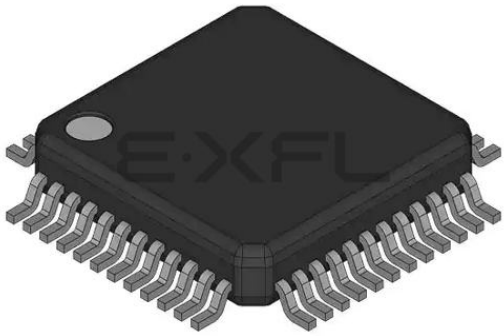


Welcome to E-XFL.COM

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	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K 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=mc908ap16cfae

MC68HC908AP64
MC68HC908AP32
MC68HC908AP16
MC68HC908AP8

Data Sheet

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

http://www.freescale.com

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
January 2007	4	15.7.2 ADC Clock Control Register — Changed “The ADC clock should be set to between 500kHz and 2MHz” to “The ADC clock should be set to between 500kHz and 1MHz”	254
August 2005	3	Table 22-4 . DC Electrical Characteristics (5V) — Updated V_{OL} values.	299
		Table 22-6 . Oscillator Specifications (5V) and Table 22-10 . Oscillator Specifications (3V) — Corrected internal oscillator clock frequency, f_{ICLK}. Updated crystal oscillator component values C_L, C_1, C_2, R_B, and R_S.	301, 305
October 2003	2.5	Added MC68HC908AP16/AP8 information throughout.	—
		Section 10. Monitor ROM (MON) — Corrected RAM address to $60.	167
		Section 24. Electrical Specifications — Added run and wait I_{DD} data for 8MHz at 3V.	421
August 2003	2.4	Section 24. Electrical Specifications — Updated stop I_{DD} data.	417, 421
July 2003	2.3	Removed MC68HC908AP16 references throughout.	—
		Table 1-2 . Pin Functions — Added footnote for V_{REG}.	30
		5.3 Configuration Register 1 (CONFIG1) — Clarified LVIPWRD and LVIREGD bits.	67
		Section 8. Clock Generator Module (CGM), 8.7.2 Stop Mode — Updated BSC bit behavior.	125
		10.5 ROM-Resident Routines — Corrected data size limits and control byte size for EE_READ and EE_WRITE.	168–193
		Figure 12-2 . Timebase Control Register (TBCR) — Corrected register address.	207
		Section 24. Electrical Specifications — Updated.	415
May 2003	2.2	Updated for f_{NOM} = 125kHz and filter components in CGM section.	101
		Updated electricals.	415

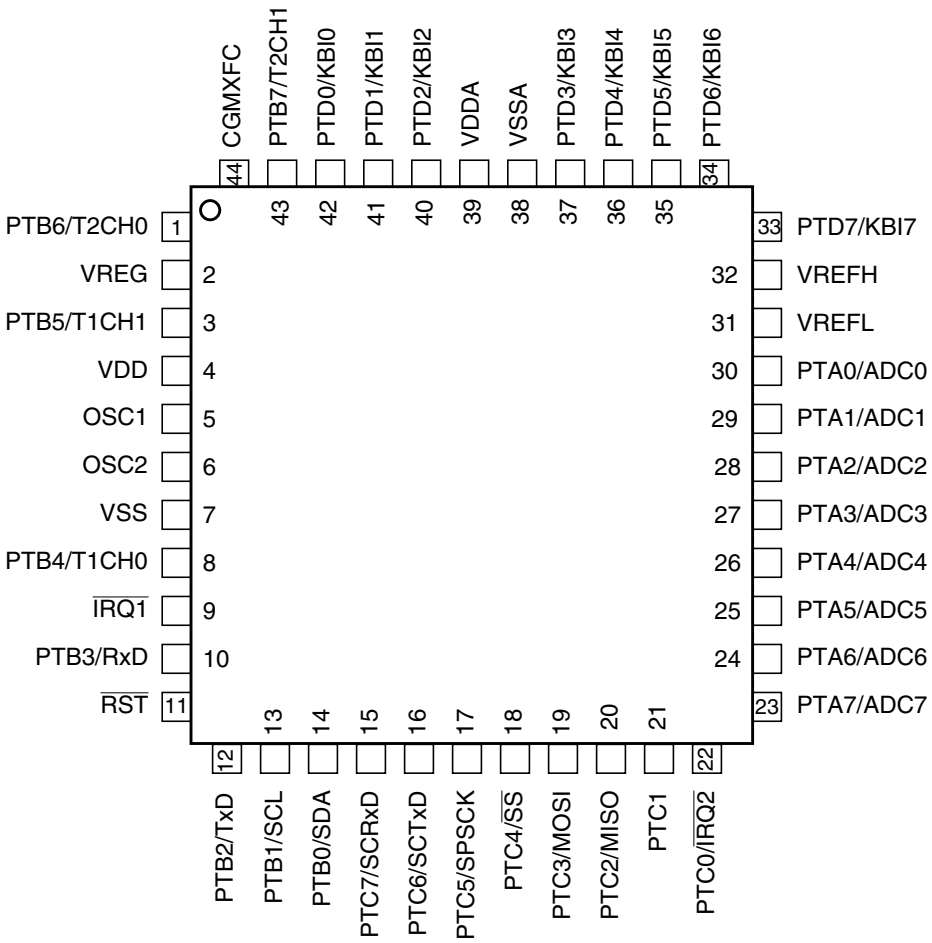
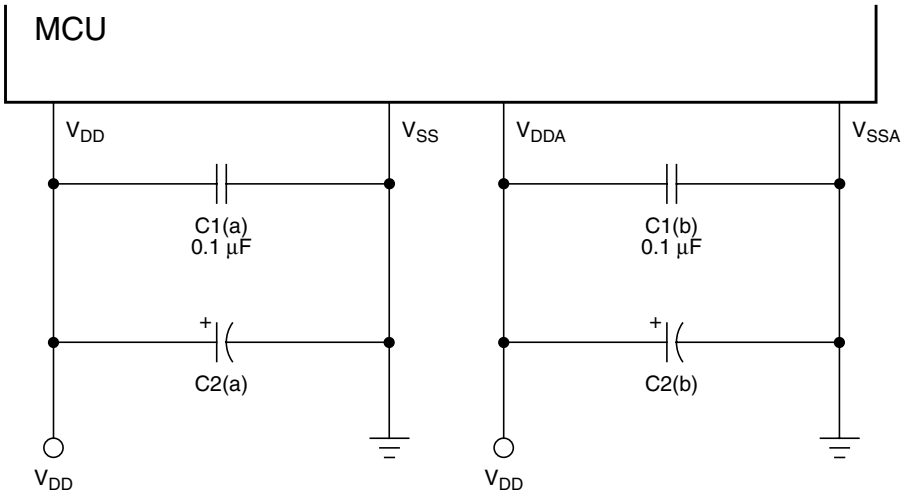


Figure 1-3. 44-Pin QFP Pin Assignments

V_{DDA} and V_{SSA} are the power supply and ground pins for the analog circuits of the MCU. These pins should be decoupled as per the digital power supply pins.



NOTE: Component values shown represent typical applications.

Figure 1-5. Power Supply Bypassing

1.7 Regulator Power Supply Configuration (VREG)

V_{REG} is the output from the on-chip regulator. All internal logics, except for the I/O pads, are powered by V_{REG} output. V_{REG} requires an external ceramic bypass capacitor of 100 nF as Figure 1-6 shows. Place the bypass capacitor as close to the V_{REG} pin as possible.

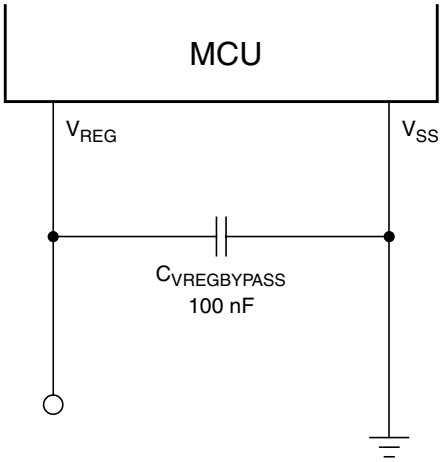


Figure 1-6. Regulator Power Supply Bypassing

Table 4-1. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Cycles
			V	H	I	N	Z	C				
BIT #*opr* BIT *opr* BIT *opr* BIT *opr*,X BIT *opr*,X BIT ,X BIT *opr*,SP BIT *opr*,SP	Bit Test	(A) & (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP1 SP2	A5 B5 C5 D5 E5 F5 9EE5 9ED5	ii dd hh ll ee ff ff ff ee ff	2 3 4 4 3 2 4 5
BLE *opr*	Branch if Less Than or Equal To (Signed Operands)	PC ← (PC) + 2 + *rel* ? (Z) ∣ (N ⊕ V)=1	–	–	–	–	–	–	REL	93	rr	3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? (C) ∣ (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT *opr*	Branch if Less Than (Signed Operands)	PC ← (PC) + 2 + *rel* ? (N ⊕ V) = 1	–	–	–	–	–	–	REL	91	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel*	–	–	–	–	–	–	REL	20	rr	3
BRCLR *n,opr,rel*	Branch if Bit *n* in M Clear	PC ← (PC) + 3 + *rel* ? (Mn) = 0	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN *rel*	Branch Never	PC ← (PC) + 2	–	–	–	–	–	–	REL	21	rr	3
BRSET *n,opr,rel*	Branch if Bit *n* in M Set	PC ← (PC) + 3 + *rel* ? (Mn) = 1	–	–	–	–	–	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET *n,opr*	Set Bit *n* in M	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	4 4 4 4 4 4 4 4

Table 8-8. READSP (Read Stack Pointer) Command

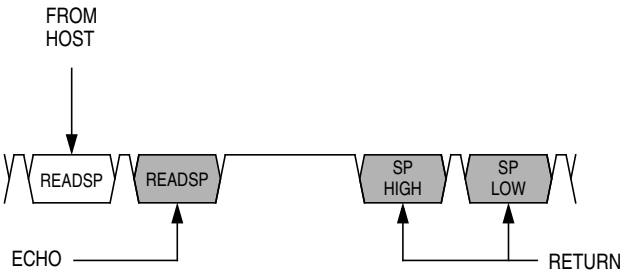
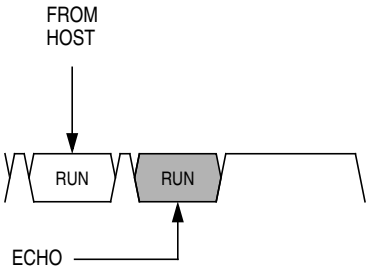
Description	Reads stack pointer
Operand	None
Data Returned	Returns incremented stack pointer value (SP + 1) in high-byte:low-byte order
Opcode	$0C
Command Sequence	

Table 8-9. RUN (Run User Program) Command

Description	Executes PULH and RTI instructions
Operand	None
Data Returned	None
Opcode	$28
Command Sequence	

The MCU executes the SWI and PSHH instructions when it enters monitor mode. The RUN command tells the MCU to execute the PULH and RTI instructions. Before sending the RUN command, the host can modify the stacked CPU registers to prepare to run the host program. The READSP command returns the incremented stack pointer value, SP + 1. The high and low bytes of the program counter are at addresses SP + 5 and SP + 6.

During the software execution, it does not consume any dedicated RAM location, the run-time heap will extend the system stack, all other RAM location will not be affected.

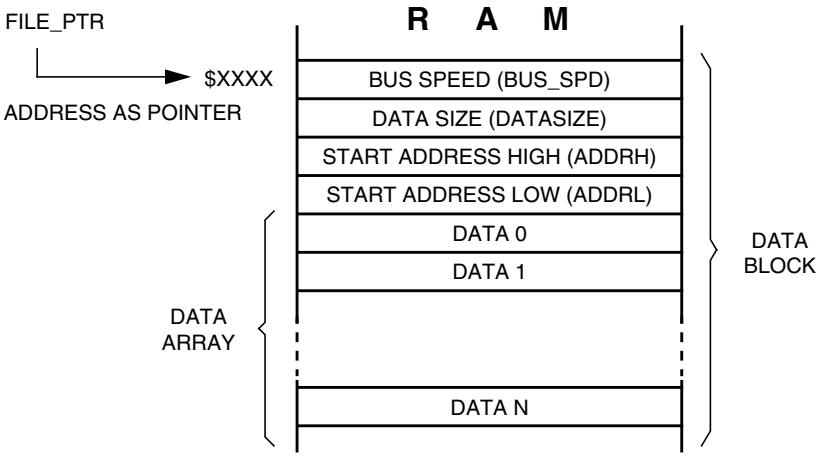


Figure 8-9. Data Block Format for ROM-Resident Routines

The control and data bytes are described below.

- **Bus speed** — This one byte indicates the operating bus speed of the MCU. The value of this byte should be equal to 4 times the bus speed. E.g., for a 4MHz bus, the value is 16 ($10). This control byte is useful where the MCU clock source is switched between the PLL clock and the crystal clock.
- **Data size** — This one byte indicates the number of bytes in the data array that are to be manipulated. The maximum data array size is 255. Routines EE_WRITE and EE_READ are restricted to manipulate a data array between 7 to 15 bytes. Whereas routines ERARNGE and MON_ERARNGE do not manipulate a data array, thus, this data size byte has no meaning.
- **Start address** — These two bytes, high byte followed by low byte, indicate the start address of the FLASH memory to be manipulated.
- **Data array** — This data array contains data that are to be manipulated. Data in this array are programmed to FLASH memory by the programming routines: PRGRNGE, MON_PRGRNGE, EE_WRITE. For the read routines: LDRNGE and EE_READ, data is read from FLASH and stored in this array.

8.5.1 PRGRNGE

PRGRNGE is used to program a range of FLASH locations with data loaded into the data array.

Table 8-11. PRGRNGE Routine

Routine Name	PRGRNGE
Routine Description	Program a range of locations
Calling Address	$FC34
Stack Used	15 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE) Start address high (ADDRH) Start address (ADDRL) Data 1 (DATA1) : Data N (DATAN)

8.5.6 EE_WRITE

EE_WRITE is used to write a set of data from the data array to FLASH.

Table 8-16. EE_WRITE Routine

Routine Name	EE_WRITE
Routine Description	Emulated EEPROM write. Data size ranges from 7 to 15 bytes at a time.
Calling Address	$FF36
Stack Used	30 bytes
Data Block Format	Bus speed (BUS_SPD) Data size (DATASIZE)(1) Starting address (ADDRH)(2) Starting address (ADDRL)(1) Data 1 : Data N

1. The minimum data size is 7 bytes. The maximum data size is 15 bytes.
2. The start address must be a page boundary start address.

The start location of the FLASH to be programmed is specified by the address ADDRH:ADDRL and the number of bytes in the data array is specified by DATASIZE. The minimum number of bytes that can be programmed in one routine call is 7 bytes, the maximum is 15 bytes. ADDRH:ADDRL must always be the start of boundary address (the page start address: $X000, $X200, $X400, $X600, $X800, $XA00, $XC00, or $XE00) and DATASIZE must be the same size when accessing the same page.

In some applications, the user may want to repeatedly store and read a set of data from an area of non-volatile memory. This is easily possible when using an EEPROM array. As the write and erase operations can be executed on a byte basis. For FLASH memory, the minimum erase size is the page — 512 bytes per page for MC68HC908AP64. If the data array size is less than the page size, writing and erasing to the same page cannot fully utilize the page. Unused locations in the page will be wasted. The EE_WRITE routine is designed to emulate the properties similar to the EEPROM. Allowing a more efficient use of the FLASH page for data storage.

When the user dedicates a page of FLASH for data storage, and the size of the data array defined, each call of the EE_WRTIE routine will automatically transfer the data in the data array (in RAM) to the next blank block of locations in the FLASH page. Once a page is filled up, the EE_WRITE routine automatically erases the page, and starts reuse the page again. In the 512-byte page, an 9-byte control block is used by the routine to monitor the utilization of the page. In effect, only 503 bytes are used for data storage. (see Figure 8-10). The page control operations are transparent to the user.

When using this routine to store a 8-byte data array, the FLASH page can be programmed 62 times before the an erase is required. In effect, the write/erase endurance is increased by 62 times. When a 15-byte data array is used, the write/erase endurance is increased by 33 times. Due to the FLASH page size limitation, the data array is limited from 7 bytes to 15 bytes.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0020	TIM1 Status and Control Register (T1SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$0021	TIM1 Counter Register High (T1CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0022	TIM1 Counter Register Low (T1CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0023	TIM Counter Modulo Register High (TMODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0024	TIM1 Counter Modulo Register Low (T1MODL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	1	1	1	1	1	1	1	1
$0025	TIM1 Channel 0 Status and Control Register (T1SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0026	TIM1 Channel 0 Register High (T1CH0H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$0027	TIM1 Channel 0 Register Low (T1CH0L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$0028	TIM1 Channel 1 Status and Control Register (T1SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0029	TIM1 Channel 1 Register High (T1CH1H)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	Indeterminate after reset							
$002A	TIM1 Channel 1 Register Low (T1CH1L)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$002B	TIM2 Status and Control Register (T2SC)	Read:	TOF	TOIE	TSTOP	0	0	PS2	PS1	PS0
		Write:	0			TRST				
		Reset:	0	0	1	0	0	0	0	0
$002C	TIM2 Counter Register High (T2CNTH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002D	TIM2 Counter Register Low (T2CNTL)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$002E	TIM2 Counter Modulo Register High (T2MODH)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
		Reset:	1	1	1	1	1	1	1	1

= Unimplemented

Figure 9-2. TIM I/O Register Summary (Sheet 1 of 2)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$002F	TIM2 Counter Modulo Register Low (T2MODL)	Read: / Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$0030	TIM2 Channel 0 Status and Control Register (T2SC0)	Read:	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0031	TIM2 Channel 0 Register High (T2CH0H)	Read: / Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0032	TIM2 Channel 0 Register Low (T2CH0L)	Read: / Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							
$0033	TIM2 Channel 1 Status and Control Register (T2SC1)	Read:	CH1F	CH1IE	0	MS1A	ELS1B	ELS1A	TOV1	CH1MAX
		Write:	0							
		Reset:	0	0	0	0	0	0	0	0
$0034	TIM2 Channel 1 Register High (T2CH1H)	Read: / Write:	Bit 15	14	13	12	11	10	9	Bit 8
		Reset:	Indeterminate after reset							
$0035	TIM2 Channel 1 Register Low (T2CH1L)	Read: / Write:	Bit 7	6	5	4	3	2	1	Bit 0
		Reset:	Indeterminate after reset							

= Unimplemented

Figure 9-2. TIM I/O Register Summary (Sheet 2 of 2)

9.4.1 TIM Counter Prescaler

The TIM clock source can be one of the seven prescaler outputs. The prescaler generates seven clock rates from the internal bus clock. The prescaler select bits, PS[2:0], in the TIM status and control register select the TIM clock source.

9.4.2 Input Capture

With the input capture function, the TIM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TIM latches the contents of the TIM counter into the TIM channel registers, TCHxH:TCHxL. The polarity of the active edge is programmable. Input captures can generate TIM CPU interrupt requests.

9.4.3 Output Compare

With the output compare function, the TIM can generate a periodic pulse with a programmable polarity, duration, and frequency. When the counter reaches the value in the registers of an output compare channel, the TIM can set, clear, or toggle the channel pin. Output compares can generate TIM CPU interrupt requests.

9.4.3.1 Unbuffered Output Compare

Any output compare channel can generate unbuffered output compare pulses as described in 9.4.3 Output Compare. The pulses are unbuffered because changing the output compare value requires writing the new value over the old value currently in the TIM channel registers.

11.4 Functional Description

Figure 11-2 shows the structure of the SCI module. The SCI allows full-duplex, asynchronous, NRZ serial communication among 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.

The baud rate clock source for the SCI can be selected via the configuration bit, SCIBDSRC, of the CONFIG2 register ($001D).

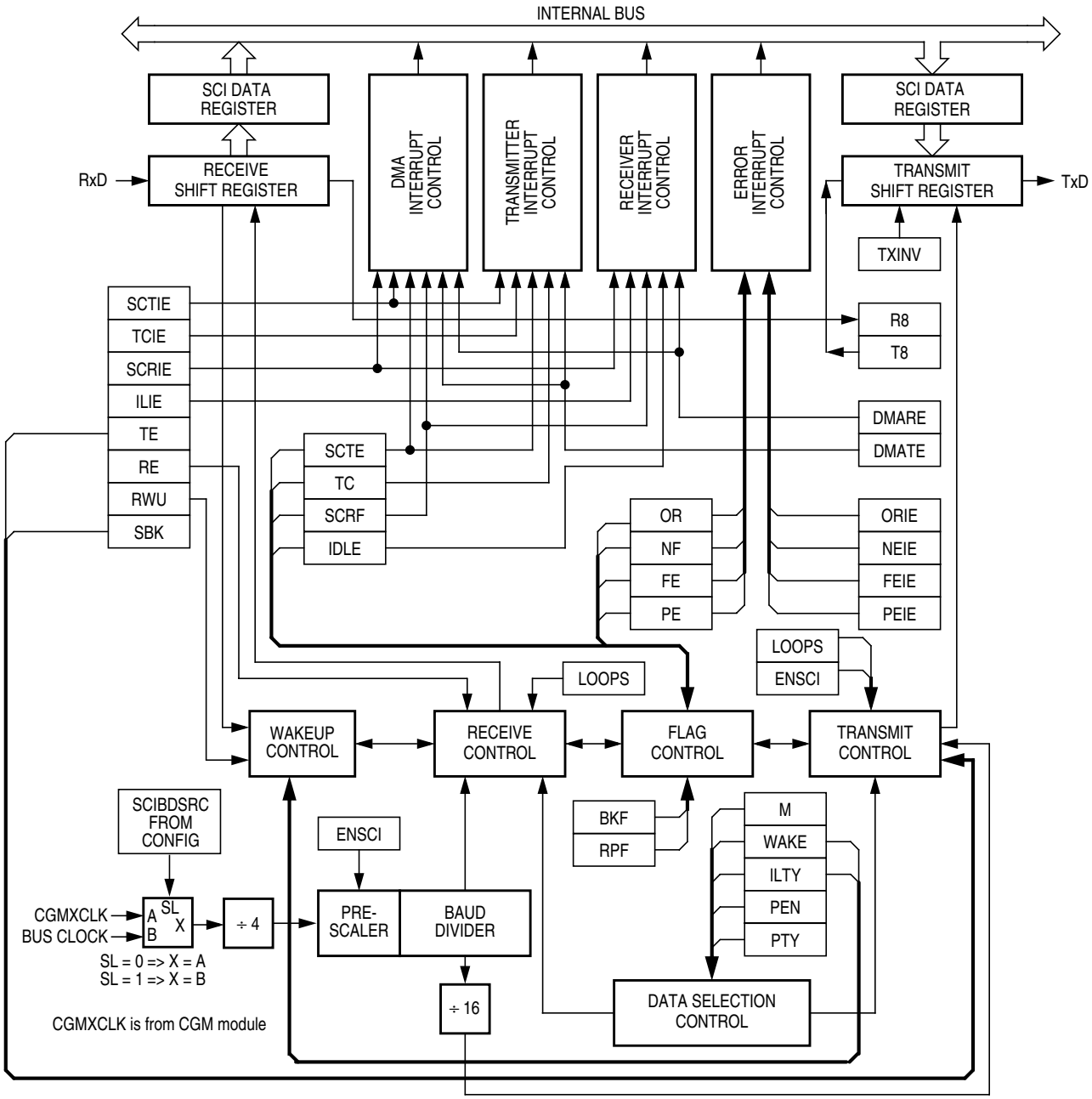


Figure 11-2. SCI Module Block Diagram

NEIE — Receiver Noise Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the noise error bit, NE. Reset clears NEIE.

1 = SCI error CPU interrupt requests from NE bit enabled
0 = SCI error CPU interrupt requests from NE bit disabled

FEIE — Receiver Framing Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the framing error bit, FE. Reset clears FEIE.

1 = SCI error CPU interrupt requests from FE bit enabled
0 = SCI error CPU interrupt requests from FE bit disabled

PEIE — Receiver Parity Error Interrupt Enable Bit

This read/write bit enables SCI error CPU interrupt requests generated by the parity error bit, PE. (See 11.8.4 SCI Status Register 1.) Reset clears PEIE.

1 = SCI error CPU interrupt requests from PE bit enabled
0 = SCI error CPU interrupt requests from PE bit disabled

11.8.4 SCI Status Register 1

SCI status register 1 (SCS1) contains flags to signal these conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error
- Parity error

Address: $0016

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
Write:								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented

Figure 11-12. SCI Status Register 1 (SCS1)

SCTE — SCI Transmitter Empty Bit

This clearable, read-only bit is set when the SCDR transfers a character to the transmit shift register. SCTE can generate an SCI transmitter CPU interrupt request. When the SCTIE bit in SCC2 is set, SCTE generates an SCI transmitter CPU interrupt request. In normal operation, clear the SCTE bit by reading SCS1 with SCTE set and then writing to SCDR. Reset sets the SCTE bit.

1 = SCDR data transferred to transmit shift register
0 = SCDR data not transferred to transmit shift register

The SCI transmitter empty bit, SCTE, in IRSCS1 becomes set when the IRSCDR transfers a byte to the transmit shift register. The SCTE bit indicates that the IRSCDR can accept new data from the internal data bus. If the SCI transmit interrupt enable bit, SCTIE, in IRSCC2 is also set, the SCTE bit generates a transmitter interrupt request.

When the transmit shift register is not transmitting a character, the TxD pin goes to the idle condition, logic 1. If at any time software clears the ENSCI bit in IRSCI control register 1 (IRSCC1), the transmitter and receiver relinquish control of the port pins.

12.5.2.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in IRSCC2 loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in IRSCC1. As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character.

The SCI recognizes a break character when a start bit is followed by eight or nine logic 0 data bits and a logic 0 where the stop bit should be.

Receiving a break character has the following effects on SCI registers:

- Sets the framing error bit (FE) in IRSCS1
- Sets the SCI receiver full bit (SCRF) in IRSCS1
- Clears the SCI data register (IRSCDR)
- Clears the R8 bit in IRSCC3
- Sets the break flag bit (BKF) in IRSCS2
- May set the overrun (OR), noise flag (NF), parity error (PE), or reception in progress flag (RPF) bits

12.5.2.4 Idle Characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in IRSCC1. The preamble is a synchronizing idle character that begins every transmission.

If the TE bit is cleared during a transmission, the TxD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the character currently being transmitted.

NOTE

When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current character shifts out to the TxD pin. Setting TE after the stop bit appears on TxD causes data previously written to the IRSCDR to be lost.

Toggle the TE bit for a queued idle character when the SCTE bit becomes set and just before writing the next byte to the IRSCDR.

12.5.2.5 Transmitter Interrupts

The following conditions can generate CPU interrupt requests from the SCI transmitter:

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit character with no errors is

$$\left|\frac{154 - 147}{154}\right| \times 100 = 4.54\%$$

For a 9-bit character, data sampling of the stop bit takes the receiver
10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 12-10, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles + 3 RT cycles = 163 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is

$$\left|\frac{170 - 163}{170}\right| \times 100 = 4.12\%$$

Fast Data Tolerance

Figure 12-11 shows how much a fast received character can be misaligned without causing a noise error or a framing error. The fast stop bit ends at RT10 instead of RT16 but is still there for the stop bit data samples at RT8, RT9, and RT10.

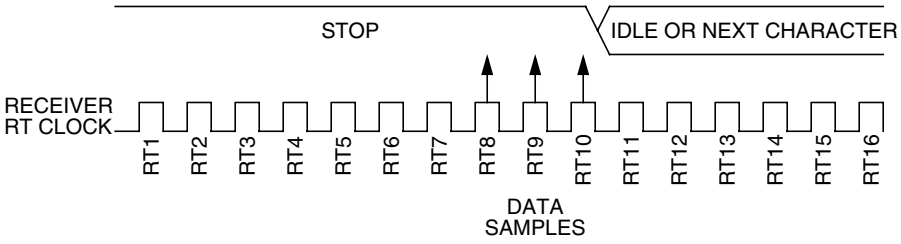


Figure 12-11. Fast Data

For an 8-bit character, data sampling of the stop bit takes the receiver
9 bit times × 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 12-11, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 10 bit times × 16 RT cycles = 160 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is

$$\left|\frac{154 - 160}{154}\right| \times 100 = 3.90\%$$

For a 9-bit character, data sampling of the stop bit takes the receiver
10 bit times × 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 12-11, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 11 bit times × 16 RT cycles = 176 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is

Table 12-9. IRSCI Baud Rate Selection Examples

SCP1 and SCP0	Prescaler Divisor (PD)	SCR2, SCR1, and SCR0	Baud Rate Divisor (BD)	Baud Rate (f_{BUS} = 4.9152 MHz)
00	1	000	1	—
00	1	001	2	—
00	1	010	4	76800
00	1	011	8	38400
00	1	100	16	19200
00	1	101	32	9600
00	1	110	64	4800
00	1	111	128	2400
01	3	000	1	—
01	3	001	2	51200
01	3	010	4	25600
01	3	011	8	12800
01	3	100	16	6400
01	3	101	32	3200
01	3	110	64	1600
01	3	111	128	800
10	4	000	1	76800
10	4	001	2	38400
10	4	010	4	19200
10	4	011	8	9600
10	4	100	16	4800
10	4	101	32	2400
10	4	110	64	1200
10	4	111	128	600
11	13	000	1	23632
11	13	001	2	11816
11	13	010	4	5908
11	13	011	8	2954
11	13	100	16	1477
11	13	101	32	739
11	13	110	64	369
11	13	111	128	185

- The shift register is cleared.
- The SPI state counter is cleared, making it ready for a new complete transmission.
- All the SPI port logic is defaulted back to being general-purpose I/O.

These items are reset only by a system reset:

- All control bits in the SPCR register
- All control bits in the SPSCR register (MODFEN, ERRIE, SPR1, and SPR0)
- The status flags SPRF, OVRF, and MODF

By not resetting the control bits when SPE is low, the user can clear SPE between transmissions without having to set all control bits again when SPE is set back high for the next transmission.

By not resetting the SPRF, OVRF, and MODF flags, the user can still service these interrupts after the SPI has been disabled. The user can disable the SPI by writing 0 to the SPE bit. The SPI can also be disabled by a mode fault occurring in an SPI that was configured as a master with the MODFEN bit set.

13.10 Low-Power Modes

The WAIT and STOP instructions put the MCU in low power-consumption standby modes.

13.10.1 Wait Mode

The SPI module remains active after the execution of a WAIT instruction. In wait mode the SPI module registers are not accessible by the CPU. Any enabled CPU interrupt request from the SPI module can bring the MCU out of wait mode.

If SPI module functions are not required during wait mode, reduce power consumption by disabling the SPI module before executing the WAIT instruction.

To exit wait mode when an overflow condition occurs, enable the OVRF bit to generate CPU interrupt requests by setting the error interrupt enable bit (ERRIE). (See 13.8 Interrupts.)

13.10.2 Stop Mode

The SPI module is inactive after the execution of a STOP instruction. The STOP instruction does not affect register conditions. SPI operation resumes after an external interrupt. If stop mode is exited by reset, any transfer in progress is aborted, and the SPI is reset.

13.11 SPI During Break Interrupts

The system integration module (SIM) controls whether status bits in other modules can be cleared during the break state. The BCFE bit in the SIM break flag control register (SBFCR) enables software to clear status bits during the break state. (See Chapter 7 System Integration Module (SIM).)

To allow software to clear status bits during a break interrupt, write a logic 1 to the BCFE bit. If a status bit is cleared during the break state, it remains cleared when the MCU exits the break state.

To protect status bits during the break state, write a logic 0 to the BCFE bit. With BCFE at logic 0 (its default state), software can read and write I/O registers during the break state without affecting status bits. Some status bits have a 2-step read/write clearing procedure. If software does the first step on such a bit before the break, the bit cannot change during the break state as long as BCFE is at logic 0. After the break, doing the second step clears the status bit.

16.2 Port A

Port A is an 8-bit special-function port that shares all of its pins with the analog-to-digital converter (ADC) module. Port A pins also have LED direct drive capability.

16.2.1 Port A Data Register (PTA)

The port A data register contains a data latch for each of the eight port A pins.

Address:	$0000							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTA7	PTA6	PTA5	PTA4	PTA3	PTA2	PTA1	PTA0
Reset:	Unaffected by reset							
Alternative Function:	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0
Additional Function:	LED drive	LED drive	LED drive	LED drive	LED drive	LED drive	LED drive	LED drive

Figure 16-2. Port A Data Register (PTA)

PTA[7:0] — Port A Data Bits

These read/write bits are software-programmable. Data direction of each port A pin is under the control of the corresponding bit in data direction register A. Reset has no effect on port A data.

ADC7–ADC0 — ADC Channels 7 to 0

ADC7–ADC0 are pins used for the input channels to the analog-to-digital converter module. The channel select bits, ADCH[4:0], in the ADC status and control register define which port pin will be used as an ADC input and overrides any control from the port I/O logic.

NOTE

Care must be taken when reading port A while applying analog voltages to ADC7–ADC0 pins. If the appropriate ADC channel is not enabled, excessive current drain may occur if analog voltages are applied to the PTAx/ADCx pin, while PTA is read as a digital input. Those ports not selected as analog input channels are considered digital I/O ports.

LED drive — Direct LED drive pins

PTA7–PTA0 pins can be configured for direct LED drive. See 16.2.3 Port-A LED Control Register (LEDA).

16.2.2 Data Direction Register (DDRA)

Data direction register A determines whether each port A pin is an input or an output. Writing a logic 1 to a DDRA bit enables the output buffer for the corresponding port A pin; a logic 0 disables the output buffer.

Address:	$0004							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
Reset:	0	0	0	0	0	0	0	0

Figure 16-3. Data Direction Register A (DDRA)

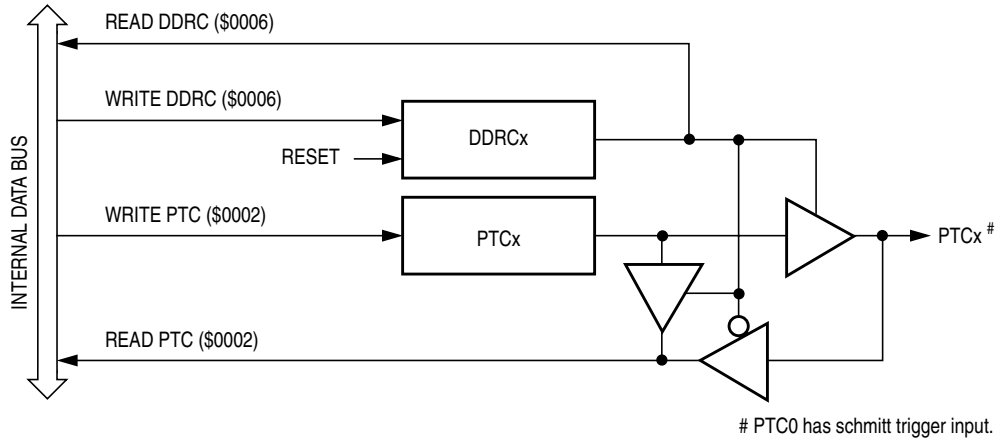


Figure 16-11. Port C I/O Circuit

When DDRCx is a logic 1, reading address $0002 reads the PTCx data latch. When DDRCx is a logic 0, reading address $0002 reads the voltage level on the pin. The data latch can always be written, regardless of the state of its data direction bit.

Table 16-4 summarizes the operation of the port C pins.

Table 16-4. Port C Pin Functions

DDRC Bit	PTC Bit	I/O Pin Mode	Accesses to DDRC	Accesses to PTC	
			Read/Write	Read	Write
0	X(1)	Input, Hi-Z(2)	DDRC[7:0]	Pin	PTC[7:0](3)
1	X	Output	DDRC[7:0]	PTC[7:0]	PTC[7:0]

1. X = don't care.
2. Hi-Z = high impedance.
3. Writing affects data register, but does not affect input.

16.5 Port D

Port D is an 8-bit special function port that shares all of its pins with the keyboard interrupt module.

16.5.1 Port D Data Register (PTD)

The port D data register contains a data latch for each of the eight port D pins.

Address:	$0003							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PTD7	PTD6	PTD5	PTD4	PTD3	PTD2	PTD1	PTD0
Reset:	Unaffected by reset							
Alternative Function:	KBI7	KBI6	KBI5	KBI4	KBI3	KBI2	KBI1	KBI0

Figure 16-12. Port D Data Register (PTD)

Chapter 20
Low-Voltage Inhibit (LVI)

20.1 Introduction

This section describes the low-voltage inhibit (LVI) module. The LVI module monitors the voltage on the V_{DD} pin and V_{REG} pin, and can force a reset when V_{DD} voltage falls below V_{TRIPF1}, or V_{REG} voltage falls below V_{TRIPF2}.

NOTE

The V_{REG} pin is the output of the internal voltage regulator and is guaranteed to meet operating specification as long as V_{DD} is within the MCU operating voltage.

The LVI feature is intended to provide the safe shutdown of the microcontroller and thus protection of related circuitry prior to any application V_{DD} voltage collapsing completely to an unsafe level. It is not intended that users operate the microcontroller at lower than the specified operating voltage, V_{DD}.

20.2 Features

Features of the LVI module include:

- Independent voltage monitoring circuits for V_{DD} and V_{REG}
- Independent disable for V_{DD} and V_{REG} LVI circuits
- Programmable LVI reset
- Programmable stop mode operation

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$FE0F	LVI Status Register (LVISR)	Read:	LVIOUT	0	0	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 20-1. LVI I/O Register Summary

20.3 Functional Description

Figure 20-2 shows the structure of the LVI module. The LVI is enabled out of reset. The LVI module contains independent bandgap reference circuit and comparator for monitoring the V_{DD} voltage and the V_{REG} voltage. An LVI reset performs a MCU internal reset and drives the $\overline{RST}$ pin low to provide low-voltage protection to external peripheral devices.

LVISTOP, LVIPWRD, LVIRSTD, and LVIREGD are in the CONFIG1 register. See Chapter 3 Configuration & Mask Option Registers (CONFIG & MOR) for details of the LVI configuration bits. Once