repeat START. A STOP signal is defined as a low to high transition of SDA while SCL is at logic high (see Figure 14-2).

14.5.6 Arbitration Procedure

The interface circuit is a multi-master system which allows more than one master to be connected. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. The clock low period is equal to the longest clock low period and the clock high period is equal to the shortest one among the masters. A data arbitration procedure determines the priority. A master will lose arbitration if it transmits a logic 1 while another transmits a logic 0. The losing master will immediately switch over to slave receive mode and stops its data and clock outputs. The transition from master to slave will not generate a STOP condition. Meanwhile a software bit will be set by hardware to indicates loss of arbitration.

14.5.7 Clock Synchronization

Since wired-AND logic is performed on SCL line, a high to low transition on the SCL line will affect the devices connected to the bus. The devices start counting their low period once a device’s clock has gone low, it will hold the SCL line low until the clock high state is reached. However, the change of low to high

14.6.1 MMIIC Address Register (MMADR)

Address: $0048

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MMAD7	MMAD6	MMAD5	MMAD4	MMAD3	MMAD2	MMAD1	MMEXTAD
Reset:	1	0	1	0	0	0	0	0

Figure 14-4. MMIIC Address Register (MMADR)

MMAD[7:1] — Multi-Master Address

These seven bits represent the MMIIC interface's own specific slave address when in slave mode, and the calling address when in master mode. Software must update MMAD[7:1] as the calling address while entering master mode and restore its own slave address after master mode is relinquished. This register is cleared as $A0 upon reset.

MMEXTAD — Multi-Master Expanded Address

This bit is set to expand the address of the MMIIC in slave mode. When set, the MMIIC will acknowledge the following addresses from a calling master: $MMAD[7:1], 0000000, and 0001100. Reset clears this bit.

1 = MMIIC responds to the following calling addresses:
$MMAD[7:1], 0000000, and 0001100.
0 = MMIIC responds to address $MMAD[7:1]

For example, when MMADR is configured as:

MMAD7	MMAD6	MMAD5	MMAD4	MMAD3	MMAD2	MMAD1	MMEXTAD
1	1	0	1	0	1	0	1

The MMIIC module will respond to the calling address:

Bit 7	6	5	4	3	2	Bit 1
1	1	0	1	0	1	0

or the general calling address:

0	0	0	0	0	0	0

or the calling address:

Bit 7	6	5	4	3	2	Bit 1
0	0	0	1	1	0	0

Note that bit-0 of the 8-bit calling address is the MMRW bit from the calling master.

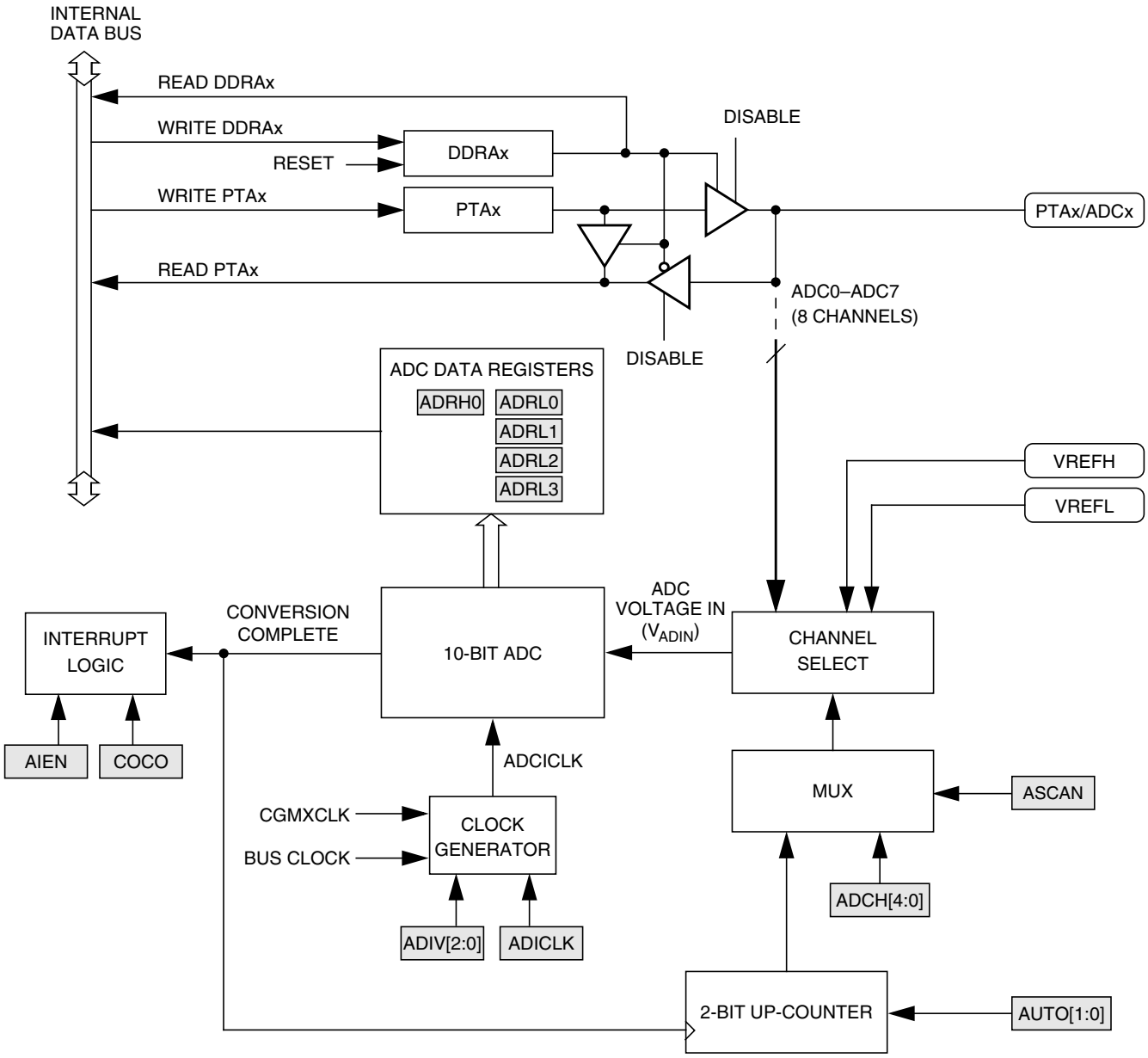


Figure 15-2. ADC Block Diagram

15.3.3 Conversion Time

Conversion starts after a write to the ADSCR. One conversion will take between 16 and 17 ADC clock cycles, therefore:

$$\text{Conversion time} = \frac{\text{16 to17 ADC cycles}}{\text{ADC frequency}}$$

Number of bus cycles = conversion time × bus frequency

- Software clear — Software can clear an interrupt latch by writing to the appropriate acknowledge bit in the interrupt status and control register (INTSCR). Writing a logic 1 to the ACK bit clears the IRQ latch.
- Reset — A reset automatically clears the interrupt latch.

The external interrupt pin is falling-edge-triggered and is software-configurable to be either falling-edge or falling-edge and low-level-triggered. The MODE bit in the INTSCR controls the triggering sensitivity of the $\overline{\text{IRQ}}$ pin.

When an interrupt pin is edge-triggered only, the interrupt remains set until a vector fetch, software clear, or reset occurs.

When an interrupt pin is both falling-edge and low-level-triggered, the interrupt remains set until both of the following occur:

- Vector fetch or software clear
- Return of the interrupt pin to logic 1

The vector fetch or software clear may occur before or after the interrupt pin returns to logic 1. As long as the pin is low, the interrupt request remains pending. A reset will clear the latch and the MODE1 control bit, thereby clearing the interrupt even if the pin stays low.

When set, the IMASK bit in the INTSCR mask all external interrupt requests. A latched interrupt request is not presented to the interrupt priority logic unless the IMASK bit is clear.

NOTE

The interrupt mask (I) in the condition code register (CCR) masks all interrupt requests, including external interrupt requests.

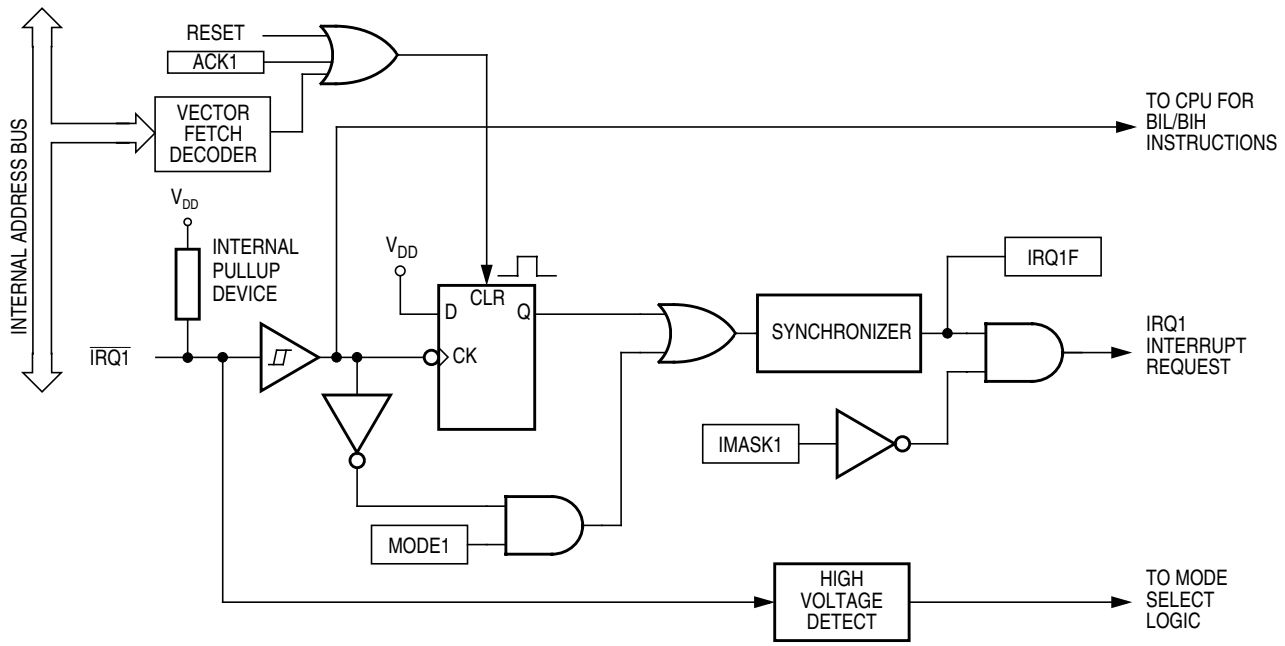


Figure 17-2. IRQ1 Block Diagram

IMASK1 — IRQ1 Interrupt Mask Bit

Writing a logic 1 to this read/write bit disables IRQ1 interrupt requests. Reset clears IMASK1.

1 = $\overline{\text{IRQ1}}$ interrupt requests disabled
0 = $\overline{\text{IRQ1}}$ interrupt requests enabled

MODE1 — IRQ1 Edge/Level Select Bit

This read/write bit controls the triggering sensitivity of the $\overline{\text{IRQ1}}$ pin. Reset clears MODE1.

1 = $\overline{\text{IRQ1}}$ interrupt requests on falling edges and low levels
0 = $\overline{\text{IRQ1}}$ interrupt requests on falling edges only

17.6.2 IRQ2 Status and Control Register

The IRQ2 status and control register (INTSCR2) controls and monitors operation of IRQ2. The INTSCR2 has the following functions:

- Enables/disables the internal pullup device on $\overline{\text{IRQ2}}$ pin
- Shows the state of the IRQ2 flag
- Clears the IRQ2 latch
- Masks IRQ2 interrupt request
- Controls triggering sensitivity of the $\overline{\text{IRQ2}}$ interrupt pin

Address: $001C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	PUC0ENB	0	0	IRQ2F	0	IMASK2	MODE2
Write:						ACK2		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 17-5. IRQ2 Status and Control Register (INTSCR2)

PUC0ENB — $\overline{\text{IRQ2}}$ Pin Pullup Enable Bit.

Setting this bit to logic 1 disables the pullup on PTC0/$\overline{\text{IRQ2}}$ pin.
Reset clears this bit.

1 = $\overline{\text{IRQ2}}$ pin internal pullup is disabled
0 = $\overline{\text{IRQ2}}$ pin internal pullup is enabled

IRQ2F — IRQ2 Flag Bit

This read-only status bit is high when the IRQ2 interrupt is pending.

1 = $\overline{\text{IRQ2}}$ interrupt pending
0 = $\overline{\text{IRQ2}}$ interrupt not pending

ACK2 — IRQ2 Interrupt Request Acknowledge Bit

Writing a logic 1 to this write-only bit clears the IRQ2 latch. ACK2 always reads as logic 0. Reset clears ACK2.

IMASK2 — IRQ2 Interrupt Mask Bit

Writing a logic 1 to this read/write bit disables IRQ2 interrupt requests. Reset clears IMASK2.

1 = IRQ2 interrupt requests disabled
0 = IRQ2 interrupt requests enabled

MODE2 — IRQ2 Edge/Level Select Bit

This read/write bit controls the triggering sensitivity of the $\overline{\text{IRQ2}}$ pin. Reset clears MODE2.

1 = $\overline{\text{IRQ2}}$ interrupt requests on falling edges and low levels
0 = $\overline{\text{IRQ2}}$ interrupt requests on falling edges only

NOTE

Service the COP immediately after reset and before entering or after exiting STOP Mode to guarantee the maximum time before the first COP counter overflow.

A COP reset pulls the $\overline{\text{RST}}$ pin low for 32 ICLK cycles and sets the COP bit in the SIM reset status register (SRSR).

In monitor mode, the COP is disabled if the $\overline{\text{RST}}$ pin or the $\overline{\text{IRQ1}}$ is held at V_{TST}. During the break state, V_{TST} on the $\overline{\text{RST}}$ pin disables the COP.

NOTE

Place COP clearing instructions in the main program and not in an interrupt subroutine. Such an interrupt subroutine could keep the COP from generating a reset even while the main program is not working properly.

19.3 I/O Signals

The following paragraphs describe the signals shown in Figure 19-1.

19.3.1 ICLK

ICLK is the internal oscillator output signal. See Chapter 22 Electrical Specifications for ICLK frequency specification.

19.3.2 STOP Instruction

The STOP instruction clears the COP prescaler.

19.3.3 COPCTL Write

Writing any value to the COP control register (COPCTL) (see 19.4 COP Control Register) clears the COP counter and clears bits 12 through 5 of the prescaler. Reading the COP control register returns the low byte of the reset vector.

19.3.4 Power-On Reset

The power-on reset (POR) circuit clears the COP prescaler 4096 ICLK cycles after power-up.

19.3.5 Internal Reset

An internal reset clears the COP prescaler and the COP counter.

19.3.6 Reset Vector Fetch

A reset vector fetch occurs when the vector address appears on the data bus. A reset vector fetch clears the COP prescaler.

19.3.7 COPD (COP Disable)

The COPD signal reflects the state of the COP disable bit (COPD) in the CONFIG1 register. (See Figure 19-2 . Configuration Register 1 (CONFIG1).)

Chapter 24
Ordering Information

24.1 Introduction

This section contains device ordering numbers.

24.2 MC Order Numbers

Table 24-1. MC Order Numbers

MC Order Number	RAM Size (bytes)	FLASH Size (bytes)	Package	Operating Temperature Range
MC68HC908AP64CB	2,048	62,368	42-pin SDIP	–40 to +85 °C
MC68HC908AP64CFB	2,048	62,368	44-pin QFP	–40 to +85 °C
MC68HC908AP64CFA	2,048	62,368	48-pin LQFP	–40 to +85 °C
MC68HC908AP32CB	2,048	32,768	42-pin SDIP	–40 to +85 °C
MC68HC908AP32CFB	2,048	32,768	44-pin QFP	–40 to +85 °C
MC68HC908AP32CFA	2,048	32,768	48-pin LQFP	–40 to +85 °C
MC68HC908AP16CB	1,024	16,384	42-pin SDIP	–40 to +85 °C
MC68HC908AP16CFB	1,024	16,384	44-pin QFP	–40 to +85 °C
MC68HC908AP16CFA	1,024	16,384	48-pin LQFP	–40 to +85 °C
MC68HC908AP8CB	1,024	8,192	42-pin SDIP	–40 to +85 °C
MC68HC908AP8CFB	1,024	8,192	44-pin QFP	–40 to +85 °C
MC68HC908AP8CFA	1,024	8,192	48-pin LQFP	–40 to +85 °C