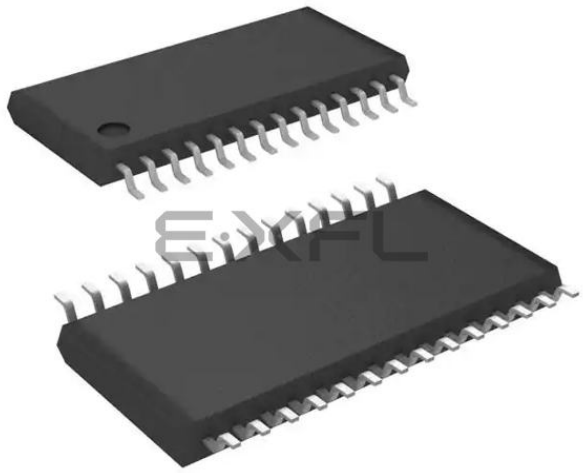


Welcome to E-XFL.COM

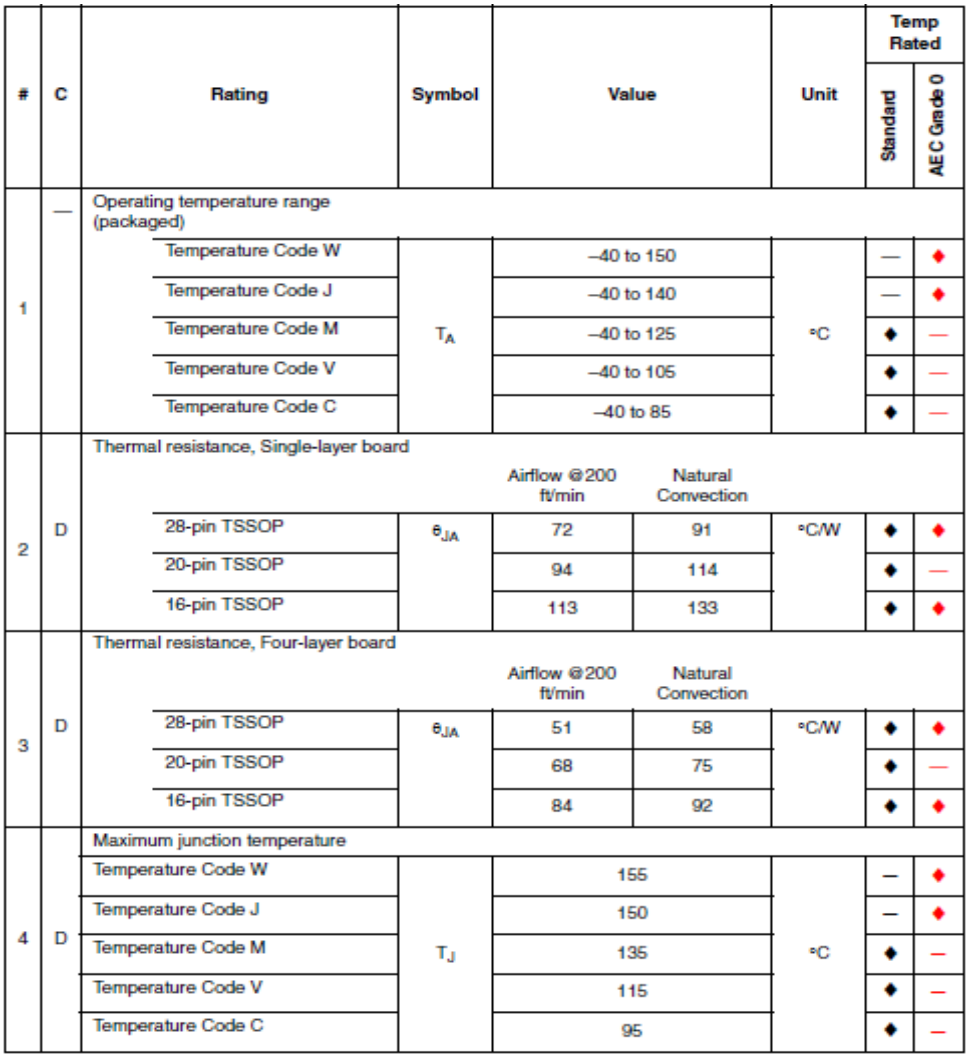
What is "Embedded - Microcontrollers"?

"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	S08
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, LINbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	22
Program Memory Size	32KB (32K 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 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	28-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	28-TSSOP
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/s9s08sg32e1vtl

Table 1. MC9S08SG32 Rev. 1 Addendum

<table>
<tr><th>Location</th><th>Description</th></tr>
<tr><td>Chapter “Electrical Characteristics”/Section “Thermal Characteristics”/Table A-3. Thermal Characteristics/Page 293</td><td>Update Table A-3. Thermal Characteristics as follows:
• —Change the value for row “Thermal resistance,Single-layer board/28-pin TSSOP/Airflow @200ft/min.” from 71 to 72 C/W
—Change the value for 16-pin TSSOP/Thermalresistance
1.Single layer board / Airflow @ 200ft/min. from 108 to 113 C/W.
2.Four layer board / Airflow @ 200ft/min. from 78 to 84 C/W.
• Update parameter 4 of Table “A-3.Thermal Characteristics” .
 
<table>
<tr><th rowspan="2">#</th><th rowspan="2">C</th><th rowspan="2">Rating</th><th rowspan="2">Symbol</th><th rowspan="2">Value</th><th rowspan="2">Unit</th><th colspan="2">Temp Rated</th></tr>
<tr><th>Standard</th><th>AEC Grade 0</th></tr>
<tr><td rowspan="6">1</td><td rowspan="6">—</td><td colspan="6">Operating temperature range (packaged)</td></tr>
<tr><td>Temperature Code W</td><td rowspan="5">T_A</td><td>–40 to 150</td><td rowspan="5">°C</td><td>—</td><td>♦</td></tr>
<tr><td>Temperature Code J</td><td>–40 to 140</td><td>—</td><td>♦</td></tr>
<tr><td>Temperature Code M</td><td>–40 to 125</td><td>♦</td><td>—</td></tr>
<tr><td>Temperature Code V</td><td>–40 to 105</td><td>♦</td><td>—</td></tr>
<tr><td>Temperature Code C</td><td>–40 to 85</td><td>♦</td><td>—</td></tr>
<tr><td rowspan="4">2</td><td rowspan="4">D</td><td colspan="6">Thermal resistance, Single-layer board</td></tr>
<tr><td></td><td></td><td>Airflow @200 ft/min</td><td>Natural Convection</td><td></td><td></td><td></td></tr>
<tr><td>28-pin TSSOP</td><td rowspan="3">θ_{JA}</td><td>72</td><td>91</td><td rowspan="3">°C/W</td><td>♦</td><td>♦</td></tr>
<tr><td>20-pin TSSOP</td><td>94</td><td>114</td><td>♦</td><td>—</td></tr>
<tr><td>16-pin TSSOP</td><td>113</td><td>133</td><td>♦</td><td>♦</td></tr>
<tr><td rowspan="4">3</td><td rowspan="4">D</td><td colspan="6">Thermal resistance, Four-layer board</td></tr>
<tr><td></td><td></td><td>Airflow @200 ft/min</td><td>Natural Convection</td><td></td><td></td><td></td></tr>
<tr><td>28-pin TSSOP</td><td rowspan="3">θ_{JA}</td><td>51</td><td>58</td><td rowspan="3">°C/W</td><td>♦</td><td>♦</td></tr>
<tr><td>20-pin TSSOP</td><td>68</td><td>75</td><td>♦</td><td>—</td></tr>
<tr><td>16-pin TSSOP</td><td>84</td><td>92</td><td>♦</td><td>♦</td></tr>
<tr><td rowspan="6">4</td><td rowspan="6">D</td><td colspan="6">Maximum junction temperature</td></tr>
<tr><td>Temperature Code W</td><td rowspan="5">T_J</td><td>155</td><td rowspan="5">°C</td><td>—</td><td>♦</td></tr>
<tr><td>Temperature Code J</td><td>150</td><td>—</td><td>♦</td></tr>
<tr><td>Temperature Code M</td><td>135</td><td>♦</td><td>—</td></tr>
<tr><td>Temperature Code V</td><td>115</td><td>♦</td><td>—</td></tr>
<tr><td>Temperature Code C</td><td>95</td><td>♦</td><td>—</td></tr>
</table>
</td></tr>
<tr><td>Chapter “Electrical Characteristics”/Section “DC Characteristics”/Table A-6. DC Characteristics/Page 298</td><td>In the Table “DC Characteristics” add note 11 and 12 for parameter #18.
Note 11: Device functionality is guaranteed between the LVD threshold VLVD0 and VDD Min. When VDD is below the minimum operating voltage (VDD Min), the analog parameters for the IO pins, ACMP and ADC, are not guaranteed to meet data sheet performance parameters.
Note 12: In addition to LVD, it is recommended to also use the LVW feature. LVW can trigger an interrupt and be used as an indicator to warn that the VDD is dropping,so that the software can take actions accordingly before the VDD drops below VDD Min.</td></tr>
</table>

MC9S08SG32 Data Sheet

Covers MC9S08SG32
MC9S08SG16

MC9S08SG32
Rev. 8
5/2010

1.2 MCU Block Diagram

The block diagram in Figure 1-1 shows the structure of the MC9S08SG32 Series MCU.

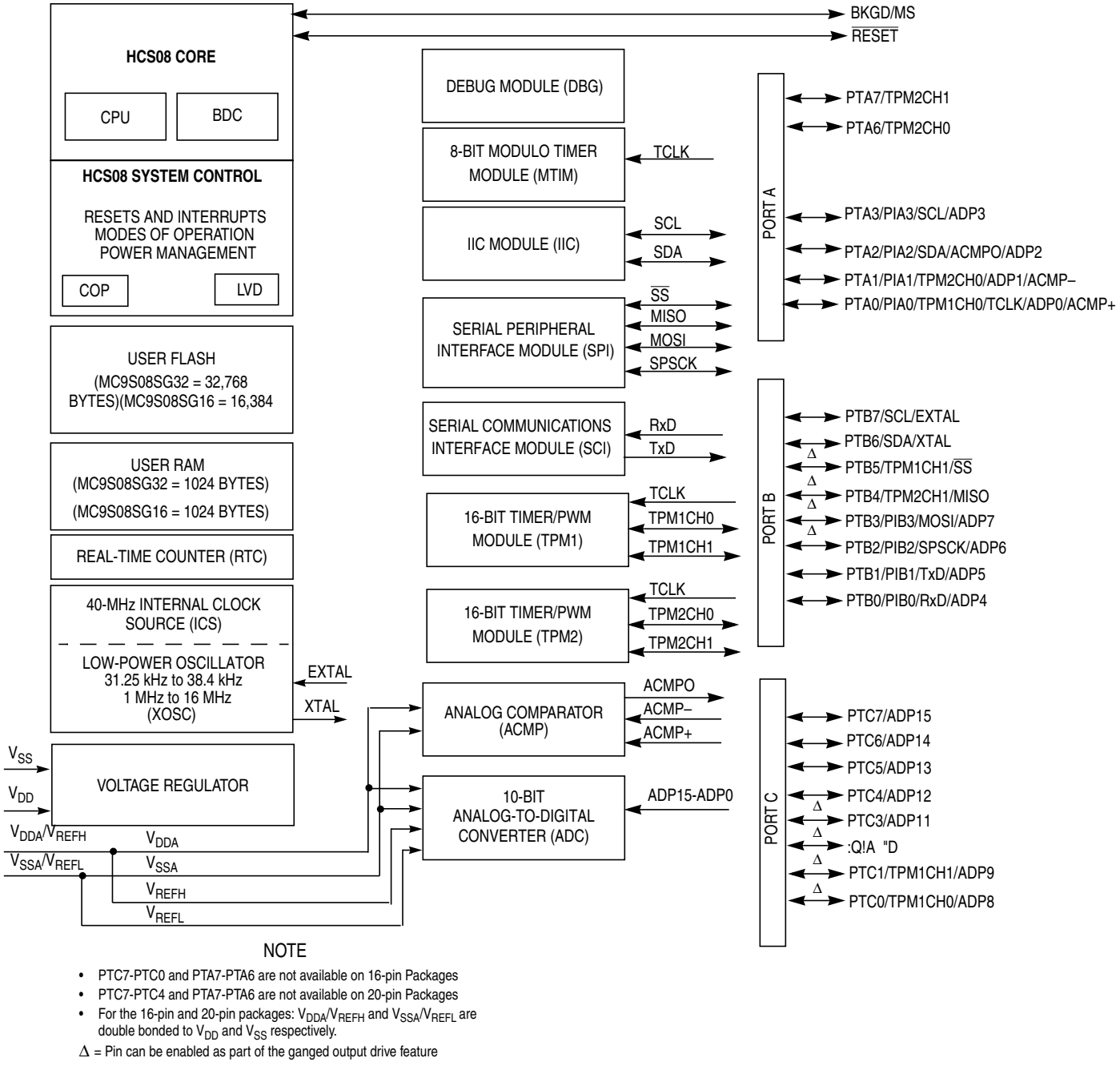


Figure 1-1. MC9S08SG32 Series Block Diagram

Whenever any reset is initiated (whether from an external signal or from an internal system), the $\overline{\text{RESET}}$ pin is driven low for about 66 bus cycles. The reset circuitry decodes the cause of reset and records it by setting a corresponding bit in the system reset status register (SRS).

NOTE

- This pin does not contain a clamp diode to V_{DD} and should not be driven above V_{DD}.
- The voltage measured on the internally pulled up $\overline{\text{RESET}}$ pin will not be pulled to V_{DD}. The internal gates connected to this pin are pulled to V_{DD}. If the $\overline{\text{RESET}}$ pin is required to drive to a V_{DD} level, an external pullup should be used.
- In EMC-sensitive applications, an external RC filter is recommended on the $\overline{\text{RESET}}$. See Figure 2-4 for an example.

2.2.4 Background / Mode Select (BKGD/MS)

During a power-on-reset (POR) or background debug force reset (see Section 5.7.2, “System Background Debug Force Reset Register (SBDFR),” for more information), the BKGD/MS pin functions as a mode select pin. Immediately after any reset, the pin functions as the background pin and can be used for background debug communication. The BKGD/MS pin contains an internal pullup device.

If nothing is connected to this pin, the MCU enters normal operating mode at the rising edge of the internal reset after a POR or force BDC reset. If a debug system is connected to the 6-pin standard background debug header, it can hold BKGD/MS low during a POR or immediately after issuing a background debug force reset, which will force the MCU to active background mode.

The BKGD pin is used primarily for background debug controller (BDC) communications using a custom protocol that uses 16 clock cycles of the target MCU’s BDC clock per bit time. The target MCU’s BDC clock could be as fast as the maximum bus clock rate, so there must never be any significant capacitance connected to the BKGD/MS pin that could interfere with background serial communications.

Although the BKGD pin is a pseudo open-drain pin, the background debug communication protocol provides brief, actively driven, high speedup pulses to ensure fast rise times. Small capacitances from cables and the absolute value of the internal pullup device play almost no role in determining rise and fall times on the BKGD pin.

2.2.5 General-Purpose I/O and Peripheral Ports

The MC9S08SG32 Series of MCUs support up to 22 general-purpose I/O pins which are shared with on-chip peripheral functions (timers, serial I/O, ADC, etc.).

When a port pin is configured as a general-purpose output or a peripheral uses the port pin as an output, software can select one of two drive strengths and enable or disable slew rate control. When a port pin is configured as a general-purpose input or a peripheral uses the port pin as an input, software can enable a pull-up device. Immediately after reset, all of these pins are configured as high-impedance general-purpose inputs with internal pull-up devices disabled.

The first byte of a series of sequential bytes being programmed in burst mode will take the same amount of time to program as a byte programmed in standard mode. Subsequent bytes will program in the burst program time provided that the conditions above are met. In the case the next sequential address is the beginning of a new row, the program time for that byte will be the standard time instead of the burst time. This is because the high voltage to the array must be disabled and then enabled again. If a new burst command has not been queued before the current command completes, then the charge pump will be disabled and high voltage removed from the array.

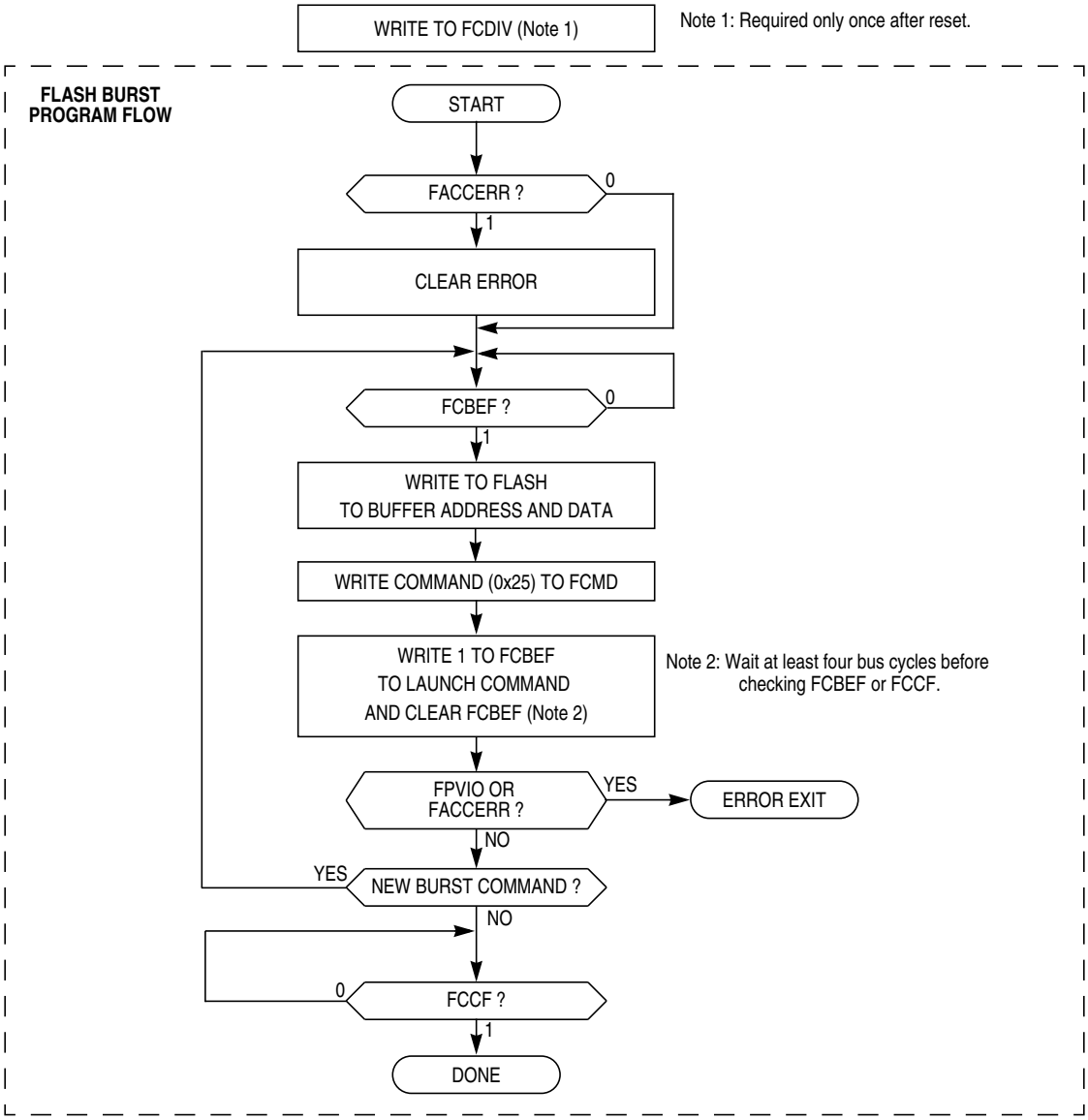


Figure 4-3. FLASH Burst Program Flowchart

4.5.5 Access Errors

An access error occurs whenever the command execution protocol is violated.

Any of the following specific actions will cause the access error flag (FACCERR) in FSTAT to be set. Before any command can be processed, write a 1 to FACCERR in FSTAT to clear the access error flag (FACCERR).

- Writing to a FLASH address before the internal FLASH clock frequency has been set by writing to the FCDIV register
- Writing to a FLASH address while FCBEF is not set (A new command cannot be started until the command buffer is empty.)
- Writing a second time to a FLASH address before launching the previous command (There is only one write to FLASH for every command.)
- Writing a second time to FCMD before launching the previous command (There is only one write to FCMD for every command.)
- Writing to any FLASH control register other than FCMD after writing to a FLASH address
- Writing any command code other than the five allowed codes (0x05, 0x20, 0x25, 0x40, or 0x41) to FCMD
- Writing any FLASH control register other than the write to FSTAT (to clear FCBEF and launch the command) after writing the command to FCMD
- The MCU enters stop mode while a program or erase command is in progress (The command is aborted.)
- Writing the byte program, burst program, or page erase command code (0x20, 0x25, or 0x40) with a background debug command while the MCU is secured (The background debug controller can only do blank check and mass erase commands when the MCU is secure.)
- Writing 0 to FCBEF to cancel a partial command

4.5.6 FLASH Block Protection

The block protection feature prevents the protected region of FLASH from program or erase changes. Block protection is controlled through the FLASH protection register (FPROT). When enabled, block protection begins at any 512 byte boundary below the last address of FLASH, 0xFFFF. (See Section 4.7.4, “FLASH Protection Register (FPROT and NVPROT)”).

After exit from reset, FPROT is loaded with the contents of the NVPROT location, which is in the nonvolatile register block of the FLASH memory. FPROT cannot be changed directly from application software so a runaway program cannot alter the block protection settings. Because NVPROT is within the last 512 bytes of FLASH, if any amount of memory is protected, NVPROT is itself protected and cannot be altered (intentionally or unintentionally) by the application software. FPROT can be written through background debug commands, which allows a way to erase and reprogram a protected FLASH memory.

The block protection mechanism is illustrated in Figure 4-4. The FPS bits are used as the upper bits of the last address of unprotected memory. This address is formed by concatenating FPS7:FPS1 with logic 1 bits as shown. For example, to protect the last 1536 bytes of memory (addresses 0xFA00 through 0xFFFF), the FPS bits must be set to 1111 100, which results in the value 0xF9FF as the last address of unprotected

It is a good programming practice to write to the port data register before changing the direction of a port pin to become an output. This ensures that the pin will not be driven momentarily with an old data value that happened to be in the port data register.

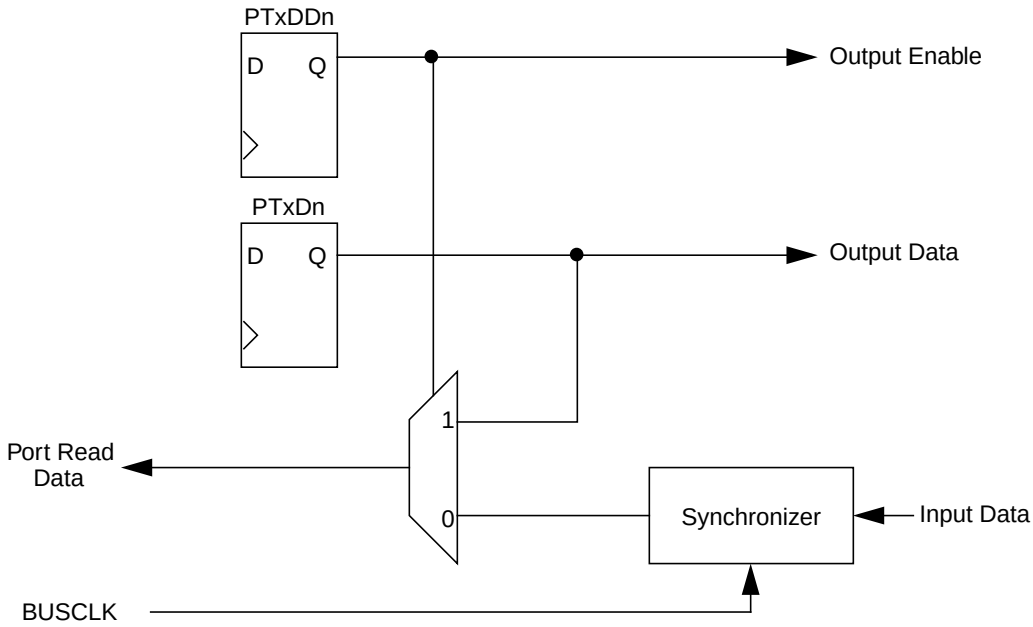


Figure 6-1. Parallel I/O Block Diagram

6.2 Pull-up, Slew Rate, and Drive Strength

Associated with the parallel I/O ports is a set of registers located in the high page register space that operate independently of the parallel I/O registers. These registers are used to control pull-ups, slew rate, and drive strength for the pins.

An internal pull-up device can be enabled for each port pin by setting the corresponding bit in the pull-up enable register (PTxPEn). The pull-up device is disabled if the pin is configured as an output by the parallel I/O control logic or any shared peripheral function regardless of the state of the corresponding pull-up enable register bit. The pull-up device is also disabled if the pin is controlled by an analog function.

Slew rate control can be enabled for each port pin by setting the corresponding bit in the slew rate control register (PTxSEn). When enabled, slew control limits the rate at which an output can transition in order to reduce EMC emissions. Slew rate control has no effect on pins that are configured as inputs.

An output pin can be selected to have high output drive strength by setting the corresponding bit in the drive strength select register (PTxDSn). When high drive is selected, a pin is capable of sourcing and sinking greater current. Even though every I/O pin can be selected as high drive, the user must ensure that the total current source and sink limits for the MCU are not exceeded. Drive strength selection is intended to affect the DC behavior of I/O pins. However, the AC behavior is also affected. High drive allows a pin to drive a greater load with the same switching speed as a low drive enabled pin into a smaller load. Because of this, the EMC emissions may be affected by enabling pins as high drive.

6.6.1 Port A Registers

Port A is controlled by the registers listed below.

6.6.1.1 Port A Data Register (PTAD)

	7	6	5	4	3	2	1	0
R W	PTAD7	PTAD6	R	R	PTAD3	PTAD2	PTAD1	PTAD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-3. Port A Data Register (PTAD)

Table 6-2. PTAD Register Field Descriptions

Field	Description
7:6, 3:0 PTAD[7:6, 3:0]	**Port A Data Register Bits** — For port A pins that are inputs, reads return the logic level on the pin. For port A pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port A pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTAD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pull-ups/pull-downs disabled.
5:4 Reserved	**Reserved Bits** — These bits are unused on this MCU, writes have no affect and could read as 1s or 0s.

6.6.1.2 Port A Data Direction Register (PTADD)

	7	6	5	4	3	2	1	0
R W	PTADD7	PTADD6	R	R	PTADD3	PTADD2	PTADD1	PTADD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-4. Port A Data Direction Register (PTADD)

Table 6-3. PTADD Register Field Descriptions

Field	Description
7:6, 3:0 PTADD[7:6, 3:0]	**Data Direction for Port A Bits** — These read/write bits control the direction of port A pins and what is read for PTAD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port A bit n and PTAD reads return the contents of PTADn.
5:4 Reserved	**Reserved Bits** — These bits are unused on this MCU, writes have no affect and could read as 1s or 0s.

6.6.1.7 Port A Interrupt Pin Select Register (PTAPS)

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTAPS3	PTAPS2	PTAPS1	PTAPS0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-9. Port A Interrupt Pin Select Register (PTAPS)

Table 6-8. PTAPS Register Field Descriptions

Field	Description
3:0 PTAPS[3:0]	**Port A Interrupt Pin Selects** — Each of the PTAPSn bits enable the corresponding port A interrupt pin. 0 Pin not enabled as interrupt. 1 Pin enabled as interrupt.

6.6.1.8 Port A Interrupt Edge Select Register (PTAES)

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTAES3	PTAES2	PTAES1	PTAES0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-10. Port A Edge Select Register (PTAES)

Table 6-9. PTAES Register Field Descriptions

Field	Description
3:0 PTAES[3:0]	**Port A Edge Selects** — Each of the PTAESn bits serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pull-up or pull-down device if enabled. 0 A pull-up device is connected to the associated pin and detects falling edge/low level for interrupt generation. 1 A pull-down device is connected to the associated pin and detects rising edge/high level for interrupt generation.

7.3.5 Extended Addressing Mode (EXT)

In extended addressing mode, the full 16-bit address of the operand is located in the next two bytes of program memory after the opcode (high byte first).

7.3.6 Indexed Addressing Mode

Indexed addressing mode has seven variations including five that use the 16-bit H:X index register pair and two that use the stack pointer as the base reference.

7.3.6.1 Indexed, No Offset (IX)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair as the address of the operand needed to complete the instruction.

7.3.6.2 Indexed, No Offset with Post Increment (IX+)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair as the address of the operand needed to complete the instruction. The index register pair is then incremented (H:X = H:X + 0x0001) after the operand has been fetched. This addressing mode is only used for MOV and CBEQ instructions.

7.3.6.3 Indexed, 8-Bit Offset (IX1)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction.

7.3.6.4 Indexed, 8-Bit Offset with Post Increment (IX1+)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction. The index register pair is then incremented (H:X = H:X + 0x0001) after the operand has been fetched. This addressing mode is used only for the CBEQ instruction.

7.3.6.5 Indexed, 16-Bit Offset (IX2)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus a 16-bit offset included in the instruction as the address of the operand needed to complete the instruction.

7.3.6.6 SP-Relative, 8-Bit Offset (SP1)

This variation of indexed addressing uses the 16-bit value in the stack pointer (SP) plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction.

- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock (ADACK) and averaging. Noise that is synchronous to ADCK cannot be averaged out.

9.6.2.4 Code Width and Quantization Error

The ADC quantizes the ideal straight-line transfer function into 1024 steps (in 10-bit mode). Each step ideally has the same height (1 code) and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N bit converter (in this case N can be 8 or 10), defined as 1LSB, is:

$$1_{LSB} = (V_{REFH} - V_{REFL}) / 2^N \qquad \textit{Eqn. 9-2}$$

There is an inherent quantization error due to the digitization of the result. For 8-bit or 10-bit conversions the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be ± 1/2LSB in 8- or 10-bit mode. As a consequence, however, the code width of the first (0x000) conversion is only 1/2LSB and the code width of the last (0xFF or 0x3FF) is 1.5LSB.

9.6.2.5 Linearity Errors

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors but the system should be aware of them because they affect overall accuracy. These errors are:

- Zero-scale error (E_{ZS}) (sometimes called offset) — This error is defined as the difference between the actual code width of the first conversion and the ideal code width (1/2LSB). If the first conversion is 0x001, then the difference between the actual 0x001 code width and its ideal (1LSB) is used.
- Full-scale error (E_{FS}) — This error is defined as the difference between the actual code width of the last conversion and the ideal code width (1.5LSB). If the last conversion is 0x3FE, then the difference between the actual 0x3FE code width and its ideal (1LSB) is used.
- Differential non-linearity (DNL) — This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL) — This error is defined as the highest-value the (absolute value of the) running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE) — This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

9.6.2.6 Code Jitter, Non-Monotonicity, and Missing Codes

Analog-to-digital converters are susceptible to three special forms of error. These are code jitter, non-monotonicity, and missing codes.

Code jitter is when, at certain points, a given input voltage converts to one of two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the

10.3.4 IIC Status Register (IICS)

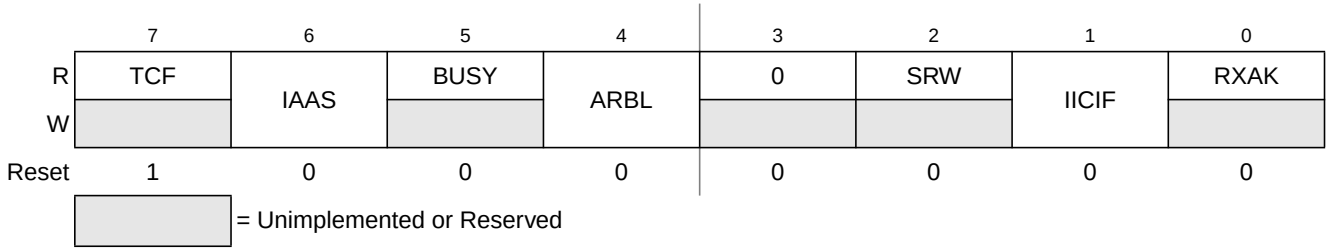


Figure 10-6. IIC Status Register (IICS)

Table 10-7. IICS Field Descriptions

Field	Description
7 TCF	**Transfer Complete Flag.** This bit is set on the completion of a byte transfer. This bit is only valid during or immediately following a transfer to the IIC module or from the IIC module.The TCF bit is cleared by reading the IICD register in receive mode or writing to the IICD in transmit mode. 0 Transfer in progress 1 Transfer complete
6 IAAS	**Addressed as a Slave.** The IAAS bit is set when the calling address matches the programmed slave address or when the GCAEN bit is set and a general call is received. Writing the IICC register clears this bit. 0 Not addressed 1 Addressed as a slave
5 BUSY	**Bus Busy.** The BUSY bit indicates the status of the bus regardless of slave or master mode. The BUSY bit is set when a start signal is detected and cleared when a stop signal is detected. 0 Bus is idle 1 Bus is busy
4 ARBL	**Arbitration Lost.** This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software by writing a 1 to it. 0 Standard bus operation 1 Loss of arbitration
2 SRW	**Slave Read/Write.** When addressed as a slave, the SRW bit indicates the value of the R/W command bit of the calling address sent to the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IICIF	**IIC Interrupt Flag.** The IICIF bit is set when an interrupt is pending. This bit must be cleared by software, by writing a 1 to it in the interrupt routine. One of the following events can set the IICIF bit: • One byte transfer completes • Match of slave address to calling address • Arbitration lost 0 No interrupt pending 1 Interrupt pending
0 RXAK	**Receive Acknowledge**. When the RXAK bit is low, it indicates an acknowledge signal has been received after the completion of one byte of data transmission on the bus. If the RXAK bit is high it means that no acknowledge signal is detected. 0 Acknowledge received 1 No acknowledge received

Chapter 15
Serial Peripheral Interface (S08SPIV3)

15.1 Introduction

The serial peripheral interface (SPI) module provides for full-duplex, synchronous, serial communication between the MCU and peripheral devices. These peripheral devices can include other microcontrollers, analog-to-digital converters, shift registers, sensors, memories, and so forth.

The SPI runs at a baud rate up to that of the bus clock divided by two in master mode and bus clock divided by four in slave mode. The SPI operation can be interrupt driven or software can poll the status flags.

All devices in the MC9S08SG32 Series MCUs contain one SPI module, as shown in the following block diagram. Figure 15-1 shows the MC9S08SG32 Series block diagram with the SPI modules highlighted.

15.1.1 Features

Features of the SPI module include:

- Master or slave mode operation
- Full-duplex or single-wire bidirectional option
- Programmable transmit bit rate
- Double-buffered transmit and receive
- Serial clock phase and polarity options
- Slave select output
- Selectable MSB-first or LSB-first shifting

15.1.2 Block Diagrams

This section includes block diagrams showing SPI system connections, the internal organization of the SPI module, and the SPI clock dividers that control the master mode bit rate.

15.1.2.1 SPI System Block Diagram

Figure 15-2 shows the SPI modules of two MCUs connected in a master-slave arrangement. The master device initiates all SPI data transfers. During a transfer, the master shifts data out (on the MOSI pin) to the slave while simultaneously shifting data in (on the MISO pin) from the slave. The transfer effectively exchanges the data that was in the SPI shift registers of the two SPI systems. The SPSCK signal is a clock output from the master and an input to the slave. The slave device must be selected by a low level on the slave select input ($\overline{SS}$ pin). In this system, the master device has configured its $\overline{SS}$ pin as an optional slave select output.

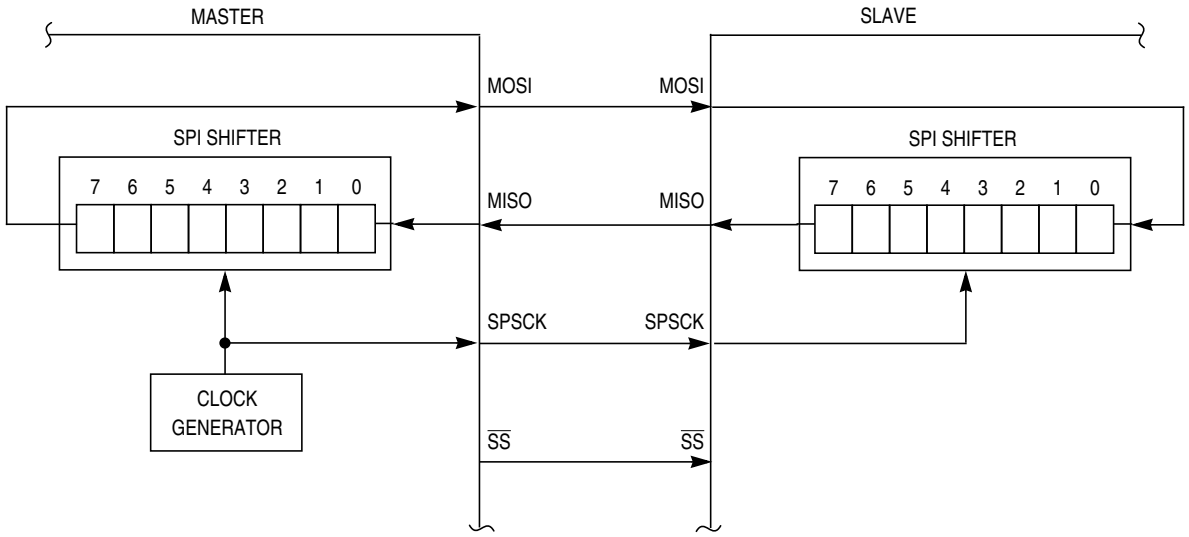


Figure 15-2. SPI System Connections

...
configure the channel pin as output port pin and set the output pin;
configure the channel to generate the EPWM signal but keep ELSnB:ELSnA as 00;
configure the other registers (TPMxMODH, TPMxMODL, TPMxCnVH, TPMxCnVL, ...);
configure CLKSB:CLKSA bits (TPM v3 starts to generate the high-true EPWM signal, however TPM does not control the channel pin, so the EPWM signal is not available);
wait until the TOF is set (or use the TOF interrupt);
enable the channel output by configuring ELSnB:ELSnA bits (now EPWM signal is available);
...

the host must perform ((8 – CNT) – 1) dummy reads of the FIFO to advance it to the first significant entry in the FIFO.

In most trigger modes, the information stored in the FIFO consists of 16-bit change-of-flow addresses. In these cases, read DBGFH then DBGFL to get one coherent word of information out of the FIFO. Reading DBGFL (the low-order byte of the FIFO data port) causes the FIFO to shift so the next word of information is available at the FIFO data port. In the event-only trigger modes (see Section 17.3.5, “Trigger Modes”), 8-bit data information is stored into the FIFO. In these cases, the high-order half of the FIFO (DBGFH) is not used and data is read out of the FIFO by simply reading DBGFL. Each time DBGFL is read, the FIFO is shifted so the next data value is available through the FIFO data port at DBGFL.

In trigger modes where the FIFO is storing change-of-flow addresses, there is a delay between CPU addresses and the input side of the FIFO. Because of this delay, if the trigger event itself is a change-of-flow address or a change-of-flow address appears during the next two bus cycles after a trigger event starts the FIFO, it will not be saved into the FIFO. In the case of an end-trace, if the trigger event is a change-of-flow, it will be saved as the last change-of-flow entry for that debug run.

The FIFO can also be used to generate a profile of executed instruction addresses when the debugger is not armed. When ARM = 0, reading DBGFL causes the address of the most-recently fetched opcode to be saved in the FIFO. To use the profiling feature, a host debugger would read addresses out of the FIFO by reading DBGFH then DBGFL at regular periodic intervals. The first eight values would be discarded because they correspond to the eight DBGFL reads needed to initially fill the FIFO. Additional periodic reads of DBGFH and DBGFL return delayed information about executed instructions so the host debugger can develop a profile of executed instruction addresses.

17.3.3 Change-of-Flow Information

To minimize the amount of information stored in the FIFO, only information related to instructions that cause a change to the normal sequential execution of instructions is stored. With knowledge of the source and object code program stored in the target system, an external debugger system can reconstruct the path of execution through many instructions from the change-of-flow information stored in the FIFO.

For conditional branch instructions where the branch is taken (branch condition was true), the source address is stored (the address of the conditional branch opcode). Because BRA and BRN instructions are not conditional, these events do not cause change-of-flow information to be stored in the FIFO.

Indirect JMP and JSR instructions use the current contents of the H:X index register pair to determine the destination address, so the debug system stores the run-time destination address for any indirect JMP or JSR. For interrupts, RTI, or RTS, the destination address is stored in the FIFO as change-of-flow information.

17.3.4 Tag vs. Force Breakpoints and Triggers

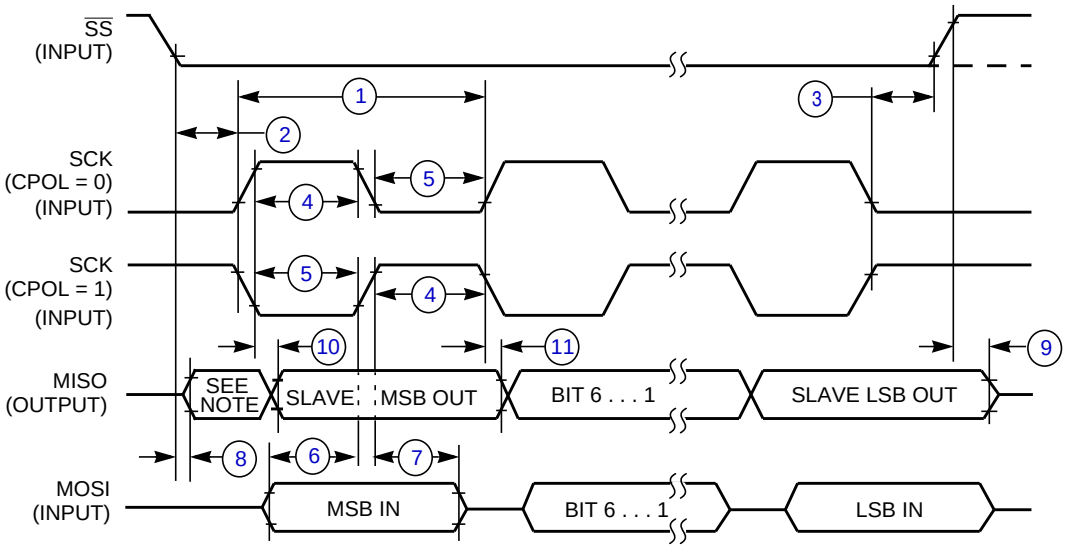
Tagging is a term that refers to identifying an instruction opcode as it is fetched into the instruction queue, but not taking any other action until and unless that instruction is actually executed by the CPU. This distinction is important because any change-of-flow from a jump, branch, subroutine call, or interrupt causes some instructions that have been fetched into the instruction queue to be thrown away without being executed.

A.7 Supply Current Characteristics

This section includes information about power supply current in various operating modes.

Table A-7. Supply Current Characteristics

#	C	Parameter	Symbol	V_{DD} (V)	Typ[1]	Max[2]	Unit	Temp Rated	
								Standard	AEC Grade 0
1	C	Run supply current[3] measured at (CPU clock = 4 MHz, f_{Bus} = 2 MHz)	RI_{DD}	5	1.4	3	mA	♦	♦
	C			3	1.3	2.5	mA	♦	♦
2	P	Run supply current[3] measured at (CPU clock = 16 MHz, f_{Bus} = 8 MHz)	RI_{DD}	5	4.7	7.5	mA	♦	♦
	C			3	4.6	7	mA	♦	♦
3	C	Run supply current[4] measured at (CPU clock = 32 MHz, f_{Bus} = 16MHz)	RI_{DD}	5	8.9	10	mA	♦	♦
	C			3	8.7	9.6	mA	♦	♦
4		Stop3 mode supply current							
	C	–40°C (C,V, and M suffix)	$S3I_{DD}$	5	0.96	–	µA	♦	—
	P	25°C (All parts)			1.3	–	µA	♦	—
	P[5]	85°C (C suffix only)			16.9	35	µA	♦	—
	P[5]	105°C (V suffix only)			37	90	µA	♦	—
	P[5]	125°C (M suffix only)			84	150	µA	♦	—
	C	–40°C (C,V, and M suffix)		3	0.85	–	µA	♦	—
	P	25°C (All parts)			1.2	–	µA	♦	—
	P[5]	85°C (C suffix only)			14.8	30	µA	♦	—
	P[5]	105°C (V suffix only)			32.7	80	µA	♦	—
	P[5]	125°C (M suffix only)			75	130	µA	♦	—



NOTE:
1. Not defined but normally LSB of character just received

Figure A-17. SPI Slave Timing (CPHA = 1)