SC9RS08MZ8

Data Sheet: Technical Data

RS08 Microcontrollers

Find the most current versions of all documents at:
http://www.freescale.com

SC9RS08MZ8
Rev. 1
9/2008

SC9RS08MZ8 Features

8-Bit RS08 Central Processor Unit (CPU)

- Up to 20 MHz CPU at 2.7 V to 5.5 V across temperature range of –40°C to 85°C
- Subset of HC08 instruction set with added BGND instruction

On-Chip Memory

- Flash read/program/erase over full operating voltage and temperature
- Random-access memory (RAM)
- Security circuitry to prevent unauthorized access to RAM and flash contents

Power-Saving Modes

- Stop mode
- Wakeup from power-saving modes using real-time interrupt (RTI) or ACMP

Clock Source Options

- Oscillator (XOSC) — Loop-control Pierce oscillator; crystal or ceramic resonator range of 31.25 kHz to 39.0625 kHz or 1 MHz to 5 MHz
- Internal Clock Source (ICS) — Internal clock source module containing a frequency-locked-loop (FLL) controlled by internal or external reference; precision trimming of internal reference allows 0.2% resolution and 2% deviation over temperature and voltage; supports bus frequencies up to 10 MHz

System Protection

- Watchdog computer operating properly (COP) reset with option to run from dedicated 1 kHz internal clock source or bus clock
- Low-voltage detection with reset or interrupt
- Illegal opcode detection with reset
- Illegal address detection with reset
- Flash-block protection

Development Support

- Single-wire background debug interface
- Breakpoint capability to allow single breakpoint setting during in-circuit debugging

Peripherals

- ADC — 12-channel, 8-bit resolution; 2.5 µs conversion time; automatic compare function; operation in stop; fully functional from 2.7 V to 5.5 V
- TPM — One 2-channel; selectable input capture, output compare, or buffered edge- or center-aligned PWM on each channel
- MTIM1 and MTIM2 — Two 8-bit modulo timers
- ACMP — Analog comparator; full rail-to-rail supply operation; option to compare to fixed internal bandgap reference voltage; can operate in stop mode

Input/Output

- 18 GPIOs including one output only pin and one input only pin
- Hysteresis and configurable pullup device on all input pins; configurable slew rate and drive strength on all output pins

Package Options

- 20-pin SOIC

SC9RS08MZ8 Data Sheet

Covers: SC9RS08MZ8

SC9RS08MZ8
Rev. 1
9/2008

Revision History

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://freescale.com

The following revision history table summarizes changes contained in this document.

Revision Number	Revision Date	Description of Changes
1	9/11/2008	Initial publish released.

This product incorporates SuperFlash® technology licensed from SST.

Freescale and the Freescale logo are trademarks of Freescale Semiconductor, Inc.
© Freescale Semiconductor, Inc., 2008. All rights reserved.

List of Chapters

Contents

Section Number **Title** **Page**

Chapter 1
Device Overview

Chapter 2
Pins and Connections

Chapter 3
Modes of Operation

Chapter 4
Memory

Chapter 5
Resets, Interrupts, and General System Control

Chapter 6
Parallel Input/Output Control

Chapter 7
Central Processor Unit (RS08CPUV1)

Chapter 8
Analog Comparator (RS08ACMPV1)

Chapter 9
8-Bit Analog-to-Digital Converter (RS08ADC10V1)

Chapter 10
Internal Clock Source (RS08ICSOSCV1)

Chapter 11
Modulo Timer (RS08MTIMV1)

Chapter 12
16-Bit Timer/PWM (RS08TPMV2)

Chapter 13
Development Support

Appendix A
Electrical Characteristics

Appendix B
Ordering Information

Chapter 1
Device Overview

1.1 Overview

The SC9RS08MZ8 microcontroller unit (MCU) is an extremely low-cost, small pin count, high performance device intended for home appliances, medical equipment. As a general purpose microcontroller, this device is composed of standard on-chip modules including a very small and highly efficient RS08 CPU core, 254 bytes RAM, 8K bytes flash, two 8-bit modulo timers,12-channel 8-bit ADC, 2-channel 16-bit Timer/PWM, and analog comparator. The device is available in 20-pin SOIC package.

1.2 MCU Block Diagram

The block diagram, Figure 1-1, shows the SC9RS08MZ8 MCU structure.

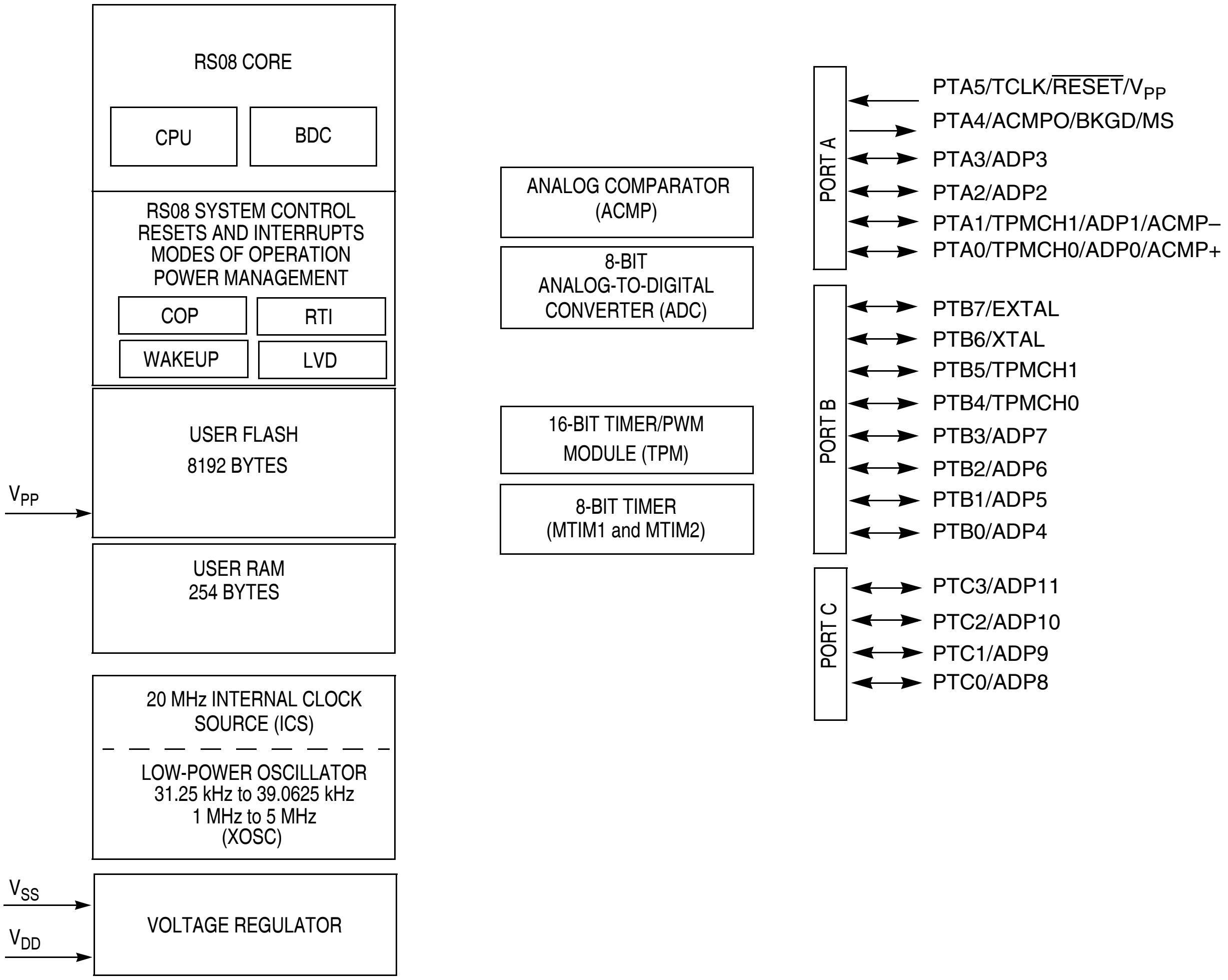


Figure 1-1. SC9RS08MZ8 Block Diagram

Table 1-1 provides the functional versions of the on-chip modules.

Table 1-1. Block Versions

Module	Version
RS08 CPU	1
Analog Comparator (RS08 ACMP)	1
Modulo Timer (RS08 MTIM)	1
Internal Clock Source (RS08 ICSOSC)	1
Analog-to-Digital Converter (RS08 ADC10)	1
16-Bit Timer/PWM (RS08 TPM)	2
XOSC	1

1.3 System Clock Distribution

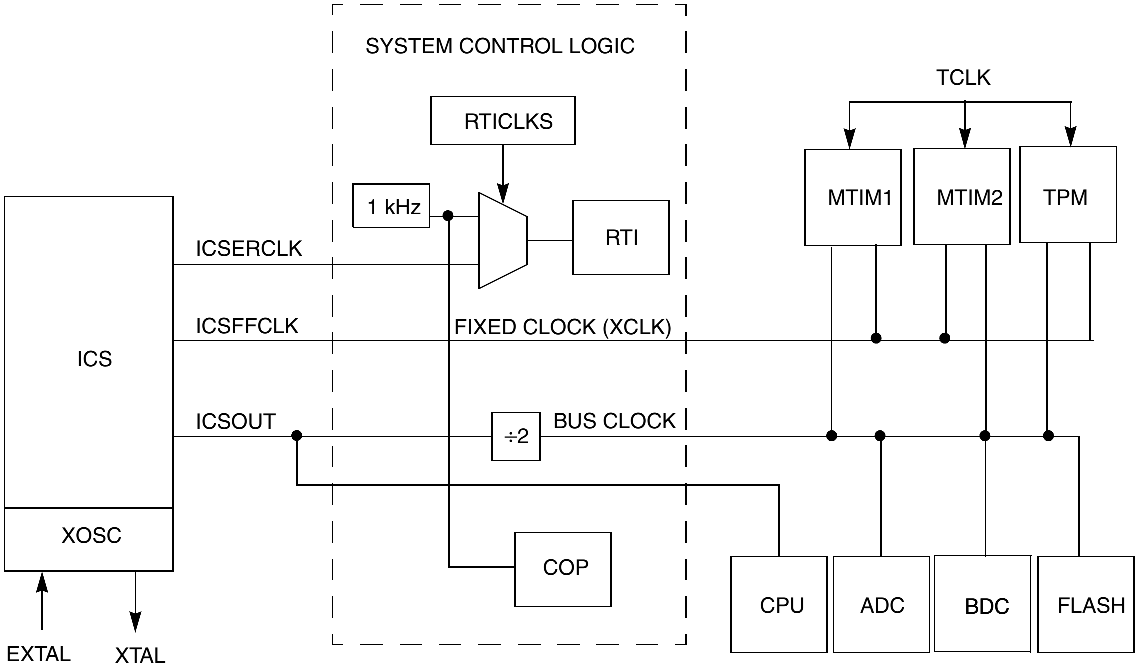


Figure 1-2. System Clock Distribution Diagram

Figure 1-2 shows a simplified clock connection diagram for the MCU. The bus clock frequency is half the ICS output frequency and used by all internal modules.

Chapter 2
Pins and Connections

2.1 Introduction

This chapter describes signals that connect to package pins. It includes a pinout diagram, a signal properties table, and a detailed signal discussion.

2.2 Device Pin Assignment

Figure 2-1 show the pin assignments in the packages for the SC9RS08MZ8.

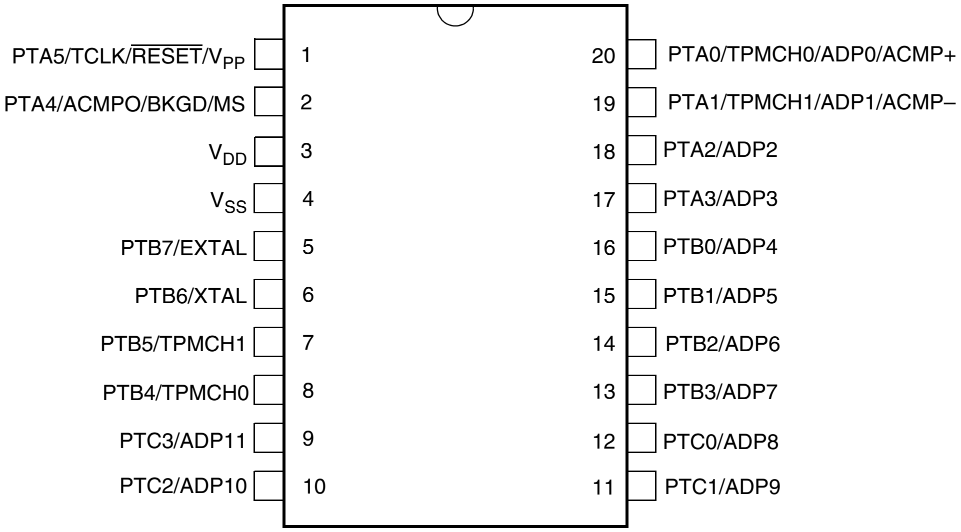
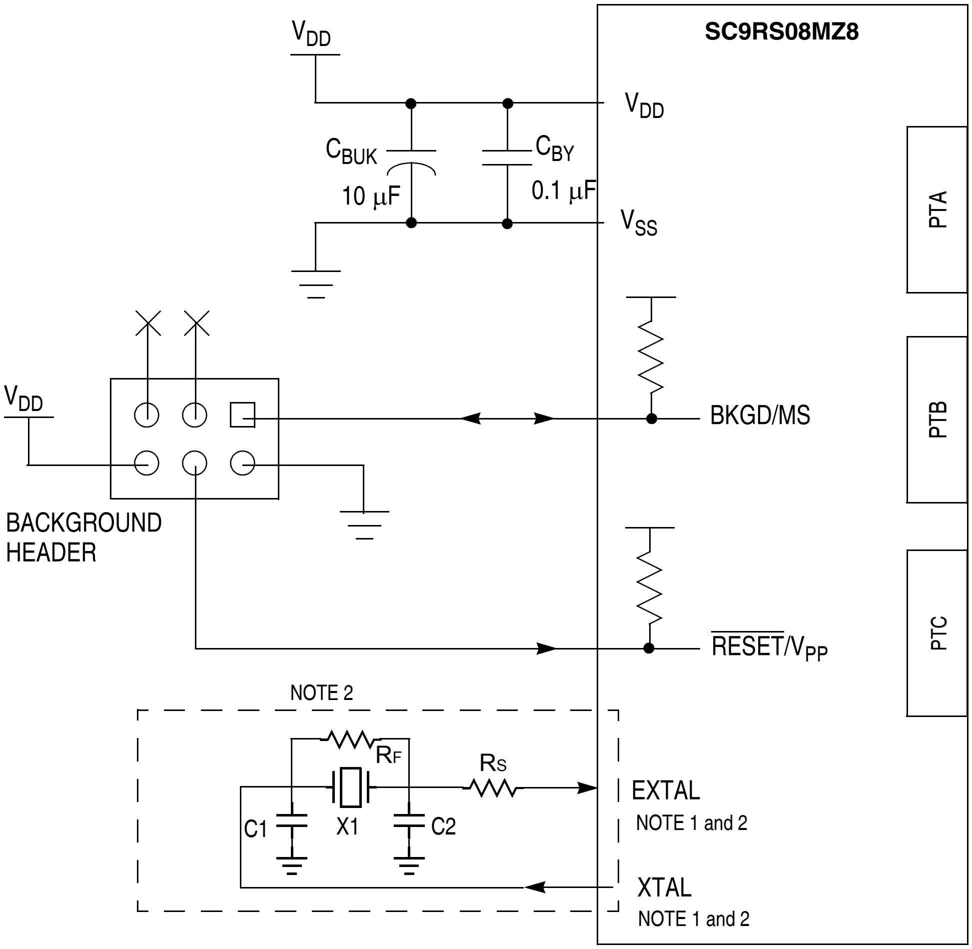


Figure 2-1. SC9RS08MZ8 20-Pin SOIC Package

2.3 Recommended System Connections

Figure 2-2 shows the reference connection for background debug and flash programming.



NOTES:

[1] Not required if using the internal clock option.

[2] These are the same pins as PTB6 and PTB7.

Figure 2-2. Reference System Connection Diagram

2.4 Pin Detail

This section provides a detailed description of system connections.

2.4.1 Power Pins

V_{DD} and V_{SS} are the primary power supply pins for the MCU. This voltage source supplies power to all I/O buffer circuitry and to an internal voltage regulator. The internal voltage regulator provides a regulated lower-voltage source to the CPU and other MCU internal circuitry.

Typically, application systems have two separate capacitors across the power pins: a bulk electrolytic capacitor such as a 10 µF tantalum capacitor, to provide bulk charge storage for the overall system, and a bypass capacitor such as a 0.1µF ceramic capacitor, located as near to the MCU power pins as practical to suppress high-frequency noise.

2.4.2 PTA5/TCLK/$\overline{\text{RESET}}$/V_{PP} Pin

After a power-on reset (POR) into user mode, the PTA5/TCLK/$\overline{\text{RESET}}$/V_{pp} pin defaults to a general-purpose input port pin, PTA5. Setting RSTPE in SOPT configures the pin to be the $\overline{\text{RESET}}$ input pin. After configured as $\overline{\text{RESET}}$, the pin remains as $\overline{\text{RESET}}$ until the next reset. The $\overline{\text{RESET}}$ pin can be used to reset the MCU from an external source when the pin is driven low. When enabled as the $\overline{\text{RESET}}$ pin (RSTPE = 1), the internal pullup device is automatically enabled.

External V_{PP} voltage (typically 12 V, see Appendix A, "Electrical Characteristics") is required on this pin when performing flash programming or erasing. The V_{PP} connection is always connected to the internal flash module regardless of the pin function. To avoid over stressing the flash, external V_{PP} voltage must be removed and voltage higher than V_{DD} must be avoided when flash programming or erasing does not occur.

NOTE

This pin does not contain a clamp diode to V_{DD} and must not be driven above V_{DD} when flash programming or erasing does not occur.

2.4.3 PTA4/ACMPO/BKGD/MS Pin

The background/mode select function is shared with an output-only pin on PTA4 pin and the optional analog comparator output. While in reset, the pin functions as a mode select pin. Immediately after reset rises, the pin functions as the background pin and can be used for background debug communication. While functioning as a background / mode select pin, this pin has an internal pullup device enabled. To use as an output-only port, clear BKGDPE in SOPT.

If nothing is connected to this pin, the MCU enters normal operating mode at the rising edge of reset. If a debug system is connected to the 6-pin standard background debug header, it can hold BKGD/MS low during the power-on-reset, which forces 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 equals the bus clock rate; therefore, significant capacitance must not be 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 the internal pullup device do not affect rise and fall times on the BKGD pin.

2.4.4 General-Purpose I/O and Peripheral Ports

The remaining pins are shared among general-purpose I/O and on-chip peripheral functions such as timers and the analog comparator. Immediately after reset, all of these pins are configured as high-impedance general-purpose inputs with internal pullup/pulldown devices disabled.

NOTE

To avoid extra current drain from floating input pins, the reset initialization routine in the application program must enable on-chip pullup/pulldown devices or change the direction of unused pins to outputs.

Table 2-1. Pin Availability by Package Pin-Count

Pin Number		<— Lowest **Priority** —> Highest				
20	**16**	**Port Pin**	**Alt 1**	**Alt 2**	**Alt 3**	**Alt 4**
1	1	PTA5		TCLK	$\overline{\text{RESET}}$	V_{PP}
2	2	PTA4	ACMPO	BKGD	MS	
3	3					V_{DD}
4	4					V_{ss}
5	5	PTB7				EXTAL
6	6	PTB6				XTAL
7	7	PTB5	TPMCH1[1]			
8	8	PTB4	TPMCH0[1]			
9	—	PTC3			ADP11	
10	—	PTC2			ADP10	
11	—	PTC1			ADP9	
12	—	PTC0			ADP8	
13	9	PTB3			ADP7	
14	10	PTB2			ADP6	
15	11	PTB1			ADP5	
16	12	PTB0			ADP4	
17	13	PTA3			ADP3	
18	14	PTA2			ADP2	
19	15	PTA1		TPMCH1[2]	ADP1	ACMP–
20	16	PTA0		TPMCH0[2]	ADP0	ACMP+

[1] TPM pins can be remapped to PTA0 and PTA1

Chapter 3
Modes of Operation

3.1 Introduction

This chapter describes the SC9RS08MZ8 operating modes. It also details entry into each mode, exit from each mode, and functionality while in each of the modes.

3.2 Features

- Active background mode for code development
- Stop mode:
  - System clocks are stopped
  - All internal circuits remain powered for fast recovery

3.3 Run Mode

Run mode is the normal operating mode for the SC9RS08MZ8. This mode is selected when the BKGD/MS pin is high at the rising edge of reset. In this mode, the CPU executes code from internal memory beginning at the address $3FFD. A JMP instruction (opcode $BC) with operand located at $3FFE–$3FFF must be programmed into the user application for correct reset operation. The operand defines the location where the user program starts. Instead of using the vector fetching process as in HC08/S08 families, the user program is responsible for performing a JMP instruction to relocate the program counter to the correct user program start location.

3.4 Active Background Mode

The active background mode functions are managed through the background debug controller (BDC) in the RS08 core. The BDC provides the means for analyzing MCU operation during software development.

Active background mode is entered in any of four ways:

- When the BKGD/MS pin is low during power-on-reset (POR) or immediately after issuing a background debug force reset (BDC_RESET) command
- When a BACKGROUND command is received through the BKGD pin
- When a BGND instruction is executed
- When a BDC breakpoint is encountered

After active background mode is entered, the CPU stays in a suspended state waiting for serial background commands rather than executing instructions from the user application program.

Background commands are of two types:

- Non-intrusive commands — Commands that can be issued while the user program is running, can be issued through the BKGD pin while the MCU is in run mode. Non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
  - — Memory access commands
  - — Memory-access-with-status commands
  - — BACKGROUND command
- Active background commands — Can be executed only while the MCU is in active background mode, include commands to:
  - — Read or write CPU registers
  - — Trace one user program instruction at a time
  - — Leave active background mode to return to the user application program (GO)

Active background mode is used to program user application code into the flash program memory before the MCU is operated in run mode for the first time. When the SC9RS08MZ8 is shipped, the flash program memory is usually erased so no program can be executed in run mode until the flash memory is initially programmed. The active background mode can also be used to erase and reprogram the flash memory after it is programmed.

For additional information about active background mode, refer to the Chapter 13, “Development Support.”

3.5 Stop Mode

Stop mode is entered upon execution of a STOP instruction when the STOPE bit in the system option register is set. In stop mode, all internal clocks to the CPU and modules are halted. If the STOPE bit is not set when the CPU executes a STOP instruction, the MCU does not enter stop mode, and an illegal opcode reset is forced.

Table 3-1 summarizes the behavior of the MCU in stop mode.

Table 3-1. Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS[1]	ACMP[2]	Regulator[3]	I/O Pins	RTI	ADC[4]
Stop	Standby	Standby	Optionally on	Optionally on	Optionally on	States held	Optionally on	Optionally on

[1] ICS requires IREFSTEN = 1 and LVDE and LVDSE must be set to allow operation in stop.

[2] If bandgap reference is required, the LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

[3] When BDM is enabled, the Regulator is on. Or else, only when LVDE and LVDSE bits both in the SPMSC1 to be set, Regulator is in on mode.

[4] Requires the asynchronous ADC clock, LVDE and LVDSE bits both in the SPMSC1 to be set, otherwise ADC is in standby mode.

Upon entering stop mode, all clocks in the MCU are halted. The ICS is turned off by default when the IREFSTEN bit is cleared and the voltage regulator enters standby. The states of all of the internal registers and logic, as well as the RAM content, are maintained. The I/O pin states are held.

Exit from stop is done by asserting reset, any enabled asynchronous interrupt, or the real-time interrupt. The asynchronous interrupts are LVD interrupt, ADC interrupt or the ACMP interrupt.

If stop is exited by asserting the $\overline{\text{RESET}}$ pin, the MCU is reset and program execution starts at location $3FFD. If exited by an asynchronous interrupt or real-time interrupt, the next instruction after the location where the STOP instruction was executed is executed accordingly. The user program must probe for the corresponding interrupt source that woke the CPU.

A separate self-clocked source (≈1 kHz) for the real-time interrupt allows a wakeup from stop mode with no external components. When RTIS = 000, the real-time interrupt function is disabled. When MCU is in STOP mode, LVD is disabled, and RTICLKS = 1, the internal 1 kHz oscillator is disabled and power consumption is lower.

The external clock source can also be enabled for the real-time interrupt to allow a wakeup from stop mode with no external components. Setting the ERCLKEN=1 and EREFSTEN=1 enables the external clock source when in STOP mode.

To enable ADC in STOP mode, the asynchronous ADC clock and LVD must be enabled by setting LVDE

and LVDSE, otherwise the ADC is in standby.

To enable XOSC to operate with an external reference clock source in STOP mode, LVD must be enabled by setting LVDE and LVDSE.

3.5.1 Active BDM Enabled in Stop Mode

Entry into active background mode from run mode is enabled if the ENBDM bit in BDCSCR is set. This register is described in the Chapter 13, “Development Support.” If ENBDM is set when the CPU executes a STOP instruction, the system clocks to the background debug logic remain active when the MCU enters stop mode so background debug communication is still possible. The voltage regulator does not enter its low-power standby state. It maintains full internal regulation.

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access. They report an error indicating that the MCU is in stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the ENBDM bit is set. After active background mode is entered, all background commands are available.

Table 3-2 summarizes the MCU behavior in stop when entry into the active background mode is enabled.

Table 3-2. BDM Enabled Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS	ACMP	Regulator	I/O Pins	RTI	ADC
Stop	Standby	Standby	On	Optionally on	on	States held	Optionally on	Optionally on

3.5.2 LVD Enabled in Stop Mode

The LVD system can generate an interrupt or a reset when the supply voltage drops below the LVD voltage. The voltage regulator remains active if the LVD is enabled in stop (LVDE and LVDSE bits in SPMSC1 both set) when the CPU executes a STOP instruction.

Table 3-3 summarizes the behavior of the MCU in stop when LVD is enabled.

Table 3-3. LVD Enabled Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS	ACMP	Regulator	I/O Pins	RTI	ADC[1]
Stop	Standby	Standby	Optionally on	Optionally on	On	States held	Optionally on	Optionally on

[1] Requires the asynchronous ADC clock to be enabled.

Chapter 4
Memory

4.1 Memory Map

The memory map of the MCU is divided into the following groups:

- Fast access RAM using tiny and short instructions ($0000 – $000D)
- Indirect data access D[X] ($000E)
- Index register X for D[X] ($000F)
- Frequently used peripheral registers ($0010 – $001E, $0020 – $002F)
- PAGESEL register ($001F)
- RAM for SC9RS08MZ8 ($0030 – $00BF, $0100 – $015F)
- Paging window ($00C0 – $00FF)
- Other peripheral registers ($0200 – $023F)
- Nonvolatile memory for SC9RS08MZ8 ($2000 – $3FFF)

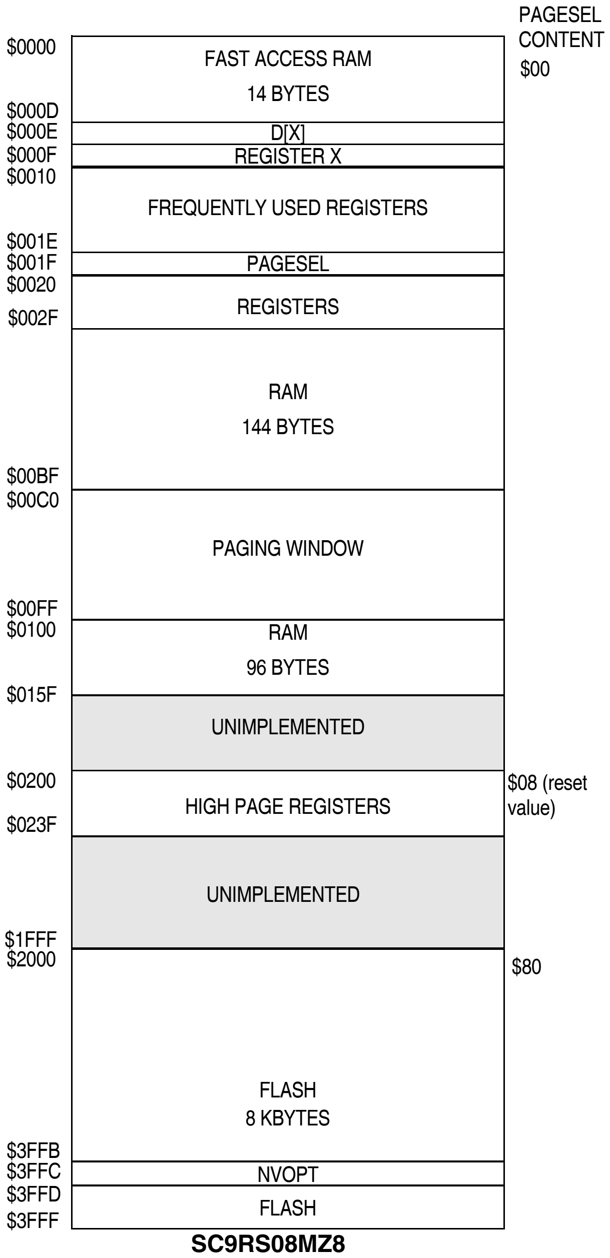


Figure 4-1. SC9RS08MZ8 Memory Maps

4.2 Unimplemented Memory

Attempting to access data or an instruction at an unimplemented memory address causes reset.

4.3 Indexed/Indirect Addressing

Register D[X] and register X combined perform indirect data access. Register D[X] is mapped to address $000E. Register X is located in address $000F. The 8-bit register X contains the address used when register D[X] is accessed. Register X is cleared to zero upon reset. By programming register X, any location on the first page ($0000–$00FF) can be read/written via register D[X]. Figure 4-2 shows the relationship between D[X] and register X. For example, in HC08/S08 syntax *latex* is comparable to *lda D[X]* in RS08 coding when register X has been programmed with the index value.

The physical location of $000E is in RAM. Accessing the location through D[X] returns $000E RAM content when register X contains $0E. The physical location of $000F is register X, itself. Reading the location through D[X] returns register X content. Writing to the location modifies register X.

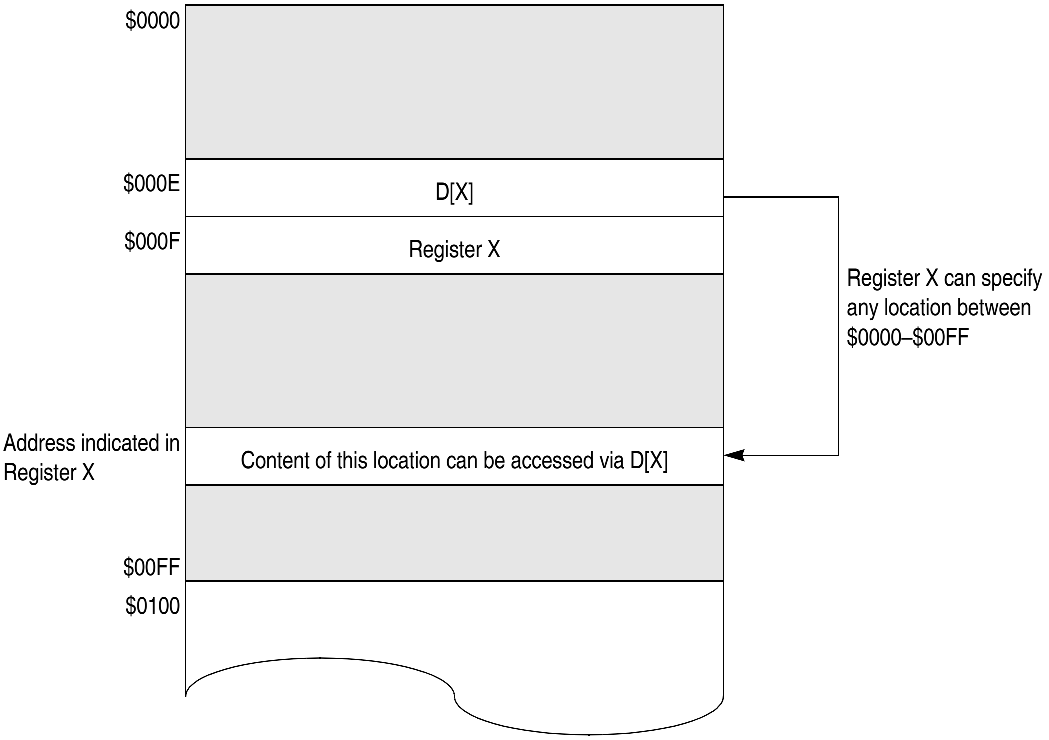


Figure 4-2. Indirect Addressing Registers

4.4 RAM and Register Addresses and Bit Assignments

Use short and direct addressing mode instructions to read and write to the fast access RAM area. For tiny addressing mode instructions, the operand is encoded with the opcode to a single byte.

Frequently used registers can make use of the short addressing mode instructions for faster load, store, and clear operations. For short addressing mode instructions, the operand is encoded along with the opcode to a single byte.

Table 4-1. Register Summary (Sheet 1 of 3)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$0000 – $000D	—	Fast Access RAM							
$000E	**D[X]**[1]	Bit 7	6	5	4	3	2	1	Bit 0
$000F	**X**	Bit 7	6	5	4	3	2	1	Bit 0
$0010	**ADCSC1**	COCO	AIEN	ADCO	ADCH				
$0011	**ADCSC2**	ADACT	ADTRG	ACFE	ACFGT	0	0	R	R
$0012	**ADCRH**	0	0	0	0	0	0	ADR9	ADR8
$0013	**ADCRL**	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
$0014	Reserved	—	—	—	—	—	—	—	—
$0015	**TPMC0SC**	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
$0016	**TPMC0VH**	Bit15	14	13	12	11	10	9	Bit8
$0017	**TPMC0VL**	Bit7	6	5	4	3	2	1	Bit0
$0018	**TPMC1SC**	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
$0019	**TPMC1VH**	Bit15	14	13	12	11	10	9	Bit8
$001A	**TPMC1VL**	Bit7	6	5	4	3	2	1	Bit0
$001B	**ACMPSC**	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD	
$001C	**PTAD**	0	0	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
$001D	**PTBD**	PTBD7	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2	PTBD1	PTBD0
$001E	**PTCD**	0	0	0	0	PTCD3	PTCD2	PTCD1	PTCD0
$001F	**PAGESEL**	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6
$0020	**MTIM1SC**	TOF	TOIE	TRST	TSTP	0	0	0	0
$0021	**MTIM1CLK**	0	0	CLKS		PS			
$0022	**MTIM1CNT**	COUNT							
$0023	**MTIM1MOD**	MOD							
$0024	**MTIM2SC**	TOF	TOIE	TRST	TSTP	0	0	0	0
$0025	**MTIM2CLK**	0	0	CLKS		PS			
$0026	**MTIM2CNT**	COUNT							
$0027	**MTIM2MOD**	MOD							
$0028 – $002F	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$0030 – $00BF		RAM							
$00C0 – $00FF		Paging Window							
$0100 – $015F		RAM							
$0160 – $01FF	Unimplemented	— —	— —	— —	— —	— —	— —	— —	— —
$0200	**SRS**	POR	PIN	COP	ILOP	ILAD	0	LVD	0
$0201	**SOPT**	COPE	COPT	STOPE	—	TPMCH1PS	TPMCH0PS	BKGDPE	RSTPE

Table 4-1. Register Summary (Sheet 2 of 3) (continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$0202	**SIP1**	—	—	ACMP	ADC	TPM	MTIM2	MTIM1	RTI
$0203	Unimplemented	—	—	—	—	—	—	—	—
$0204	Reserved	—	—	—	—	—	—	—	—
$0205	Unimplemented	—	—	—	—	—	—	—	—
$0206	**SDIDH**	REV3	REV2	REV1	REV0	ID[11:8]			
$0207	**SDIDL**	ID[7:0]							
$0208	**SRTISC**	RTIF	RTIACK	RTICLKS	RTIE	0	RTIS		
$0209	**SPMSC1**	LVDF	LVDACK	LVDIE	LVDRE	LVDSE	LVDE	0	BGBE
$020A – $020B	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$020C – $020F	Unimplemented	— —	— —	— —	— —	— —	— —	— —	— —
$0210	**FOPT**	0	0	0	0	0	0	0	SECD
$0211	**FLCR**	0	0	0	0	HVEN	MASS	0	PGM
$0212 – $0213	Reserved	— —	— —	— —	— —	— —	— —	— —	— —
$0214	**ADCCVH**	0	0	0	0	—	—	ADCV9	ADCV8
$0215	**ADCCVL**	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
$0216	**ADCCFG**	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
$0217	**APCTL1**	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
$0218	**APCTL2**	0	0	0	0	ADPC11	ADPC10	ADPC9	ADPC8
$0219 – $021F	Unimplemented	— —	— —	— —	— —	— —	— —	— —	— —
$0220	**PTADD**	0	0	0	0	PTADD3	PTADD2	PTADD1	PTADD0
$0221	**PTAPE**	0	0	PTAPE5	0	PTAPE3	PTAPE2	PTAPE1	PTAPE0
$0222	**PTAPUD**	0	0	PTAPUD5	0	PTAPUD3	PTAPUD2	PTAPUD1	PTAPUD0
$0223	**PTASE**	0	0	0	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
$0224	**PTBDD**	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
$0225	**PTBPE**	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
$0226	**PTBPUD**	PTBPUD7	PTBPUD6	PTBPUD5	PTBPUD4	PTBPUD3	PTBPUD2	PTBPUD1	PTBPUD0
$0227	**PTBSE**	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
$0228	**PTCDD**	0	0	0	0	PTCDD3	PTCDD2	PTCDD1	PTCDD0
$0229	**PTCPE**	0	0	0	0	PTCPE3	PTCPE2	PTCPE1	PTCPE0
$022A	**PTCPUD**	0	0	0	0	PTCPUD3	PTCPUD2	PTCPUD1	PTCPUD0
$022B	**PTCSE**	0	0	0	0	PTCSE3	PTCSE2	PTCSE1	PTCSE0
$022C – $022F	Unimplemented	— —	— —	— —	— —	— —	— —	— —	— —
$0230	**TPMSC**	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
$0231	**TPMCNTH**	Bit15	14	13	12	11	10	9	Bit8
$0232	**TPMCNTL**	Bit7	6	5	4	3	2	1	Bit0
$0233	**TPMMODH**	Bit15	14	13	12	11	10	9	Bit8
$0234	**TPMMODL**	Bit7	6	5	4	3	2	1	Bit0
$0235	Reserved	—	—	—	—	—	—	—	—

Table 4-1. Register Summary (Sheet 3 of 3) (continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$0236	Reserved	—	—	—	—	—	—	—	—
$0237	**PTADS**	0	0	0	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
$0238	**PTBDS**	PTBDS7	PTBDS6	PTBDS5	PTBDS4	PTBDS3	PTBDS2	PTBDS1	PTBDS0
$0239	**PTCDS**	0	0	0	0	PTCDS3	PTCDS2	PTCDS1	PTCDS0
$023A – $023B	Unimplemented	— —	— —	— —	— —	— —	— —	— —	— —
$023C	**ICSC1**	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
$023D	**ICSC2**	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
$023E	**ICSTRM**	TRIM							
$023F	**ICSSC**	0	0	0	0	CLKST		OSCINIT	FTRIM
$3FF8	Reserved	—	—	—	—	—	—	—	—
$3FF9	Reserved	—	—	—	—	—	—	—	—
$3FFA[2]	Reserved	Reserved for Room Temperature ICS Trim							
$3FFB[2]	Reserved	Reserved							FTRIM
$3FFC	**NVOPT**	0	0	0	0	0	0	0	SECD

▨ = Unimplemented or Reserved

[1] Physical RAM in $000E can be accessed through D[X] register when the content of the index register X is $0E.

[2] If using the MCU untrimmed, $3FFA and $3FFB may be used by applications.

The ICS factory-trimmed value will be stored in 0x3FFA and 0x3FFB (bit 0). The factory-trimmed bus frequency is 10 MHz.

4.5 RAM

The device includes three static RAM sections. The locations from $0000 to $000D can be directly accessed using the more efficient tiny addressing mode instructions and short addressing mode instructions. Location $000E RAM can be accessed through D[X] register when register X is $0E or through the paging window location $00CE when PAGESEL register is $00. The second section of RAM starts from $0030 to $00BF and can be accessed using direct addressing mode instructions. The third section of RAM starts from $0100 to $015F.

The RAM retains data when the MCU is in low-power wait and stop mode. RAM data is unaffected by any reset if the supply voltage does not drop below the minimum value for RAM retention.

4.6 Flash

The flash memory is for program storage. In-circuit programming allows the operating program to be loaded into the flash memory after final assembly of the application product. You can program the entire array through the single-wire background debug interface. Because the device does not include on-chip charge pump circuitry, external V_{PP} is required for program and erase operations.

4.6.1 Features

Flash memory features include:

- Up to 1000 program/erase cycles at typical voltage and temperature
- Security feature for flash

4.6.2 Flash Programming Procedure

Flash memory is programmed on a row basis. A row consists of 64 consecutive bytes starting from addresses $2X00, $2X40, $2X80, or $2XC0. To program a row of flash memory:

1. Apply external V_{PP}.
2. Set the PGM bit. This configures the memory for program operation and enables the latching of address and data for programming.
3. Write any data to any flash location via the high-page-accessing window $00C0–$00FF, within the address range of the row to be programmed. (Prior to the data writing operation, the PAGESEL register must be configured correctly to map the high-page-accessing window to the corresponding flash row.)
4. Wait for a time, t_{nvs}.
5. Set the HVEN bit.
6. Wait for a time, t_{pgs}.
7. Write data to the flash location to be programmed.
8. Wait for a time, t_{prog}.
9. Repeat steps seven and eight until all bytes within the row are programmed.
10. Clear the PGM bit.
11. Wait for a time, t_{nvh}.
12. Clear the HVEN bit.
13. After time, t_{rcv}, the memory can be accessed in read mode again.
14. Remove external V_{PP}.

This program sequence is repeated throughout the memory until all data is programmed.

NOTE

Software code executed from flash locations cannot program or erase flash memory. To program or erase flash, commands must be executed from RAM or BDC commands. User code must not enter wait or stop during an erase or program sequence.

These operations must be performed in the order shown, or other unrelated operations may occur between the steps.

4.6.3 Flash Mass Erase Operation

To mass erase the entire flash memory:

1. Apply external V_{PP}.
2. Set the MASS bit in the flash control register.
3. Write any data to any flash location via the high page accessing window \$00C0–\$00FF. (Prior to the data writing operation, the PAGESEL register must be configured correctly to map the high-page-accessing window to any flash locations.)
4. Wait for a time, t_{nvs}.
5. Set the HVEN bit.
6. Wait for a time t_{me}.
7. Clear the MASS bit.
8. Wait for a time, t_{nvh1}.
9. Clear the HVEN bit.
10. After t_{rcv} time, the memory can be accessed in read mode again.
11. Remove external V_{PP}.

NOTE

Software code executed from flash locations cannot program or erase flash memory. To program or erase flash, commands must be executed from RAM or BDC commands. User code must not enter wait or stop during an erase or program sequence.

These operations must be performed in the order shown or other unrelated operations may occur between the steps.

4.6.4 Security

The SC9RS08MZ8 includes circuitry to help prevent unauthorized access to flash memory contents. When security is engaged, flash is a secure resource. The RAM, direct-page registers, and background debug controller are unsecured resources. Attempts to access a secure memory location are blocked (reads return all 0s) if they are through the background debug interface, or when BKGDPE is set.

Security is engaged or disengaged based on the state of a nonvolatile register bit (SECD) in the FOPT register. During reset, the nonvolatile location NVOPT contents are copied from flash into the working FOPT register in high-page register space. Engage security by programming the NVOPT location. You can do this while the flash memory is programmed. The erased state (SECD = 1) makes the MCU unsecure. When SECD in NVOPT is programmed (SECD = 0), the next time the device is reset via POR, internal reset, or external reset, security is engaged. To disengage security, mass erase must be performed via BDM commands and followed by any reset.

The separate background debug controller can still be used for registers and RAM access. Via BDM commands, flash mass erase is possible by writing to the flash control register that follows the flash mass erase procedure listed in Section 4.6.3, “Flash Mass Erase Operation.”

Security can always be disengaged through the background debug interface when you:

1. Mass erase flash via background BDM commands or RAM loaded program.
2. Perform reset. The device boots up with security disengaged.

NOTE

> When the device boots up to normal operating mode, where MS pin is high during reset, with SECD programmed (SECD = 0), flash security is engaged. BKGDPE is reset to 0, all BDM communication is blocked, and background debug is not allowed.

4.7 Flash Registers and Control Bits

The flash module has a nonvolatile register NVOPT ($3FFC) in flash memory which is copied into the corresponding control register FOPT ($0210) at reset.

4.7.1 Flash Options Register (FOPT and NVOPT)

During reset, the nonvolatile location NVOPT contents are copied from flash into FOPT. Bits 7 through 1 are not used and always read 0. This register may be read at any time, but writes have no meaning or effect. To change the value in this register, erase and reprogram the NVOPT location in flash memory then issue a new MCU reset.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	SECD
W	░	░	░	░	░	░	░	░
Reset	This register is loaded from nonvolatile location NVOPT during reset.							

░ = Unimplemented or Reserved

Figure 4-3. Flash Options Register (FOPT)

Table 4-2. FOPT Field Descriptions

Field	Description
0 SECD	**Security State Code** — This bit field determines the MCU security state. When the MCU is secured, the flash memory contents cannot be accessed by instructions from any unsecured source including the background debug interface; see Section 4.6.4, “Security.” 0 Security engaged. 1 Security disengaged.

4.7.2 Flash Control Register (FLCR)

	7	6	5	4	3	2	1	0
R	0	0	0	0	HVEN	MASS	0	PGM[1]
W	▨	▨	▨	▨			▨	
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 4-4. Flash Control Register (FLCR)

Table 4-3. FLCR Field Descriptions

Field	Description
3 HVEN	**High Voltage Enable** — This read/write bit enables high voltages to the flash array for program and erase operations. HVEN can be set only if PGM = 1 or MASS = 1 and the proper sequence for program or erase is followed. 0 High voltage disabled to array. 1 High voltage enabled to array.
2 MASS	**Mass Erase Control Bit** — This read/write bit configures the memory for mass erase operation. 0 Mass-erase operation not selected. 1 Mass-erase operation selected.
0 PGM[1]	**Program Control Bit** — This read/write bit configures the memory for program operation. PGM is interlocked with the MASS bit, so both bits cannot be equal to 1 or set to 1 at the same time. 0 Program operation not selected. 1 Program operation selected.

---

[1] When flash security is engaged, writing to PGM bit has no effect. As a result, flash programming is not allowed.

4.8 Page Select Register (PAGESEL)

There is a 64-byte window ($00C0 – $00FF) in the direct-page reserved for paging access. Programming the page-select register determines the corresponding 64-byte block on the memory map for direct-page access. For example, when the PAGESEL register is programmed with value $08, the high-page registers ($0200 – $023F) can be accessed through the paging window ($00C0 – $00FF) via direct addressing mode instructions.

	7	6	5	4	3	2	1	0
R	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6
W								
Reset	0	0	0	0	1	0	0	0

Figure 4-5. Page Select Register (PAGESEL)

Table 4-4. PAGESEL Field Descriptions

Field	Description
7:0 AD[13:6]	**Page Selector**— These bits define the address line bit 6 to bit 13, which determines the 64-byte block boundary of the memory block accessed via the direct page window. See Figure 4-6 and Table 4-5.

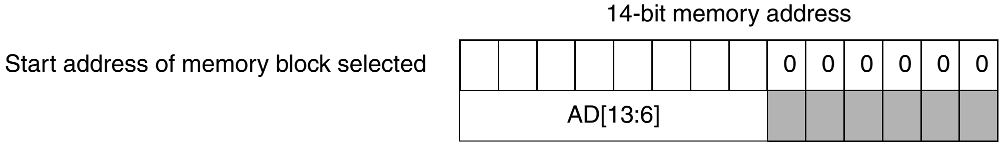


Figure 4-6. Memory Block Boundary Selector

Table 4-5 shows the memory block to be accessed through paging window ($00C0 – $00FF).

Table 4-5. Paging Window for $00C0–$00FF

Page	Memory Address
$00	$0000–$003F
$01	$0040–$007F
$02	$0080–$00BF
$03	$00C0–$00FF
$04	$0100–$013F
. . .	. . .
$FE	$3F80–$3FBF
$FF	$3FC0–$3FFF

NOTE

Physical location $0000-$000E is RAM. Physical location $000F is register X. The D[X] register is mapped to address $000E only. The physical RAM in $000E can be accessed through the D[X] register when the X register is $0E or $CE with PAGESEL as $00.

When PAGESEL register is $00, the paging window is mapped to the first page ($00 – $3F). Paged location $00C0 – $00CE is mapped to physical location $0000 – $000E, that is, RAM. Paged location $00CF is mapped to register X. Therefore, accessing address $CE returns the physical RAM content in $000E. Accessing address $000E returns D[X] register content.

Chapter 5
Resets, Interrupts, and General System Control

5.1 Introduction

This chapter discusses basic reset, interrupt mechanisms, and the various reset and interrupt sources in the SC9RS08MZ8. Some interrupt sources from peripheral modules are discussed in detail in other chapters of this data sheet. This chapter gathers basic information about all reset and interrupt sources in one place for easy reference. A few reset and wakeup sources, including the computer operating properly (COP) watchdog and real-time interrupt (RTI), are not part of on-chip peripheral systems with their own chapters but are part of the system control logic.

5.2 Features

Reset and interrupt features include:

- Multiple sources of reset for flexible system configuration and reliable operation
- System reset status register (SRS) to indicate the source of the most recent reset
- System interrupt pending register (SIP1) to indicate the status of pending system interrupts
  - Analog comparator interrupt with enable
  - Modulo timer interrupt with enable
  - Real-time interrupt with enable
  - ADC interrupt with enable
  - TPM interrupt with enable

5.3 MCU Reset

Resetting the MCU provides a way to start processing from a known set of conditions. During reset, most control and status registers are forced to initial values, and the program counter is started from location $3FFD. A JMP instruction (opcode $BC) with operand located at $3FFE – $3FFF must be programmed into the user application for correct reset operation. The operand defines the location at which the user program starts. On-chip peripheral modules are disabled and I/O pins are initially configured as general-purpose, high-impedance inputs with pullup/pulldown devices disabled.

The SC9RS08MZ8 has seven reset sources:

- External pin reset (PIN) — Enabled using RSTPE in SOPT
- Power-on reset (POR)
- Low-voltage detect (LVD)
- Computer operating properly (COP) timer

- Illegal opcode detect (ILOP)
- Illegal address detect (ILAD)
- Background debug forced reset via BDC command BDC_RESET

Each source except the background debug forced reset has an associated bit in the system reset status register (SRS).

5.4 Computer Operating Properly (COP) Watchdog

The COP watchdog is used to force a system reset if the application software fails to execute as expected. To prevent a system reset from the COP timer (when it is enabled), application software must periodically reset the COP counter. If the application program gets lost and fails to reset the COP counter before it times out, a system reset is generated to force the system back to a known starting point.

After any reset, the COPE becomes set in SOPT, which enables the COP watchdog (see Section 5.8.2, “System Options Register (SOPT).”). If the COP watchdog is not used in an application, it is disabled by clearing COPE. The COP counter is reset by writing any value to the address of SRS. This write does not affect the data in the read-only SRS. Instead, the act of writing to this address is decoded and sends a reset signal to the COP counter.

There is an associated short and long time-out controlled by COPT in SOPT. Table 5-1 summarizes the COPT bit control functions. The COP watchdog operates from the 1 kHz clock source and defaults to the associated long time-out (2^8 cycles).

Table 5-1. COP Configuration Options

COPT	COP Overflow Count[1]
0	2^5 cycles (32 ms)
1	2^8 cycles (256 ms)

[1] Values in this column are based on $t_{RTI} \approx 1$ ms. See t_{RTI} in the “SC9RS08MZ8 Data Sheet,” for the tolerance value.

Even if the application uses the reset default settings of COPE and COPT, write to the write-once SOPT registers during reset initialization to lock in the settings so they cannot be changed accidentally if the application program gets lost. The initial write to SOPT resets the COP counter.

In background debug mode, the COP counter does not increment.

When the MCU enters stop mode, the COP counter is re-initialized to zero upon entry to stop mode. The COP counter begins from zero as soon as the MCU exits stop mode.

5.5 Interrupts

The SC9RS08MZ8 does not include an interrupt controller with vector table lookup mechanism as used on the HC08 and HCS08 devices. However, the interrupt sources from modules such as LVD, ADC, RTI and ACMP are still available to wake the CPU from wait or stop mode. It is the user application’s responsibility to poll the corresponding module to determine the source of wakeup.

Each wakeup source of the module is associated with a corresponding interrupt enable bit. If the bit is disabled, the interrupt source is gated, and that particular source cannot wake the CPU from wait or stop mode. However, the corresponding interrupt flag is still set to indicate that an external wakeup event occurred.

The system interrupt pending register (SIP1) indicates the system-pending interrupt status. When the read-only bit of the SIP1 is enabled, it shows a pending interrupt to be serviced from the indicated module. Writing to the register bit has no effect. The pending interrupt flag is cleared automatically when all the corresponding interrupt flags from the indicated module are cleared.

5.6 Low-Voltage Detect (LVD) System

The SC9RS08MZ8 includes a system to protect memory contents and control MCU system states against low voltage conditions during supply voltage variations. The system is composed of a power-on reset (POR) circuit and an LVD circuit with a predefined trip voltage. The LVD circuit is enabled with LVDE in SPMSC1. The LVD is disabled upon entering stop mode unless LVDSE is set in SPMSC1. If LVDSE and LVDE are both set, the current consumption in stop with the LVD enabled is greater.

5.6.1 Power-On Reset Operation

When power is initially applied to the MCU, or when the supply voltage drops below the V_{POR} level, the POR circuit causes a reset condition. As the supply voltage rises, the LVD circuit holds the MCU in reset until the supply rises above the V_{LVD} level. The POR bit and LVD bit in SRS are set after a POR.

5.6.2 LVD Reset Operation

The LVD can be configured to generate a reset upon detecting a low voltage condition by setting LVDRE to 1. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above the level V_{LVD}. The LVD bit in the SRS register is set after an LVD reset or POR.

5.6.3 LVD Interrupt Operation

When a low voltage condition is detected and the LVD circuit is configured using SPMSC1 for interrupt operation (LVDE set, LVDIE set, and LVDRE clear), LVDF in SPMSC1 is set and an LVD interrupt request occurs.

5.7 Real-Time Interrupt (RTI)

The real-time interrupt function can be used to generate periodic interrupts. The RTI is driven from the 1 kHz internal clock oscillator or the external clock souce. The RTICLKS bit in SRTISC is used to select the RTI clock source. The 1 kHz internal or external clock sources for the RTI can be used when the MCU is in run, wait, or stop mode. When using the external oscillator in normal or wait mode, setting ERCLKEN=1. When using the external oscillator in stop mode, set ERCLKEN=1 and EREFSTEN=1.

The SRTISC register includes a read-only status flag, a write-only acknowledge bit, and a 3-bit control value (RTIS) used to select one of seven wakeup periods or disable RTI. The RTI has a local interrupt enable RTIE to allow masking of the real-time interrupt. The RTI can be disabled by writing each bit of

RTIS to 0s, and no interrupts are generated. See Section 5.8.4, “System Real-Time Interrupt Status and Control Register (SRTISC),” for more information.

5.8 Reset, Interrupt, and System Control Registers and Control Bits

Refer to the direct-page register summary in Chapter 4, “Memory,” for the absolute address assignments for all registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

Some control bits in the SOPT register are related to operation modes. Although brief descriptions of these bits are provided here, the related functions are further discussed in Chapter 3, “Modes of Operation.”

5.8.1 System Reset Status Register (SRS)

This high page register includes read-only status flags to indicate the source of the most recent reset. When a debug host forces reset by the BDC_RESET command, all of the status bits in SRS are cleared. Writing any value to this register address clears the COP watchdog timer without affecting this register’s contents. The reset state of these bits depends on what caused the MCU to reset.

	7	6	5	4	3	2	1	0
R	POR	PIN	COP	ILOP	ILAD	0	LVD	0
W	Writing any value to SRS address clears COP watchdog timer.							
POR:	1	0	0	0	0	0	1	0
LVR:	U	0	0	0	0	0	1	0
Any other reset:	0	Note 1	Note 1	Note 1	Note 1	0	0	0

[1] Any of these reset sources that are active at the time of reset entry causes the corresponding bit(s) to be set; bits corresponding to sources not active at the time of reset entry are cleared.

Figure 5-1. System Reset Status (SRS)

Table 5-2. SRS Field Descriptions

Field	Description
7 POR	**Power-On Reset** — Reset caused by power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVR) status bit is also set to indicate that the reset occurred while the internal supply was below the LVR threshold. 0 Reset not caused by POR. 1 POR caused reset.
6 PIN	**External Reset Pin** — Reset caused by an active-low level on the external reset pin. 0 Reset not caused by external reset pin. 1 External reset pin caused reset.
5 COP	**Computer Operating Properly (COP) Watchdog** — Reset caused by the COP watchdog timer timing out. COPE = 0 can block the reset source. 0 Reset not caused by COP timeout. 1 COP timeout caused reset.

Table 5-2. SRS Field Descriptions (continued)

Field	Description
4 ILOP	**Illegal Opcode** — Reset was caused by an attempt to execute an unimplemented or illegal opcode. The STOP instruction is considered illegal if STOPE = 0 in the SOPT register disables stop mode. The BGND instruction is considered illegal if ENBDM = 0 in the BDCSC register disables stop mode. 0 Reset not caused by an illegal opcode. 1 Illegal opcode caused reset.
3 ILAD	**Illegal Address** — Reset caused by an attempt to access either data or an instruction at an unimplemented memory address. 0 Reset not caused by an illegal address. 1 Illegal address caused reset.
1 LVD	**Low Voltage Detect** — If the LVDRE bit is set and the supply drops below the LVD trip voltage, an LVD reset occurs. POR sets this bit. 0 Reset not caused by LVD trip or POR. 1 Either LVD trip or POR caused reset.

5.8.2 System Options Register (SOPT)

This high page register is a write-once register so only the first write after reset is honored. It can be read at any time. Any subsequent attempt to write to SOPT (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT must be written during the user's reset initialization program to set the desired controls even if the desired settings and reset settings are the same.

	7	6	5	4	3	2	1	0
R W	COPE	COPT	STOPE	▨	TPMCH1PS	TPMCH0PS	BKGDPE	RSTPE
Reset:	1	1	0	0	0	0	1 (Note 1)	u
POR:	1	1	0	0	0	0	1 (Note1)	0

▨ = Unimplemented or Reserved

u = Unaffected

[1] When the device is power on reset, BKGEPE is reset to 1. When the device is reset into normal operating mode (MS is high during reset), BKGDPE is reset to 1 if flash security is disengaged (SECD = 1). BKGDPE is reset to 0 if flash security is engaged (SECD = 0). When the device is reset into active BDM mode (MS is low during reset), BKGDPE is always reset to 1 such that BDM communication is allowed.

Figure 5-2. System Options Register 1 (SOPT)

Table 5-3. SOPT Register Field Descriptions

Field	Description
7 COPE	**COP Watchdog Enable** — This write-once bit selects whether the COP watchdog is enabled. 0 COP watchdog timer disabled. 1 COP watchdog timer enabled (force reset on timeout).
6 COPT	**COP Watchdog Timeout** — This write-once bit selects the timeout period of the COP. 0 Short timeout period selected. 1 Long timeout period selected.

Table 5-3. SOPT Register Field Descriptions (continued)

Field	Description
5 STOPE	**Stop Mode Enable** — This write-once bit is used to enable stop mode. If stop mode is disabled and a user program attempts to execute a STOP instruction, an illegal opcode reset is forced. 0 Stop mode disabled. 1 Stop mode enabled.
3 TPMCH1PS	**TPMCH1 Pin Select—** This bit selects the location of the TPMCH1 pin of the TPM module. 0 TPMCH1 on PTA1. 1 TPMCH1 on PTB5.
2 TPMCH0PS	**TPMCH0 Pin Select—** This bit selects the location of the TPMCH0 pin of the TPM module. 0 TPMCH0 on PTA0. 1 TPMCH0 on PTB4.
1 BKGDPE[1,2]	**Background Debug Mode Pin Enable** —When set, this write-once bit enables the PTA4/ACMPO/BKGD/MS pin to function as BKGD/MS. When clear, the pin functions as one of its output only alternative functions. This pin defaults to the BKGD/MS function following any MCU reset. 0 PTA4/ACMPO/BKGD/MS pin functions as PTA4 or ACMPO. 1 PTA4/ACMPO/BKGD/MS pin functions as BKGD/MS.
0 RSTPE	**$\overline{\text{RESET}}$ Pin Enable** — When set, this write-once bit enables the PTA5/TCLK/$\overline{\text{RESET}}$/V_{PP} pin to function as $\overline{\text{RESET}}$. When clear, the pin functions as one of its input-only alternative functions. This pin is input-only port function following an MCU POR. When RSTPE is set, an internal pullup device is enabled on $\overline{\text{RESET}}$. 0 PTA5/TCLK/$\overline{\text{RESET}}$/V_{PP} pin functions as PTA5/TCLK/V_{PP}. 1 PTA5/TCLK/$\overline{\text{RESET}}$/V_{PP} pin functions as $\overline{\text{RESET}}$/V_{PP}.

[1] When the device is power on reset, BKGEPE is reset to 1. When the device is reset into normal operating mode (MS is high during reset), BKGDPE is reset to 1 if flash security is disengaged (SECD = 1). BKGDPE is reset to 0 if flash security is engaged (SECD = 0). When the device is reset into active BDM mode (MS is low during reset), BKGDPE is reset to 1 so that BDM communication is allowed.

[2] BKGDPE can write only once from value 1 to 0. Writing from value 0 to 1 by user software is not allowed. BKGDPE can be changed back to 1 only by a POR or reset with proper condition as stated in Note 1.

5.8.3 System Device Identification Register (SDIDH, SDIDL)

These high-page, read-only registers are included so host development systems can identify the RS08 derivative and revision number. This allows the development software to recognize where specific memory blocks, registers, and control bits are located in a target MCU.

	7	6	5	4	3	2	1	0
R	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0 (Note 1)	0 (Note 1)	0 (Note 1)	0 (Note 1)	1	0	0	0

▨ = Unimplemented or Reserved

[1] The revision number hard coded into these bits reflects the current silicon revision level.

Figure 5-3. System Device Identification Register — High (SDIDH)

Table 5-4. SDIDH Register Field Descriptions

Field	Description
7:4 REV[3:0]	**Revision Number** — The high-order 4 bits of address SDIDH are hard coded to reflect the current mask set revision number (0 – F).
3:0 ID[11:8]	**Part Identification Number** — Each derivative in the RS08 Family has a unique identification number. The SC9RS08MZ8 is hard coded to the value $0803. See also ID bits in Figure 5-4.

	7	6	5	4	3	2	1	0
R	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	1	1

▨ = Unimplemented or Reserved

Figure 5-4. System Device Identification Register — Low (SDIDL)

Table 5-5. SDIDL Register Field Descriptions

Field	Description
7:0 ID[7:0]	**Part Identification Number** — Each derivative in the RS08 Family has a unique identification number. The SC9RS08MZ8 is hard coded to the value $0803. See also ID bits in Figure 5-3.

5.8.4 System Real-Time Interrupt Status and Control Register (SRTISC)

This high-page register contains status and control bits for the RTI.

	7	6	5	4	3	2	1	0
R	RTIF	0	RTICLKS	RTIE	0	RTIS		
W	▨	RTIACK			▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 5-5. System RTI Status and Control Register (SRTISC)

Table 5-6. SRTISC Register Field Descriptions

Field	Description
7 RTIF	**Real-Time Interrupt Flag** — Read-only status bit indicates the periodic wakeup timer has timed out. 0 Periodic wakeup timer not timed out. 1 Periodic wakeup timer timed out.
6 RTIACK	**Real-Time Interrupt Acknowledge** — Write-only bit acknowledges real-time interrupt request (write 1 to clear RTIF). Writing 0 has no meaning or effect. Reads always return 0.
5 RTICLKS	**Real-Time Interrupt Clock Select** — Read/write bit selects the clock source for the real-time interrupt. 0 Real-time interrupt request clock source is internal 1 kHz oscillator. 1 Real-time interrupt request clock source is external clock.

Table 5-6. SRTISC Register Field Descriptions (continued)

Field	Description
4 RTIE	**Real-Time Interrupt Enable** — Read-write bit enables real-time interrupts. 0 Real-time interrupts disabled. 1 Real-time interrupts enabled.
2:0 RTIS	**Real-Time Interrupt Delay Selects** — These read/write bits select the period for the RTI. See Table 5-7.

Table 5-7. Real-Time Interrupt Period

RTIS	Using the 1 kHz Oscillator Source[1]	Using the External Clock Input
000	Disable and clear the RTI Counter	Disable and clear the RTI Counter
001	8 ms	$1/f_{extclk} \times 256$
010	32 ms	$1/f_{extclk} \times 1024$
011	64 ms	$1/f_{extclk} \times 2048$
100	128 ms	$1/f_{extclk} \times 4096$
101	256 ms	$1/f_{extclk} \times 8192$
110	512 ms	$1/f_{extclk} \times 16384$
111	1.024 s	$1/f_{extclk} \times 32768$

[1] Values are shown in this column based on f_{RTI} = 1 kHz. Consult Appendix A, “Electrical Characteristics.”

5.8.5 System Power Management Status and Control 1 Register (SPMSC1)

This high page register contains status and control bits to support the low voltage detect function, and to enable the bandgap voltage reference for use by the ACMP and the LVD module.

	7	6	5	4	3	2	1	0
R	LVDF	0	LVDIE	LVDRE[1]	LVDSE	LVDE[1]	0	BGBE
W	▨	LVDACK					▨	
Reset:	0	0	0	1	1	1	0	0

▨ = Unimplemented or Reserved

[1] This bit can be written only once after reset. Additional writes are ignored.

Figure 5-6. System Power Management Status and Control 1 Register (SPMSC1)

Table 5-8. SPMSC1 Register Field Descriptions

Field	Description
7 LVDF	**Low-Voltage Detect Flag** — Provided LVDE = 1, this read-only status bit indicates a low-voltage detect event.
6 LVDACK	**Low-Voltage Detect Acknowledge** — Write-only bit is used to acknowledge low voltage detection errors (write 1 to clear LVDF). Reads always return 0.
5 LVDIE	**Low-Voltage Detect Interrupt Enable** — Enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling). 1 Request a hardware interrupt when LVDF = 1.
4 LVDRE	**Low-Voltage Detect Reset Enable** — Write-once bit enables low-voltage detect events to generate a hardware reset (provided LVDE = 1). 0 LVDF does not generate hardware resets. 1 Force an MCU reset when LVDF = 1.
3 LVDSE	**Low-Voltage Detect Stop Enable** — Provided LVDE = 1, this read/write bit determines whether the low-voltage-detect function operates when MCU is in stop mode. 0 Low-voltage detect disabled during stop mode. 1 Low-voltage detect enabled during stop mode.
2 LVDE	**Low-Voltage Detect Enable** — Write-once bit enables low-voltage detect logic and the operation of other bits in this register. 0 LVD logic disabled. 1 LVD logic enabled.
0 BGBE	**Bandgap Buffer Enable** — Enables an internal buffer for the bandgap-voltage reference for use by the ACMP module on one of its internal channels. 0 Bandgap buffer disabled. 1 Bandgap buffer enabled.

5.8.6 System Interrupt Pending Register (SIP1)

This high-page register contains status of the pending interrupt from the modules.

	7	6	5	4	3	2	1	0
R	▨	▨	ACMP	ADC	TPM	MTIM2	MTIM1	RTI
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 5-7. System Interrupt Pending Register (SIP1)

Table 5-9. SIP1 Register Field Descriptions

Field	Description
5 ACMP	**Analog Comparator Interrupt Pending** — Read-only bit indicates a pending interrupt from the ACMP module. Clearing the ACF flag of the ACMPSC register clears this bit. Reset also clears this bit. 0 No pending ACMP interrupt; i.e., ACF flag and/or ACIE bit is cleared. 1 Pending a ACMP interrupt; i.e., ACF flag and ACIE bit are set.
4 ADC	**ADC Interrupt Pending —** Read-only bit indicates a pending interrupt from the ADC module. Clearing the COCO flag of the ADCSC1 register clears this bit. Reset also clears this bit. 0 No pending ADC interrupt, i.e., COCO flag and/or AIEN bit is cleared. 1 Pending ADC interrupt, i.e., COCO flag and AIEN bit are set.
3 TPM	**Timer/PWM Interrupt Pending —** Read-only bit indicates a pending interrupts from the TPM module. Clearing the TOF flag of the TPMSC register and/or CHnF flags of the TPMCnSC register clears this bit. Reset also clears this bit. 0 No pending Timer/PWM interrupt i.e., either TOF flag and/or TOIE bit is cleared.; Or CH0F flag and/or CH0IE bit is cleared.; Or CH1F flag and/or CH1IE bit is cleared. 1 Pending Timer/PWM interrupt, i.e., either TOF flag and TOIE bit is set.; Or CH0F flag and CH0IE bit is set.; Or CH1F flag and CH1IE bit is set.
2 MTIM2	**Modulo Timer 2 Interrupt Pending** — Read-only bit indicates a pending interrupt from the MTIM2 module. Clearing the TOF flag of the MTIM2SC register clears this bit. Reset also clears this bit. 0 No pending MTIM2 interrupt; i.e., TOF flag and/or TOIE bit is cleared. 1 Pending MTIM2 interrupt; i.e., TOF flag and TOIE bit are set.
1 MTIM1	**Modulo Timer 1 Interrupt Pending** — Read-only bit indicates a pending interrupt from the MTIM1 module. Clearing the TOF flag of the MTIM1SC register clears this bit. Reset also clears this bit. 0 No pending MTIM1 interrupt; i.e., TOF flag and/or TOIE bit is cleared. 1 Pending MTIM1 interrupt; i.e., TOF flag and TOIE bit are set.
0 RTI	**Real-Time Interrupt Pending** — Read-only bit indicates a pending interrupt from the RTI. Clearing the RTIF flag of the SRTISC register clears this bit. Reset also clears this bit. 0 No pending RTI interrupt; i.e., RTIF flag and/or RTIE bit is cleared. 1 Pending RTI interrupt; i.e., RTIF flag and RTIE bit are set.

Chapter 6
Parallel Input/Output Control

This chapter explains software controls related to parallel input/output (I/O) and pin control. See Chapter 2, "Pins and Connections," for more information about pin assignments and external hardware considerations for these pins.

All these I/O pins are shared with on-chip peripheral functions (see Table 2-1). The peripheral modules have priority over the I/Os. When a peripheral is enabled, the I/O functions associated with the shared pins are disabled. After reset, the shared peripheral functions are disabled so the I/O controls the pins. All the I/Os are configured as inputs (PTADDn = 0) with pullup/pulldown devices disabled (PTAPEn = 0), except for output-only pin PTA4, which defaults to the BKGD/MS function. All pins' default-low-drive strengths are selected (PTxDSn= 0) after reset

Reading and writing parallel I/Os is performed through the port data registers. The direction, either input or output, is controlled through the port data direction registers. The block diagram in Figure 6-1 illustrates the parallel I/O port function for an individual pin.

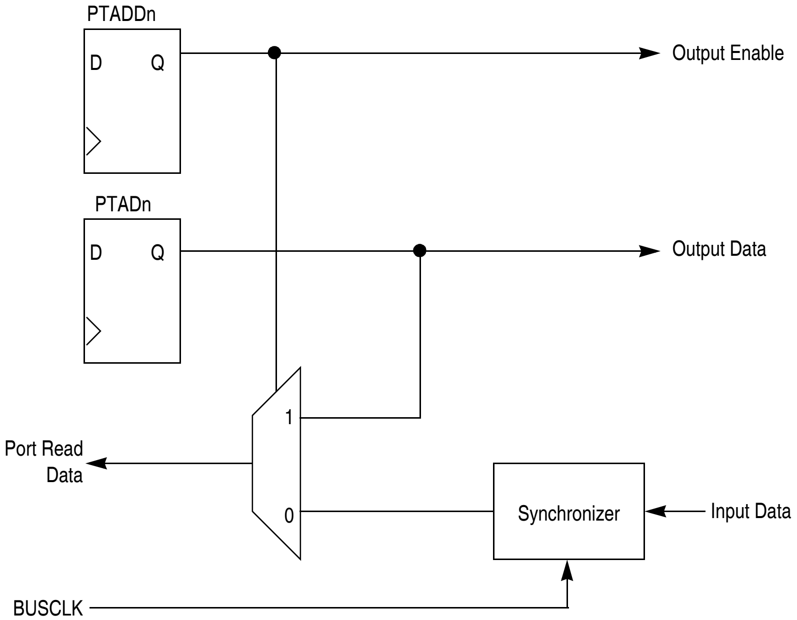


Figure 6-1. Parallel I/O Block Diagram

The data direction control bit (PTADDn) determines whether the output buffer for the associated pin is enabled, and also controls the source for port data register reads. The input buffer for the associated pin is always enabled unless the pin is enabled as an analog function or is an output-only pin.

When a shared digital function is enabled for a pin, the shared function controls the output buffer. However, the data direction register bit continues controlling the source for reads of the port data register.

When a shared analog function is enabled for a pin, the input and output buffers are disabled. A value of 0 is read for any port data bit where the bit is an input (PTADDn = 0) and the input buffer is disabled. In general, whenever a pin is shared with an alternative digital function and an analog function, the analog function has priority such that if the digital and analog functions are enabled, the analog function controls the pin.

Write to the port data register before changing the direction of a port pin to become an output. This ensures the pin is not driven temporarily with an old data value that happened to be in the port data register.

A set of registers associated with the parallel I/O ports is located in the high page register space that operate independently of the parallel I/O registers. These registers are used to control pullup/pulldown and slew rate for the pins. See Section 6.3, "Pin Control Registers."

6.1 Pin Behavior in Low-Power Modes

In wait and stop modes, all pin states are maintained because internal logic stays powered up. Upon recovery, all pin functions revert to the state they were in prior to entering stop mode.

6.2 Parallel I/O Registers

This section provides information about registers associated with the parallel I/O ports. The parallel I/O data registers are located within the $001F memory boundary of the memory map, so that short and direct addressing mode instructions can be used.

Refer to the tables in Chapter 4, "Memory," for the absolute address assignments for all parallel I/O. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file is normally used to translate these names into the appropriate absolute addresses.

6.2.1 Port A Registers

Port A parallel I/O function is controlled by the data and data direction registers described in this section.

	7	6	5	4	3	2	1	0
R	0	0	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-2. Port A Data Register (PTAD)

Table 6-1. PTAD Register Field Descriptions

Field	Description
5:0 PTAD[5:0]	**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 on the corresponding MCU pin. Reset forces PTAD to all 0s, but these 0s are not driven out on the corresponding pins because reset also configures all port pins as high-impedance inputs with pullup/pulldowns disabled.

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

Figure 6-3. Port A Data Direction Register (PTADD)

Table 6-2. PTADD Register Field Descriptions

Field	Description
3:0 PTADD[3:0]	**Data Direction for Port A Bits** — These read/write bits control the port A pins direction 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.

6.2.2 Port B Registers

Port B parallel I/O function is controlled by the data and data direction registers described in this section.

	7	6	5	4	3	2	1	0
R W	PTBD7	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2	PTBD1	PTBD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-4. Port B Data Register (PTBD)

Table 6-3. PTBD Register Field Descriptions

Field	Description
7:0 PTBD[7:0]	**Port B Data Register Bits** — For port B pins that are inputs, reads return the logic level on the pin. For port B 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 B pins configured as outputs, the logic level is driven out on the corresponding MCU pin. Reset forces PTBD to all 0s, but these 0s are not driven out on the corresponding pins because reset also configures all port pins as high-impedance inputs with pullup/pulldowns disabled.

	7	6	5	4	3	2	1	0
R W	PTBDD7	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-5. Port B Data Direction Register (PTBDD)

Table 6-4. PTBDD Register Field Descriptions

Field	Description
7:0 PTBDD[7:0]	**Data Direction for Port B Bits** — These read/write bits control the port B pins direction and what is read for PTBD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port B bit n and PTBD reads return the contents of PTBDn.

6.2.3 Port C Registers

Port C parallel I/O function is controlled by the data and data direction registers described in this section.

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTCD3	PTCD2	PTCD1	PTCD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-6. Port C Data Register (PTCD)

Table 6-5. PTCD Register Field Descriptions

Field	Description
3:0 PTCD[3:0]	**Port C Data Register Bits** — For port C pins that are inputs, reads return the logic level on the pin. For port C pins configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port C pins configured as outputs, the logic level is driven out on the corresponding MCU pin. Reset forces PTCD to all 0s, but these 0s are not driven out on the corresponding pins because reset also configures all port pins as high-impedance inputs with pullup/pulldowns disabled.

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTCDD3	PTCDD2	PTCDD1	PTCDD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-7. Port C Data Direction Register (PTCDD)

Table 6-6. PTCDD Register Field Descriptions

Field	Description
3:0 PTCDD[3:0]	**Data Direction for Port C Bits** — These read/write bits control the direction of port C pins and what is read for PTCD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port C bit n and PTCD reads return the contents of PTCDn.

6.3 Pin Control Registers

This section provides information about the registers associated with the parallel I/O ports that are used for pin control functions.

Refer to the tables in Chapter 4, “Memory,” for the absolute address assignments of the pin control registers. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

NOTE

> 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, do not exceed the total current source and sink limits for the MCU. Drive strength selection affects 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.

6.3.1 Port A Pin Control Registers

The pins associated with port A are controlled by the registers provided in this section. These registers control the pin pullup/pulldown and slew rate of the port A pins independent of the parallel I/O registers.

6.3.1.1 Internal Pulling Device Enable

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

	7	6	5	4	3	2	1	0
R	0	0	PTAPE5	0	PTAPE3	PTAPE2	PTAPE1	PTAPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-8. Internal Pulling Device Enable for Port A Register (PTAPE)

Table 6-7. PTAPE Register Field Descriptions

Field	Description
5,3:0 PTAPE[5,3:0]	**Internal Pulling Device Enable for Port A Bits** — Each of these control bits determines whether the internal pulling device is enabled for the associated PTA pin. For port A pins configured as outputs, these bits have no effect and the internal pullup devices are disabled. 0 Internal pulling device disabled for port A bit n. 1 Internal pulling device enabled for port A bit n.

6.3.1.2 Pullup/Pulldown Control

Pullup/pulldown control is used to select the pullup or pulldown device enabled by the corresponding PTAPE bit.

	7	6	5	4	3	2	1	0
R	0	0	PTAPUD5	0	PTAPUD3	PTAPUD2	PTAPUD1	PTAPUD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-9. Pullup/Pulldown Device Control for Port A (PTAPUD)

Table 6-8. PTAPUD Register Field Descriptions

Field	Description
5,3:0 PTAPUD[5,3:0]	**Pullup/Pulldown Device Control for Port A Bits** — Each of these control bits determines whether the internal pullup or pulldown device is selected for the associated PTA pin. The actual pullup/pulldown device is only enabled by enabling the associated PTAPE bit. 0 Internal pullup device is selected for port A bit n. 1 Internal pulldown device is selected for port A bit n.

6.3.1.3 Output Slew Rate Control Enable

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

	7	6	5	4	3	2	1	0
R	0	0	0	PTASE4	PTASE3	PTASE2	PTASE1	PTASE0
W								
Reset:	0	0	0	1	1	1	1	1

Figure 6-10. Slew Rate Enable for Port A Register (PTASE)

Table 6-9. PTASE Register Field Descriptions

Field	Description
4:0 PTASE[4:0]	**Output Slew Rate Enable for Port A Bits** — Each of these control bits determines whether the output slew rate control is enabled for the associated PTA pin. For port A pins configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port A bit n. 1 Output slew rate control enabled for port A bit n.

6.3.1.4 Port A Drive Strength Selection Register (PTADS)

	7	6	5	4	3	2	1	0
R	0	0	0	PTADS4	PTADS3	PTADS2	PTADS1	PTADS0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-11. Output Drive Strength Selection for Port A (PTADS)

Table 6-10. PTADS Register Field Descriptions

Field	Description
4:0 PTADS[4:0]	**Output Drive Strength Selection for Port A Bits** — Each of these control bits selects between low and high output drive for the associated PTA pin. 0 Low output drive enabled for port A bit n. 1 High output drive enabled for port A bit n.

6.3.2 Port B Pin Control Registers

The pins associated with port B are controlled by the registers provided in this section. These registers control the pin pullup/pulldown and slew rate of the port B pins independent of the parallel I/O registers.

6.3.2.1 Internal Pulling Device Enable

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

	7	6	5	4	3	2	1	0
R W	PTBPE7	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
Reset:	0	0	0	0	0	0	0	0

Figure 6-12. Internal Pulling Device Enable for Port B Register (PTBPE)

Table 6-11. PTBPE Register Field Descriptions

Field	Description
7:0 PTBPE[7:0]	**Internal Pulling Device Enable for Port A Bits** — Each of these control bits determines whether the internal pulling device is enabled for the associated PTB pin. For port B pins that are configured as outputs, these bits have no effect and the internal pullup devices are disabled. 0 Internal pulling device disabled for port B bit n. 1 Internal pulling device enabled for port B bit n.

6.3.2.2 Pullup/Pulldown Control

Pullup/pulldown control is used to select the pullup or pulldown device enabled by the corresponding PTBPE bit.

	7	6	5	4	3	2	1	0
R W	PTBPUD7	PTBPUD6	PTBPUD5	PTBPUD4	PTBPUD3	PTBPUD2	PTBPUD1	PTBPUD0
Reset:	0	0	0	0	0	0	0	0

Figure 6-13. Pullup/Pulldown Device Control for Port B (PTBPUD)

Table 6-12. PTBPUD Register Field Descriptions

Field	Description
7:0 PTBPUD[7:0]	**Pullup/Pulldown Device Control for Port B Bits** — Each of these control bits determines whether the internal pullup or pulldown device is selected for the associated PTB pin. The actual pullup/pulldown device is only enabled by enabling the associated PTBPE bit. 0 Internal pullup device is selected for port B bit n. 1 Internal pulldown device is selected for port B bit n.

6.3.2.3 Output Slew Rate Control Enable

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

	7	6	5	4	3	2	1	0
R W	PTBSE7	PTBSE6	PTBSE5	PTBSE4	PTBSE3	PTBSE2	PTBSE1	PTBSE0
Reset:	1	1	1	1	1	1	1	1

Figure 6-14. Slew Rate Enable for Port B Register (PTBSE)

Table 6-13. PTBSE Register Field Descriptions

Field	Description
7:0 PTBSE[7:0]	**Output Slew Rate Enable for Port B Bits** — Each of these control bits determines whether the output slew rate control is enabled for the associated PTB pin. For port B pins that are configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port B bit n. 1 Output slew rate control enabled for port B bit n.

6.3.2.4 Port B Drive Strength Selection Register (PTBDS)

	7	6	5	4	3	2	1	0
R W	PTBDS7	PTBDS6	PTBDS5	PTBDS4	PTBDS3	PTBDS2	PTBDS1	PTBDS0
Reset	0	0	0	0	0	0	0	0

Figure 6-15. Output Drive Strength Selection for Port B (PTBDS)

Table 6-14. PTBDS Register Field Descriptions

Field	Description
7:0 PTBDS[7:0]	**Output Drive Strength Selection for Port B Bits** — Each of these control bits selects between low and high output drive for the associated PTB pin. 0 Low output drive enabled for port B bit n. 1 High output drive enabled for port B bit n.

6.3.3 Port C Pin Control Registers

The pins associated with port C are controlled by the registers provided in this section. These registers control the pin pullup/pulldown and slew rate of the port C pins independent of the parallel I/O registers.

6.3.3.1 Internal Pulling Device Enable

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

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTCPE3	PTCPE2	PTCPE1	PTCPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-16. Internal Pulling Device Enable for Port C Register (PTCPE)

Table 6-15. PTCPE Register Field Descriptions

Field	Description
3:0 PTCPE[3:0]	**Internal Pulling Device Enable for Port C Bits** — Each of these control bits determines whether the internal pulling device is enabled for the associated PTC pin. For port C pins that are configured as outputs, these bits have no effect and the internal pullup devices are disabled. 0 Internal pulling device disabled for port C bit n. 1 Internal pulling device enabled for port C bit n.

6.3.3.2 Pullup/Pulldown Control

Pullup/pulldown control is used to select the pullup or pulldown device enabled by the corresponding PTCPE bit.

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTCPUD3	PTCPUD2	PTCPUD1	PTCPUD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-17. Pullup/Pulldown Device Control for Port C (PTCPUD)

Table 6-16. PTCPUD Register Field Descriptions

Field	Description
3:0 PTCPUD[3:0]	**Pullup/Pulldown Device Control for Port C Bits** — Each of these control bits determines whether the internal pullup or pulldown device is selected for the associated PTC pin. The actual pullup/pulldown device is only enabled by enabling the associated PTCPE bit. 0 Internal pullup device is selected for port C bit n. 1 Internal pulldown device is selected for port C bit n.

6.3.3.3 Output Slew Rate Control Enable

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

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTCSE3	PTCSE2	PTCSE1	PTCSE0
W								
Reset:	0	0	0	0	1	1	1	1

Figure 6-18. Slew Rate Enable for Port C Register (PTCSE)

Table 6-17. PTCSE Register Field Descriptions

Field	Description
3:0 PTCSE[3:0]	**Output Slew Rate Enable for Port C Bits** — Each of these control bits determines whether the output slew rate control is enabled for the associated PTC pin. For port C pins that are configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port C bit n. 1 Output slew rate control enabled for port C bit n.

6.3.3.4 Port C Drive Strength Selection Register (PTCDS)

	7	6	5	4	3	2	1	0
R	0	0	0	0	PTCDS3	PTCDS2	PTCDS1	PTCDS0
W								
Reset	0	0	0	0	0	0	0	0

Figure 6-19. Output Drive Strength Selection for Port C (PTCDS)

Table 6-18. PTCDS Register Field Descriptions

Field	Description
3:0 PTCDS[3:0]	**Output Drive Strength Selection for Port C Bits** — Each of these control bits is selected between low and high output drive for the associated PTC pin. 0 Low output drive enabled for port C bit n. 1 High output drive enabled for port C bit n.

Chapter 7
Central Processor Unit (RS08CPUV1)

7.1 Introduction

This chapter is a summary of information about the registers, addressing modes, and instruction set of the RS08 family CPU. For a more detailed discussion, refer to the RS08 Core Reference Manual, volume 1, Freescale Semiconductor document order number RS08RMv1.

The RS08 CPU has been developed to target extremely low-cost embedded applications using a process-independent design methodology, allowing it to keep pace with rapid developments in silicon processing technology.

The main features of the RS08 core are:

- Streamlined programmer's model
- Subset of HCS08 instruction set with minor instruction extensions
- Minimal instruction set for cost-sensitive embedded applications
- New instructions for shadow program counter manipulation, SHA and SLA
- New short and tiny addressing modes for code size optimization
- 16K bytes accessible memory space
- Reset will fetch the first instruction from $3FFD
- Low-power modes supported through the execution of the STOP and WAIT instructions
- Debug and FLASH programming support using the background debug controller module
- Illegal address and opcode detection with reset

7.2 Programmer's Model and CPU Registers

Figure 7-1 shows the programmer's model for the RS08 CPU. These registers are not located in the memory map of the microcontroller. They are built directly inside the CPU logic.

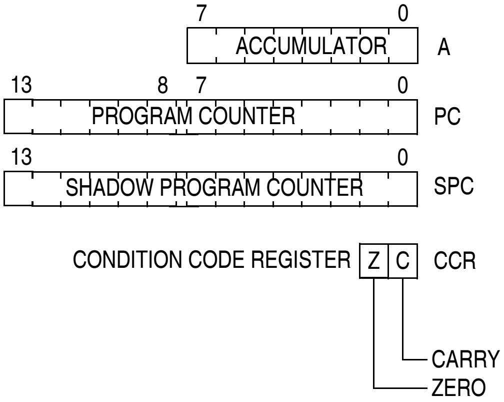


Figure 7-1. CPU Registers

In addition to the CPU registers, there are three memory mapped registers that are tightly coupled with the core address generation during data read and write operations. They are the indexed data register (D[X]), the index register (X), and the page select register (PAGESEL). These registers are located at $000E, $000F, and $001F, respectively.

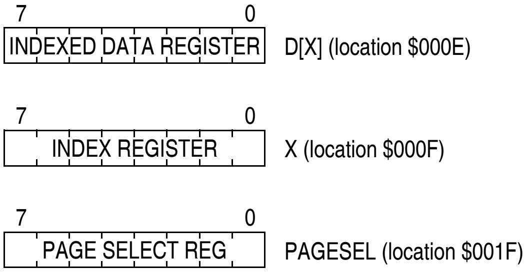


Figure 7-2. Memory Mapped Registers

7.2.1 Accumulator (A)

This general-purpose 8-bit register is the primary data register for RS08 MCUs. Data can be read from memory into A with a load accumulator (LDA) instruction. The data in A can be written into memory with a store accumulator (STA) instruction. Various addressing mode variations allow a great deal of flexibility in specifying the memory location involved in a load or store instruction. Exchange instructions allow values to be exchanged between A and SPC high (SHA) and also between A and SPC low (SLA).

Arithmetic, shift, and logical operations can be performed on the value in A as in ADD, SUB, RORA, INCA, DECA, AND, ORA, EOR, etc. In some of these instructions, such as INCA and LSLA, the value in A is the only input operand and the result replaces the value in A. In other cases, such as ADD and AND, there are two operands: the value in A and a second value from memory. The result of the arithmetic or logical operation replaces the value in A.

Some instructions, such as memory-to-memory move instructions (MOV), do not use the accumulator. DBNZ also relieves A because it allows a loop counter to be implemented in a memory variable rather than the accumulator.

During reset, the accumulator is loaded with $00.

7.2.2 Program Counter (PC)

The program counter is a 14-bit register that contains the address of the next instruction or operand to be fetched.

During normal execution, the program counter automatically increments to the next sequential memory location each time an instruction or operand is fetched. Jump, branch, and return operations load the program counter with an address other than that of the next sequential location. This is called a change-of-flow.

During reset, the program counter is loaded with $3FFD and the program will start execution from this specific location.

7.2.3 Shadow Program Counter (SPC)

The shadow program counter is a 14-bit register. During a subroutine call using either a JSR or a BSR instruction, the return address will be saved into the SPC. Upon completion of the subroutine, the RTS instruction will restore the content of the program counter from the shadow program counter.

During reset, the shadow program counter is loaded with $3FFD.

7.2.4 Condition Code Register (CCR)

The 2-bit condition code register contains two status flags. The content of the CCR in the RS08 is not directly readable. The CCR bits can be tested using conditional branch instructions such as BCC and BEQ. These two register bits are directly accessible through the BDC interface. The following paragraphs provide detailed information about the CCR bits and how they are used. Figure 7-3 identifies the CCR bits and their bit positions.

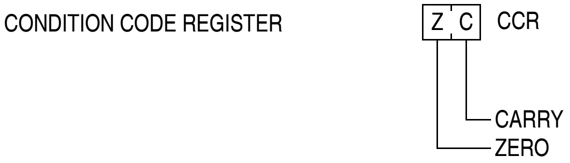


Figure 7-3. Condition Code Register (CCR)

The status bits (Z and C) are cleared to 0 after reset.

The two status bits indicate the results of arithmetic and other instructions. Conditional branch instructions will either branch to a new program location or allow the program to continue to the next instruction after the branch, depending on the values in the CCR status bit. Conditional branch instructions, such as BCC, BCS, and BNE, cause a branch depending on the state of a single CCR bit.

Often, the conditional branch immediately follows the instruction that caused the CCR bit(s) to be updated, as in this sequence:

```
         cmp    #5       ;compare accumulator A to 5
         blo    lower    ;branch if A smaller 5
more:    deca           ;do this if A not higher than or same as 5
lower:
```

Other instructions may be executed between the test and the conditional branch as long as the only instructions used are those which do not disturb the CCR bits that affect the conditional branch. For instance, a test is performed in a subroutine or function and the conditional branch is not executed until the subroutine has returned to the main program. This is a form of parameter passing (that is, information is returned to the calling program in the condition code bits).

Z — Zero Flag

The Z bit is set to indicate the result of an operation was $00.

Branch if equal (BEQ) and branch if not equal (BNE) are simple branches that branch based solely on the value in the Z bit. All load, store, move, arithmetic, logical, shift, and rotate instructions cause the Z bit to be updated.

C — Carry

After an addition operation, the C bit is set if the source operands were both greater than or equal to $80 or if one of the operands was greater than or equal to $80 and the result was less than $80. This is equivalent to an unsigned overflow. A subtract or compare performs a subtraction of a memory operand from the contents of a CPU register so after a subtract operation, the C bit is set if the unsigned value of the memory operand was greater than the unsigned value of the CPU register. This is equivalent to an unsigned borrow or underflow.

Branch if carry clear (BCC) and branch if carry set (BCS) are branches that branch based solely on the value in the C bit. The C bit is also used by the unsigned branches BLO and BHS. Add, subtract, shift, and rotate instructions cause the C bit to be updated. The branch if bit set (BRSET) and branch if bit clear (BRCLR) instructions copy the tested bit into the C bit to facilitate efficient serial-to-parallel conversion algorithms. Set carry (SEC) and clear carry (CLC) allow the carry bit to be set or cleared directly. This is useful in combination with the shift and rotate instructions and for routines that pass status information back to a main program, from a subroutine, in the C bit.

The C bit is included in shift and rotate operations so those operations can easily be extended to multi-byte operands. The shift and rotate operations can be considered 9-bit shifts that include an 8-bit operand or CPU register and the carry bit of the CCR. After a logical shift, C holds the bit that was shifted out of the 8-bit operand. If a rotate instruction is used next, this C bit is shifted into the operand for the rotate, and the bit that gets shifted out the other end of the operand replaces the value in C so it can be used in subsequent rotate instructions.

7.2.5 Indexed Data Register (D[X])

This 8-bit indexed data register allows the user to access the data in the direct page address space indexed by X. This register resides at the memory mapped location $000E. For details on the D[X] register, please refer to Section 7.3.8, “Indexed Addressing Mode (IX, Implemented by Pseudo Instructions).”

7.2.6 Index Register (X)

This 8-bit index register allows the user to index or address any location in the direct page address space. This register resides at the memory mapped location $000F. For details on the X register, please refer to Section 7.3.8, “Indexed Addressing Mode (IX, Implemented by Pseudo Instructions).”

7.2.7 Page Select Register (PAGESEL)

This 8-bit page select register allows the user to access all memory locations in the entire 16K-byte address space through a page window located from $00C0 to $00FF. This register resides at the memory mapped location $001F. For details on the PAGESEL register, please refer to the RS08 Core Reference Manual.

7.3 Addressing Modes

Whenever the MCU reads information from memory or writes information into memory, an addressing mode is used to determine the exact address where the information is read from or written to. This section explains several addressing modes and how each is useful in different programming situations.

Every opcode tells the CPU to perform a certain operation in a certain way. Many instructions, such as load accumulator (LDA), allow several different ways to specify the memory location to be operated on, and each addressing mode variation requires a separate opcode. All of these variations use the same instruction mnemonic, and the assembler knows which opcode to use based on the syntax and location of the operand field. In some cases, special characters are used to indicate a specific addressing mode (such as the # [pound] symbol, which indicates immediate addressing mode). In other cases, the value of the operand tells the assembler which addressing mode to use. For example, the assembler chooses short addressing mode instead of direct addressing mode if the operand address is from $0000 to $001F. Besides allowing the assembler to choose the addressing mode based on the operand address, assembler directives can also be used to force direct or tiny/short addressing mode by using the “>” or “<” prefix before the operand, respectively.

Some instructions use more than one addressing mode. For example, the move instructions use one addressing mode to access the source value from memory and a second addressing mode to access the destination memory location. For these move instructions, both addressing modes are listed in the documentation. All branch instructions use relative (REL) addressing mode to determine the destination for the branch, but BRCLR, BRSET, CBEQ, and DBNZ also must access a memory operand. These instructions are classified by the addressing mode used for the memory operand, and the relative addressing mode for the branch offset is assumed.

The discussion in the following paragraphs includes how each addressing mode works and the syntax clues that instruct the assembler to use a specific addressing mode.

7.3.1 Inherent Addressing Mode (INH)

This addressing mode is used when the CPU inherently knows everything it needs to complete the instruction and no addressing information is supplied in the source code. Usually, the operands that the CPU needs are located in the CPU’s internal registers, as in LSLA, CLRA, INCA, SLA, RTS, and others. A few inherent instructions, including no operation (NOP) and background (BGND), have no operands.

7.3.2 Relative Addressing Mode (REL)

Relative addressing mode is used to specify the offset address for branch instructions relative to the program counter. Typically, the programmer specifies the destination with a program label or an expression in the operand field of the branch instruction; the assembler calculates the difference between the location counter (which points at the next address after the branch instruction at the time) and the address represented by the label or expression in the operand field. This difference is called the offset and is an 8-bit two's complement number. The assembler stores this offset in the object code for the branch instruction.

During execution, the CPU evaluates the condition that controls the branch. If the branch condition is true, the CPU sign-extends the offset to a 14-bit value, adds the offset to the current PC, and uses this as the address where it will fetch the next instruction and continue execution rather than continuing execution with the next instruction after the branch. Because the offset is an 8-bit two's complement value, the destination must be within the range –128 to +127 locations from the address that follows the last byte of object code for the branch instruction.

A common method to create a simple infinite loop is to use a branch instruction that branches to itself. This is sometimes used to end short code segments during debug. Typically, to get out of this infinite loop, use the debug host (through background commands) to stop the program, examine registers and memory, or to start execution from a new location. This construct is not used in normal application programs except in the case where the program has detected an error and wants to force the COP watchdog timer to timeout. (The branch in the infinite loop executes repeatedly until the watchdog timer eventually causes a reset.)

7.3.3 Immediate Addressing Mode (IMM)

In this addressing mode, the operand is located immediately after the opcode in the instruction stream. This addressing mode is used when the programmer wants to use an explicit value that is known at the time the program is written. A # (pound) symbol is used to tell the assembler to use the operand as a data value rather than an address where the desired value will be accessed.

The size of the immediate operand is always 8 bits. The assembler automatically will truncate or extend the operand as needed to match the size needed for the instruction. Most assemblers generate a warning if a 16-bit operand is provided.

It is the programmer's responsibility to use the # symbol to tell the assembler when immediate addressing will be used. The assembler does not consider it an error to leave off the # symbol because the resulting statement is still a valid instruction (although it may mean something different than the programmer intended).

7.3.4 Tiny Addressing Mode (TNY)

TNY addressing mode is capable of addressing only the first 16 bytes in the address map, from $0000 to $000F. This addressing mode is available for INC, DEC, ADD, and SUB instructions. A system can be optimized by placing the most computation-intensive data in this area of memory.

Because the 4-bit address is embedded in the opcode, only the least significant four bits of the address must be included in the instruction; this saves program space and execution time. During execution, the CPU adds 10 high-order 0s to the 4-bit operand address and uses the combined 14-bit address ($000x) to access the intended operand.

7.3.5 Short Addressing Mode (SRT)

SRT addressing mode is capable of addressing only the first 32 bytes in the address map, from $0000 to $001F. This addressing mode is available for CLR, LDA, and STA instructions. A system can be optimized by placing the most computation-intensive data in this area of memory.

Because the 5-bit address is embedded in the opcode, only the least significant five bits of the address must be included in the instruction; this saves program space and execution time. During execution, the CPU adds nine high-order 0s to the 5-bit operand address and uses the combined 14-bit address ($000x or $001x) to access the intended operand.

7.3.6 Direct Addressing Mode (DIR)

DIR addressing mode is used to access operands located in direct address space ($0000 through $00FF).

During execution, the CPU adds six high-order 0s to the low byte of the direct address operand that follows the opcode. The CPU uses the combined 14-bit address ($00xx) to access the intended operand.

7.3.7 Extended Addressing Mode (EXT)

In the extended addressing mode, the 14-bit address of the operand is included in the object code in the low-order 14 bits of the next two bytes after the opcode. This addressing mode is only used in JSR and JMP instructions for jump destination address in RS08 MCUs.

7.3.8 Indexed Addressing Mode (IX, Implemented by Pseudo Instructions)

Indexed addressing mode is sometimes called indirect addressing mode because an index register is used as a reference to access the intended operand.

An important feature of indexed addressing mode is that the operand address is computed during execution based on the current contents of the X index register located in $000F of the memory map rather than being a constant address location that was determined during program assembly. This allows writing of a program that accesses different operand locations depending on the results of earlier program instructions (rather than accessing a location that was determined when the program was written).

The index addressing mode supported by the RS08 Family uses the register X located at $000F as an index and D[X] register located at $000E as the indexed data register. By programming the index register X, any location in the direct page can be read/written via the indexed data register D[X].

These pseudo instructions can be used with all instructions supporting direct, short, and tiny addressing modes by using the D[X] as the operand.

7.4 Special Operations

Most of what the CPU does is described by the instruction set, but a few special operations must be considered, such as how the CPU starts at the beginning of an application program after power is first applied. After the program begins running, the current instruction normally determines what the CPU will do next. Two exceptional events can cause the CPU to temporarily suspend normal program execution:

- Reset events force the CPU to start over at the beginning of the application program, which forces execution to start at $3FFD.
- A host development system can cause the CPU to go to active background mode rather than continuing to the next instruction in the application program.

7.4.1 Reset Sequence

Processing begins at the trailing edge of a reset event. The number of things that can cause reset events can vary slightly from one RS08 derivative to another; however, the most common sources are: power-on reset, the external $\overline{\text{RESET}}$ pin, low-voltage reset, COP watchdog timeout, illegal opcode detect, and illegal address access. For more information about how the MCU recognizes reset events and determines the difference between internal and external causes, refer to the Resets and Interrupts chapter.

Reset events force the MCU to immediately stop what it is doing and begin responding to reset. Any instruction that was in process will be aborted immediately without completing any remaining clock cycles. A short sequence of activities is completed to decide whether the source of reset was internal or external and to record the cause of reset. For the remainder of the time, the reset source remains active and the internal clocks are stopped to save power. At the trailing edge of the reset event, the clocks resume and the CPU exits from the reset condition. The program counter is reset to $3FFD and an instruction fetch will be started after the release of reset.

For the device to execute code from the on-chip memory starting from $3FFD after reset, care must be taken to not force the BKDG pin low on the end of reset because this will force the device into active background mode where the CPU will wait for a command from the background communication interface.

7.4.2 Interrupts

The interrupt mechanism in RS08 is not used to interrupt the normal flow of instructions; it is used to wake up the RS08 from wait and stop modes. In run mode, interrupt events must be polled by the CPU. The interrupt feature is not compatible with Freescale’s HC05, HC08, or HCS08 Families.

7.4.3 Wait and Stop Mode

Wait and stop modes are entered by executing a WAIT or STOP instruction, respectively. In these modes, the clocks to the CPU are shut down to save power and CPU activity is suspended. The CPU remains in this low-power state until an interrupt or reset event wakes it up. Please refer to the Resets and Interrupts chapter for the effects of wait and stop on other device peripherals.

7.4.4 Active Background Mode

Active background mode refers to the condition in which the CPU has stopped executing user program instructions and is waiting for serial commands from the background debug system. Refer to the Development Support chapter for detailed information on active background mode.

The arithmetic left shift pseudo instruction is also available because its operation is identical to logical shift left.

7.5 Summary Instruction Table

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in Table 7-1 through Table 7-2.

Operators

( ) = Contents of register or memory location shown inside parentheses
← = Is loaded with (read: "gets")
⇔ = Exchange with
& = Boolean AND
| = Boolean OR
⊕ = Boolean exclusive-OR
: = Concatenate
+ = Add

CPU registers

A = Accumulator
CCR = Condition code register
PC = Program counter
PCH = Program counter, higher order (most significant) six bits
PCL = Program counter, lower order (least significant) eight bits
SPC = Shadow program counter
SPCH = Shadow program counter, higher order (most significant) six bits
SPCL = Shadow program counter, lower order (least significant) eight bits

Memory and addressing

M = A memory location or absolute data, depending on addressing mode
rel = The relative offset, which is the two's complement number stored in the last byte of machine code corresponding to a branch instruction
X = Pseudo index register, memory location $000F
,X or D[X] = Memory location $000E pointing to the memory location defined by the pseudo index register (location $000F)

Condition code register (CCR) bits

Z = Zero indicator
C = Carry/borrow

CCR activity notation

– = Bit not affected
0 = Bit forced to 0
1 = Bit forced to 1
Þ = Bit set or cleared according to results of operation
U = Undefined after the operation

Machine coding notation

dd = Low-order eight bits of a direct address $0000–$00FF (high byte assumed to be $00)

ii = One byte of immediate data

hh = High-order 6-bit of 14-bit extended address prefixed with 2-bit of 0

ll = Low-order byte of 14-bit extended address

rr = Relative offset

Source form

Everything in the source forms columns, *except expressions in italic characters,* is literal information which must appear in the assembly source file exactly as shown. The initial 3- to 5-letter mnemonic is always a literal expression. All commas, pound signs (#), parentheses, and plus signs (+) are literal characters.

n — Any label or expression that evaluates to a single integer in the range 0–7.

x — Any label or expression that evaluates to a single hexadecimal integer in the range $0–$F.

opr8i — Any label or expression that evaluates to an 8-bit immediate value.

opr4a — Any label or expression that evaluates to a Tiny address (4-bit value). The instruction treats this 4-bit value as the low order four bits of an address in the 16-Kbyte address space ($0000–$000F). This 4-bit value is embedded in the low order four bits in the opcode.

opr5a — Any label or expression that evaluates to a Short address (5-bit value). The instruction treats this 5-bit value as the low order five bits of an address in the 16-Kbyte address space ($0000–$001F). This 5-bit value is embedded in the low order 5 bits in the opcode.

opr8a — Any label or expression that evaluates to an 8-bit value. The instruction treats this 8-bit value as the low order eight bits of an address in the 16-Kbyte address space ($0000–$00FF).

opr16a — Any label or expression that evaluates to a 14-bit value. On the RS08 core, the upper two bits are always 0s. The instruction treats this value as an address in the 16-Kbyte address space.

rel — Any label or expression that refers to an address that is within –128 to +127 locations from the next address after the last byte of object code for the current instruction. The assembler will calculate the 8-bit signed offset and include it in the object code for this instruction.

Address modes

INH = Inherent (no operands)

IMD = Immediate to Direct (in MOV instruction)

IMM = Immediate

DD = Direct to Direct (in MOV instruction)

DIR = Direct

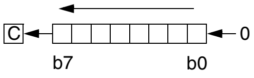
SRT = Short

TNY = Tiny

EXT = Extended

REL = 8-bit relative offset

Table 7-1. Instruction Set Summary (Sheet 1 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
ADC #*opr8i* ADC *opr8a* ADC ,X (1) ADC X	Add with Carry	A ← (A) + (M) + (C) A ← (A) + (X) + (C)	↕	↕	IMM DIR IX DIR	A9 B9 B9 B9	ii dd 0E 0F	2 3 3 3
ADD #*opr8i* ADD *opr8a* ADD *opr4a* ADD ,X (1) ADD X	Add without Carry	A ← (A) + (M)	↕	↕	IMM DIR TNY IX DIR	AB BB 6*x* 6E 6F	ii dd	2 3 3 3 3
AND #*opr8i* AND *opr8a* AND ,X (1) AND X	Logical AND	A ← (A) & (M) A ← (A) & (X)	↕	—	IMM DIR IX DIR	A4 B4 B4 B4	ii dd 0E 0F	2 3 3 3
ASLA(1)	Arithmetic Shift Left		↕	↕	INH	48		1
BCC *rel*	Branch if Carry Bit Clear	PC ← (PC) + $0002 + *rel*, if (C) = 0	—	—	REL	34	rr	3
BCLR *n,opr8a* BCLR *n*,D[X] BCLR *n*,X	Clear Bit n in Memory	Mn ← 0	—	—	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7) IX (b0) IX (b1) IX (b2) IX (b3) IX (b4) IX (b5) IX (b6) IX (b7) DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F 11 13 15 17 19 1B 1D 1F 11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd 0E 0E 0E 0E 0E 0E 0E 0E 0F 0F 0F 0F 0F 0F 0F 0F	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + $0002 + *rel*, if (C) = 1	—	—	REL	35	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + $0002 + *rel*, if (Z) = 1	—	—	REL	37	rr	3
BGND	Background	Enter Background Debug Mode	—	—	INH	BF		5+

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 7-1. Instruction Set Summary (Sheet 2 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
BHS *rel* (1)	Branch if Higher or Same (Same as BCC)	PC ← (PC) + \$0002 + *rel*, if (C) = 0	—	—	REL	34	rr	3
BLO *rel* (1)	Branch if Lower (Same as BCS)	PC ← (PC) + \$0002 + *rel*, if (C) = 1	—	—	REL	35	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + \$0002 + *rel*, if (Z) = 0	—	—	REL	36	rr	3
BRA *rel*	Branch Always	PC ← (PC) + \$0002 + *rel*	—	—	REL	30	rr	3
BRN *rel* (1)	Branch Never	PC ← (PC) + \$0002	—	—	REL	30	00	3
BRCLR *n*,*opr8a*,*rel*	Branch if Bit *n* in Memory Clear	PC ← (PC) + \$0003 + *rel*, if (Mn) = 0	—	↕	DIR (b0)	01	dd rr	5
					DIR (b1)	03	dd rr	5
					DIR (b2)	05	dd rr	5
					DIR (b3)	07	dd rr	5
					DIR (b4)	09	dd rr	5
					DIR (b5)	0B	dd rr	5
					DIR (b6)	0D	dd rr	5
					DIR (b7)	0F	dd rr	5
BRCLR *n*,D[X],*rel*					IX (b0)	01	0E rr	5
					IX (b1)	03	0E rr	5
					IX (b2)	05	0E rr	5
					IX (b3)	07	0E rr	5
					IX (b4)	09	0E rr	5
					IX (b5)	0B	0E rr	5
					IX (b6)	0D	0E rr	5
					IX (b7)	0F	0E rr	5
BRCLR *n*,X,*rel*					DIR (b0)	01	0F rr	5
					DIR (b1)	03	0F rr	5
					DIR (b2)	05	0F rr	5
					DIR (b3)	07	0F rr	5
					DIR (b4)	09	0F rr	5
					DIR (b5)	0B	0F rr	5
					DIR (b6)	0D	0F rr	5
					DIR (b7)	0F	0F rr	5

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 7-1. Instruction Set Summary (Sheet 3 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
BRSET *n*,*opr8a*,*rel*	Branch if Bit *n* in Memory Set	PC ← (PC) + $0003 + *rel*, if (Mn) = 1	—	↕	DIR (b0)	00	dd rr	5
					DIR (b1)	02	dd rr	5
					DIR (b2)	04	dd rr	5
					DIR (b3)	06	dd rr	5
					DIR (b4)	08	dd rr	5
					DIR (b5)	0A	dd rr	5
					DIR (b6)	0C	dd rr	5
					DIR (b7)	0E	dd rr	5
BRSET *n*,D[X],*rel*					IX (b0)	00	0E rr	5
					IX (b1)	02	0E rr	5
					IX (b2)	04	0E rr	5
					IX (b3)	06	0E rr	5
					IX (b4)	08	0E rr	5
					IX (b5)	0A	0E rr	5
					IX (b6)	0C	0E rr	5
					IX (b7)	0E	0E rr	5
BRSET *n*,X,*rel*					DIR (b0)	00	0F rr	5
					DIR (b1)	02	0F rr	5
					DIR (b2)	04	0F rr	5
					DIR (b3)	06	0F rr	5
					DIR (b4)	08	0F rr	5
					DIR (b5)	0A	0F rr	5
					DIR (b6)	0C	0F rr	5
					DIR (b7)	0E	0F rr	5
BSET *n*,*opr8a*	Set Bit *n* in Memory	Mn ← 1	—	—	DIR (b0)	10	dd	5
					DIR (b1)	12	dd	5
					DIR (b2)	14	dd	5
					DIR (b3)	16	dd	5
					DIR (b4)	18	dd	5
					DIR (b5)	1A	dd	5
					DIR (b6)	1C	dd	5
					DIR (b7)	1E	dd	5
BSET *n*,D[X]					IX (b0)	10	0E	5
					IX (b1)	12	0E	5
					IX (b2)	14	0E	5
					IX (b3)	16	0E	5
					IX (b4)	18	0E	5
					IX (b5)	1A	0E	5
					IX (b6)	1C	0E	5
					IX (b7)	1E	0E	5
BSET *n*,X					DIR (b0)	10	0F	5
					DIR (b1)	12	0F	5
					DIR (b2)	14	0F	5
					DIR (b3)	16	0F	5
					DIR (b4)	18	0F	5
					DIR (b5)	1A	0F	5
					DIR (b6)	1C	0F	5
					DIR (b7)	1E	0F	5

1. This is a pseudo instruction supported by the normal RS08 instruction set.

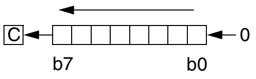
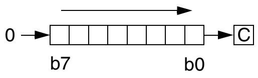
2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 7-1. Instruction Set Summary (Sheet 4 of 6)

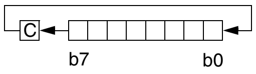
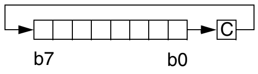
Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
BSR *rel*	Branch Subroutine	PC ← (PC) + 2 Push PC to shadow PC PC ← (PC) + *rel*	—	—	REL	AD	rr	3
CBEQA #*opr8i,rel* CBEQ *opr8a,rel* CBEQ ,X,*rel* (1),(2) CBEQ X,*rel* (1)	Compare and Branch if Equal	PC ← (PC) + $0003 + *rel*, if (A) – (M) = $00 PC ← (PC) + $0003 + *rel*, if (A) – (M) = $00 PC ← (PC) + $0003 + *rel*, if (A) – (X) = $00	—	—	IMM DIR IX DIR	41 31 31 31	ii rr dd rr 0E rr 0F rr	4 5 5 5
CLC	Clear Carry Bit	C ← 0	—	0	INH	38		1
CLR *opr8a* CLR *opr5a* CLR ,X (1) CLRA CLRX (1)	Clear	M ← $00 A ← $00 X ← $00	1	—	DIR SRT IX INH INH	3F 8*x* / 9*x* 8E 4F 8F	dd	3 2 2 1 2
CMP #*opr8i* CMP *opr8a* CMP ,X (1) CMP X (1)	Compare Accumulator with Memory	(A) – (M) (A) – (X)	↕	↕	IMM DIR IX INH	A1 B1 B1 B1	ii dd 0E 0F	2 3 3 3
COMA	Complement (One's Complement)	A ← $(\overline{A})$	↕	1	INH	43		1
DBNZ *opr8a,rel* DBNZ ,X,rel (1) DBNZA *rel* DBNZX rel (1)	Decrement and Branch if Not Zero	A ← (A) – $01 or M ← (M) - $01 PC ← (PC) + $0003 + *rel* if (result) ≠ 0 for DBNZ direct PC ← (PC) + $0002 + *rel* if (result) ≠ 0 for DBNZA X ← (X) – $01 PC ← (PC) + $0003 + *rel* if (result) ≠ 0	—	—	DIR IX INH INH	3B 3B 4B 3B	dd rr 0E rr rr 0F rr	7 7 4 7
DEC *opr8a* DEC *opr4a* DEC ,X (1) DECA DEC X	Decrement	M ← (M) – $01 A ← (A) – $01 X ← (X) – $01	↕	—	DIR TNY IX INH DIR	3A 5*x* 5E 4A 5F	dd	5 4 4 1 4
EOR #*opr8i* EOR *opr8a* EOR ,X (1) EOR X	Exclusive OR Memory with Accumulator	A ← (A ⊕ M) A ← (A ⊕ X)	↕	—	IMM DIR IX DIR	A8 B8 B8 B8	ii dd 0E 0F	2 3 3 3
INC *opr8a* INC *opr4a* INC ,X (1) INCA INCX (1)	Increment	M ← (M) + $01 A ← (A) + $01 X ← (X) + $01	↕	—	DIR TNY IX INH INH	3C 2*x* 2E 4C 2F	dd	5 4 4 1 4
JMP *opr16a*	Jump	PC ← Effective Address	—	—	EXT	BC	hh ll	4
JSR *opr16a*	Jump to Subroutine	PC ← (PC) + 3 Push PC to shadow PC PC ← Effective Address	—	—	EXT	BD	hh ll	4
LDA #*opr8i* LDA *opr8a* LDA *opr5a* LDA ,X (1)	Load Accumulator from Memory	A ← (M)	↕	—	IMM DIR SRT IX	A6 B6 C*x*/D*x* CE	ii dd	2 3 3 3

1. This is a pseudo instruction supported by the normal RS08 instruction set.
2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 7-1. Instruction Set Summary (Sheet 5 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
LDX #*opr8i* (1) LDX *opr8a* (1) LDX ,X (1)	Load Index Register from Memory	\$0F ← (M)	↕	—	IMD DIR IX	3E 4E 4E	ii 0F dd 0F 0E 0E	4 5 5
LSLA	Logical Shift Left		↕	↕	INH	48		1
LSRA	Logical Shift Right		↕	↕	INH	44		1

MOV *opr8a,opr8a*
MOV #*opr8i,opr8a*
MOV D[X],*opr8a*
MOV *opr8a*,D[X]
MOV #*opr8i*,D[X]
	Move	$(M)_{destination} \leftarrow (M)_{source}$	↕	—	DD
					IMD
					IX/DIR
					DIR/IX
					IMM/IX
						4E
						3E
						4E
						4E
						3E
							dd dd
							ii dd
							0E dd
							dd 0E
							ii 0E
								5
								4
								5
								5
								4

NOP	No Operation	None	—	—	INH	AC		1
ORA #*opr8i* ORA *opr8a* ORA ,X (1) ORA X	Inclusive OR Accumulator and Memory	A ← (A) | (M) A ← (A) | (X)	↕	—	IMM DIR IX DIR	AA BA BA BA	ii dd 0E 0F	2 3 3 3
ROLA	Rotate Left through Carry		↕	↕	INH	49		1
RORA	Rotate Right through Carry		↕	↕	INH	46		1
RTS	Return from Subroutine	Pull PC from shadow PC	—	—	INH	BE		3
SBC #*opr8i* SBC *opr8a* SBC ,X (1) SBC X	Subtract with Carry	A ← (A) – (M) – (C) A ← (A) – (X) – (C)	↕	↕	IMM DIR IX DIR	A2 B2 B2 B2	ii dd 0E 0F	2 3 3 3
SEC	Set Carry Bit	C ← 1	—	1	INH	39		1
SHA	Swap Shadow PC High with A	A ⇔ SPCH	—	—	INH	45		1
SLA	Swap Shadow PC Low with A	A ⇔ SPCL	—	—	INH	42		1
STA *opr8a* STA *opr5a* STA ,X (1) STA X	Store Accumulator in Memory	M ← (A)	↕	—	DIR SRT IX SRT	B7 Ex / Fx EE EF	dd	3 2 2 2
STX *opr8a* (1)	Store Index Register in Memory	M ← (X)	↕	—	DIR	4E	0F dd	5
STOP	Put MCU into stop mode		—	—	INH	AE		2+

1. This is a pseudo instruction supported by the normal RS08 instruction set.
2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 7-1. Instruction Set Summary (Sheet 6 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
SUB #*opr8i* SUB *opr8a* SUB *opr4a* SUB ,X (1) SUB X	Subtract	A ← (A) – (M) A ← (A) – (X)	↕	↕	IMM DIR TNY IX DIR	A0 B0 7*x* 7E 7F	ii dd	2 3 3 3 3
TAX(1)	Transfer A to X	X ← (A)	↕	—	INH	EF		2
TST opr8a (1) TSTA (1) TST ,X (1) TSTX (1)	Test for Zero	(M) – $00 (A) – $00 (X) – $00	↕	—	DD INH IX INH	4E AA 4E 4E	dd dd 00 0E 0E 0F 0F	5 2 5 5
TXA(1)	Transfer X to A	A ← (X)	↕	—	INH	CF		3
WAIT	Put MCU into WAIT mode		—	—	INH	AF		2+

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 7-2. Opcode Map

HIGH LOW	DIR	DIR	TNY	DIR/REL	INH	TNY	TNY	TNY	SRT	SRT	IMM/INH	DIR/EXT	SRT	SRT	SRT	SRT
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	5 BRSET0 3 DIR	5 BSET0 2 DIR	4 INC 1 TNY	3 BRA 2 REL	▨	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 SUB 2 IMM	3 SUB 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
1	5 BRCLR0 3 DIR	5 BCLR0 2 DIR	4 INC 1 TNY	5 CBEQ 3 DIR	4 CBEQA 3 IMM	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 CMP 2 IMM	3 CMP 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
2	5 BRSET1 3 DIR	5 BSET1 2 DIR	4 INC 1 TNY	▨	1 SLA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 SBC 2 IMM	3 SBC 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
3	5 BRCLR1 3 DIR	5 BCLR1 2 DIR	4 INC 1 TNY	▨	1 COMA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	▨	▨	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
4	5 BRSET2 3 DIR	5 BSET2 2 DIR	4 INC 1 TNY	3 BCC 2 REL	1 LSRA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 AND 2 IMM	3 AND 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
5	5 BRCLR2 3 DIR	5 BCLR2 2 DIR	4 INC 1 TNY	3 BCS 2 REL	1 SHA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	▨	▨	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
6	5 BRSET3 3 DIR	5 BSET3 2 DIR	4 INC 1 TNY	3 BNE 2 REL	1 RORA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 LDA 2 IMM	3 LDA 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
7	5 BRCLR3 3 DIR	5 BCLR3 2 DIR	4 INC 1 TNY	3 BEQ 2 REL	▨	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	▨	3 STA 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
8	5 BRSET4 3 DIR	5 BSET4 2 DIR	4 INC 1 TNY	1 CLC 1 INH	1 LSLA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 EOR 2 IMM	3 EOR 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
9	5 BRCLR4 3 DIR	5 BCLR4 2 DIR	4 INC 1 TNY	1 SEC 1 INH	1 ROLA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 ADC 2 IMM	3 ADC 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
A	5 BRSET5 3 DIR	5 BSET5 2 DIR	4 INC 1 TNY	5 DEC 2 DIR	1 DECA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 ORA 2 IMM	3 ORA 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
B	5 BRCLR5 3 DIR	5 BCLR5 2 DIR	4 INC 1 TNY	6 DBNZ 3 DIR	4 DBNZA 2 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 ADD 2 IMM	3 ADD 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
C	5 BRSET6 3 DIR	5 BSET6 2 DIR	4 INC 1 TNY	5 INC 2 DIR	1 INCA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	1 NOP 1 INH	4 JMP 3 EXT	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
D	5 BRCLR6 3 DIR	5 BCLR6 2 DIR	4 INC 1 TNY	▨	▨	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	3 BSR 2 REL	4 JSR 3 EXT	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
E	5 BRSET7 3 DIR	5 BSET7 2 DIR	4 INC 1 TNY	4 MOV 3 IMD	5 MOV 3 DD	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2+ STOP 1 INH	3 RTS 1 INH	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
F	5 BRCLR7 3 DIR	5 BCLR7 2 DIR	4 INC 1 TNY	3 CLR 2 DIR	1 CLRA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2+ WAIT 1 INH	5+ BGND 1 INH	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT

INH	**Inherent**	**REL**	**Relative**
IMM	**Immediate**	**SRT**	**Short**
DIR	**Direct**	**TNY**	**Tiny**
EXT	**Extended**		
DD	**Direct-Direct**	**IMD**	**Immediate-Direct**

▨	Gray box is decoded as illegal instruction

High Byte of Opcode in Hexadecimal **B**

Low Byte of Opcode in Hexadecimal	**0**	3 SUB 2 DIR

RS08 Cycles
Opcode Mnemonic
Number of Bytes / Addressing Mode

Chapter 8
Analog Comparator (RS08ACMPV1)

8.1 Introduction

The analog comparator module (ACMP) provides a circuit for comparing two analog input voltages or for comparing one analog input voltage with an internal bandgap reference voltage. The comparator circuit can operate across the full range of the supply voltage (rail to rail operation).

Figure 8-1 shows the SC9RS08MZ8 block diagram with the ACMP block and pins highlighted.

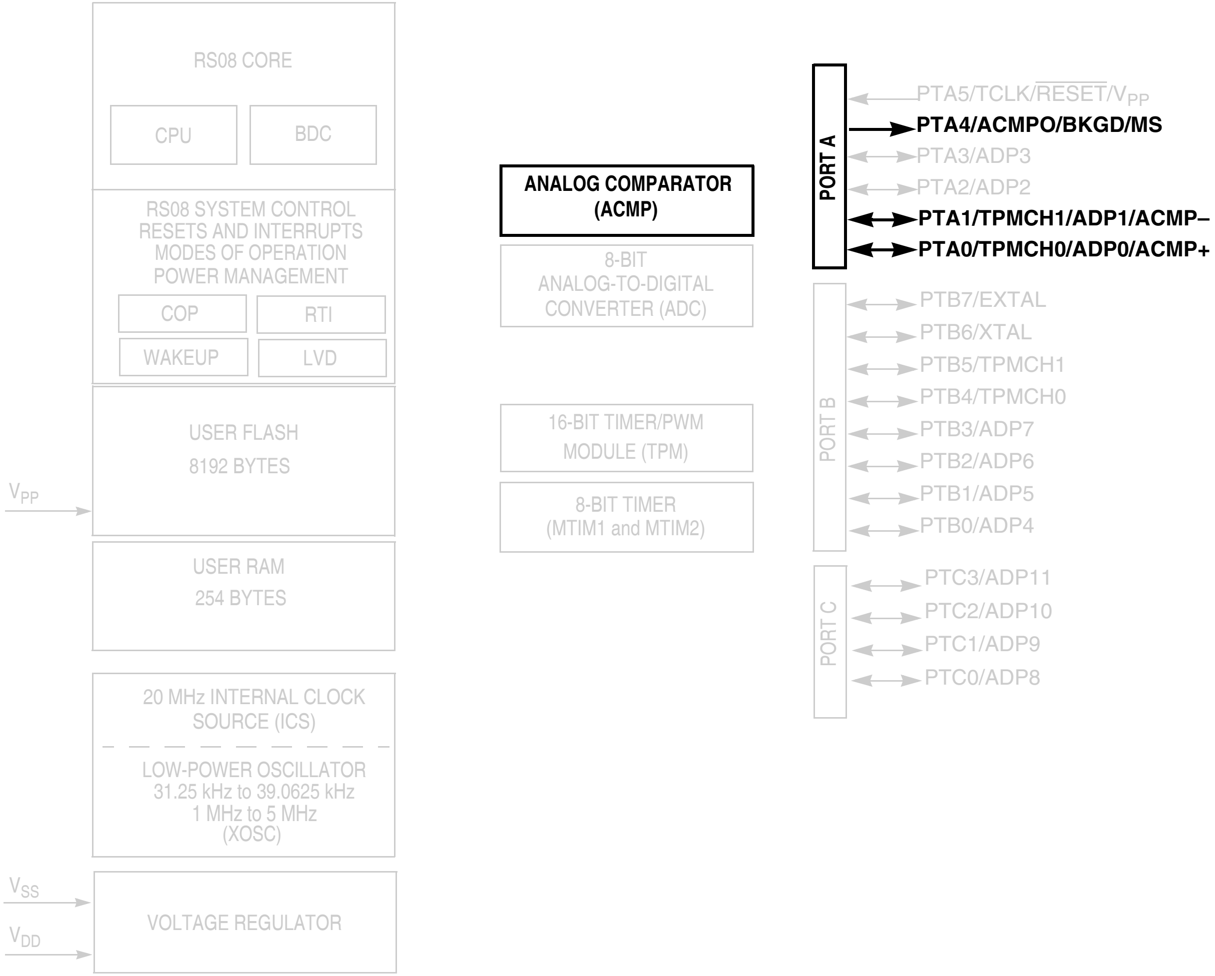


Figure 8-1. SC9RS08MZ8 Block Diagram Highlighting ACMP Block and Pins

8.1.1 Features

The ACMP has the following features:

- Full rail-to-rail supply operation
- Less than 40 mV of input offset
- Less than 15 mV of hysteresis
- Selectable interrupt on rising edge, falling edge, or either rising or falling edges of comparator output
- Option to compare to fixed internal bandgap reference voltage
- Option to allow comparator output to be visible on a pin, ACMPO
- Remains operational in stop mode

8.1.2 Modes of Operation

This section defines the ACMP operation in wait, stop, and background debug modes.

8.1.2.1 Operation in Wait Mode

The ACMP continues to operate in wait mode if enabled before executing the WAIT instruction. Therefore, the ACMP can be used to bring the MCU out of wait mode if the ACMP interrupt is enabled (ACIE = 1). For lowest possible current consumption, the ACMP must be disabled by software if not required as an interrupt source during wait mode.

8.1.2.2 Operation in Stop Mode

The ACMP continues to operate in stop mode if enabled and compare operation remains active. If ACOPE is enabled, comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. The MCU is brought out of stop when a compare event occurs and ACIE is enabled; ACF flag sets accordingly.

If stop is exited with a reset, the ACMP will be put into its reset state.

8.1.2.3 Operation in Active Background Mode

When the MCU is in active background mode, the ACMP will continue to operate normally.

8.1.3 Block Diagram

The block diagram for the analog comparator module is shown in Figure 8-2.

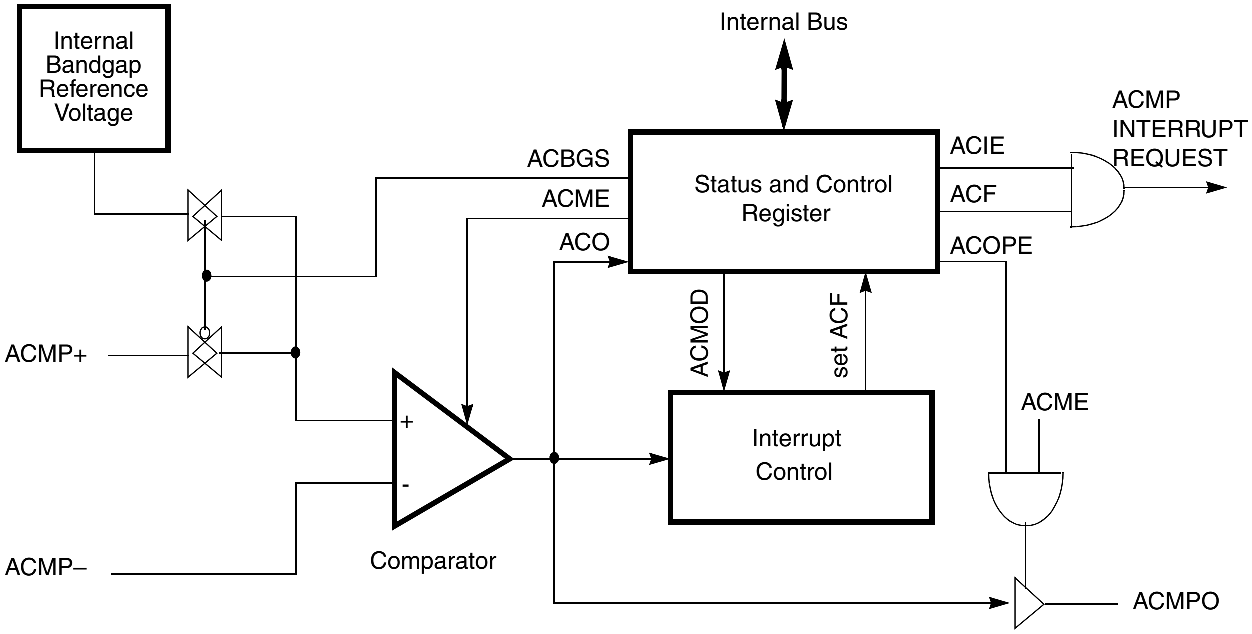


Figure 8-2. Analog Comparator (ACMP) Block Diagram

8.2 External Signal Description

The ACMP has two analog input pins, ACMP+ and ACMP–, and one digital output pin, ACMPO. Each of the input pins can accept an input voltage that varies across the full operating voltage range of the MCU. As shown in Figure 8-2, the ACMP– pin is connected to the inverting input of the comparator, and the ACMP+ pin is connected to the non-inverting input of the comparator if ACBGS=0. As shown in Figure 8-2, the ACMPO pin can be enabled to drive an external pin.

The signal properties of ACMP are shown in Table 8-1.

Table 8-1. Signal Properties

Signal	Function	I/O
ACMP–	Inverting analog input to the ACMP (Minus input)	I
ACMP+	Non-inverting analog input to the ACMP (Positive input)	I
ACMPO	Digital output of the ACMP	O

8.3 Register Definition

The ACMP includes one register:

- An 8-bit status and control register

Refer to the direct-page register summary in the memory chapter of this data sheet for the absolute address assignments for all ACMP registers.

8.3.1 ACMP Status and Control Register (ACMPSC)

ACMPSC contains the status flag and control bits which are used to enable and configure the ACMP.

	7	6	5	4	3	2	1	0
R	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD	
W					▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 8-3. ACMP Status and Control Register (ACMPSC)

Table 8-2. ACMPSC Field Descriptions

Field	Description
7 ACME	**Analog Comparator Module Enable** — ACME enables the ACMP module. 0 ACMP not enabled. 1 ACMP is enabled.
6 ACBGS	**Analog Comparator Bandgap Select** — ACBGS is used to select between the internal bandgap reference voltage or the ACMP+ pin as the non-inverting input of the analog comparator. 0 External pin ACMP+ selected as non-inverting input to comparator. 1 Internal bandgap reference voltage selected as non-inverting input to comparator.
5 ACF	**Analog Comparator Flag** — ACF is set when a compare event occurs. Compare events are defined by ACMOD. ACF is cleared by writing a one to ACF. 0 Compare event has not occurred. 1 Compare event has occurred.
4 ACIE	**Analog Comparator Interrupt Enable** — ACIE enables the interrupt for the ACMP. When ACIE is set, an interrupt will be asserted when ACF is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 ACO	**Analog Comparator Output** — Reading ACO will return the current value of the analog comparator output. ACO is reset to a 0 and will read as a 0 when the ACMP is disabled (ACME = 0).
2 ACOPE	**Analog Comparator Output Pin Enable** — ACOPE is used to enable the comparator output to be placed onto the external pin, ACMPO. ACOPE will only control the pin if the ACMP is active (ACME=1). 0 Analog comparator output not available on ACMPO. 1 Analog comparator output is driven out on ACMPO.
1:0 ACMOD	**Analog Comparator Mode** — ACMOD selects the type of compare event which sets ACF. 00 Encoding 0 — Comparator output falling edge. 01 Encoding 1 — Comparator output rising edge. 10 Encoding 2 — Comparator output falling edge. 11 Encoding 3 — Comparator output rising or falling edge.

8.4 Functional Description

The analog comparator can be used to compare two analog input voltages applied to ACMP+ and ACMP–; or it can be used to compare an analog input voltage applied to ACMP– with an internal bandgap reference voltage. ACBGS is used to select between the bandgap reference voltage or the ACMP+ pin as the input to the non-inverting input of the analog comparator.

The comparator output is high when the non-inverting input is greater than the inverting input, and it is low when the non-inverting input is less than the inverting input. ACMOD is used to select the condition which will cause ACF to be set. ACF can be set on a rising edge of the comparator output, a falling edge of the comparator output, or either a rising or a falling edge (toggle). The comparator output can be read directly through ACO. The comparator output can also be driven onto the ACMPO pin using ACOPE.

NOTE

Comparator inputs are high impedance analog pins which are sensitive to noise. Noisy V_{DD} and/or pin toggling adjacent to the analog inputs may cause the comparator offset/hysteresis performance to exceed the specified values. Maximum source impedance is restricted to the value specified in SC9RS08MZ8 *Data Sheet*. To achieve maximum performance device is recommended to enter WAIT/STOP mode for ACMP measurement and adjacent pin toggling must be avoided.

Chapter 9
8-Bit Analog-to-Digital Converter (RS08ADC10V1)

9.1 Introduction

The 8-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

> The ADC in SC9RS08MZ8 device only provides 8-bit operation. The 10-bit and 12-bit operations are not supported. This chapter reserves the description of 10-bit and 12-bit operations to be compatible with the data sheet of other Freescale 8-bit MCUs.

Figure 9-1 shows the SC9RS08MZ8 with the ADC module and pins highlighted.

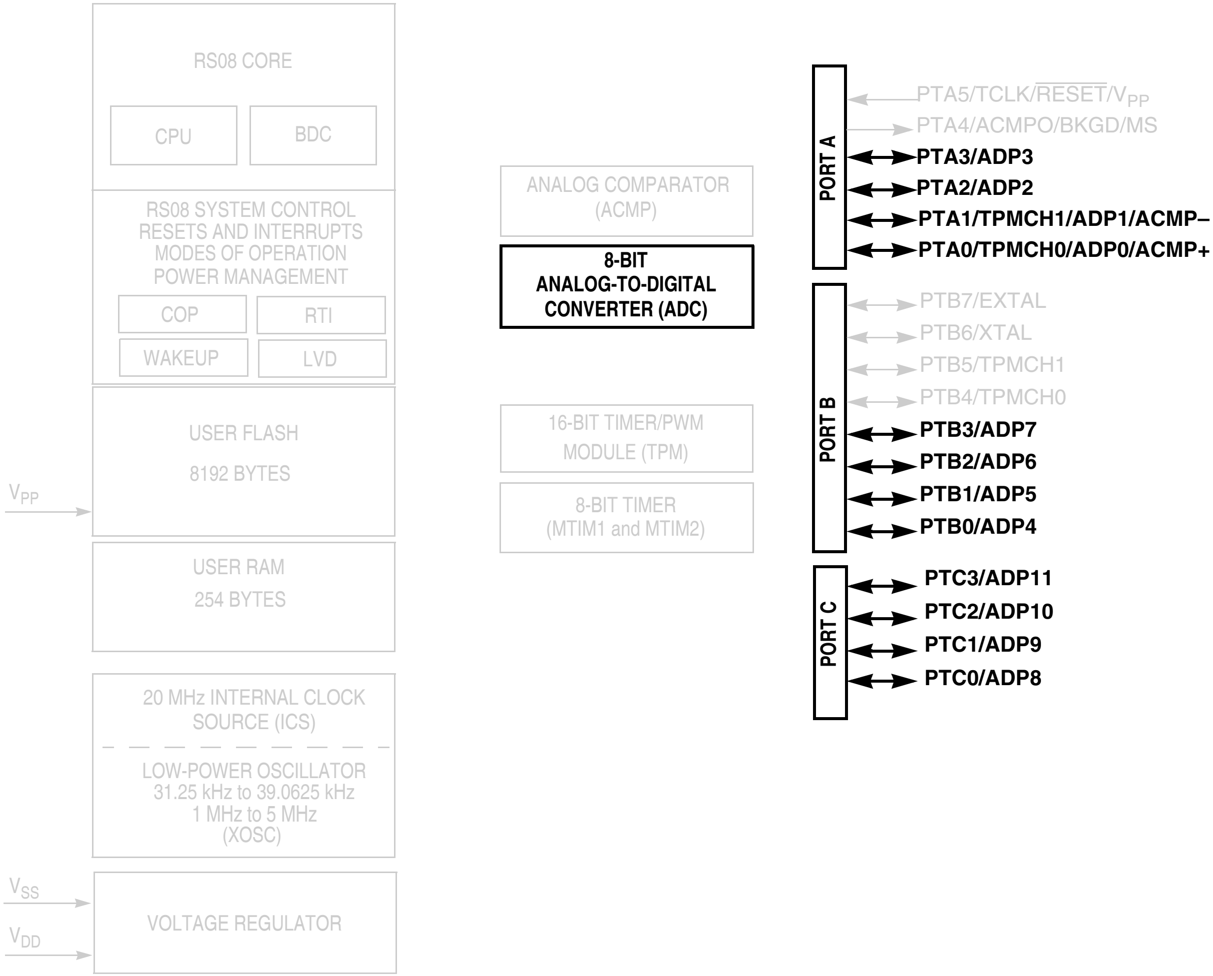


Figure 9-1. SC9RS08MZ8 Block Diagram Highlighting ADC Block and Pins

9.1.1 Module Configurations

This section provides device-specific information for configuring the ADC on SC9RS08MZ8 device.

9.1.1.1 Analog Supply and Voltage Reference Connections

The V_{DDAD} and V_{REFH} sources for the ADC are internally connected to the V_{DD} pin. The V_{SSAD} and V_{REFL} sources for the ADC are internally connected to the V_{SS} pin.

9.1.1.2 Alternate Clock

The ADC can perform conversions using the MCU bus clock, the bus clock divided by two, or the local asynchronous clock (ADACK) within the module. The alternate clock, ALTCLK, input for the SC9RS08MZ8 MCU device is not implemented.

9.1.1.3 Hardware Trigger

The ADC hardware trigger ADHWT is output from the real-time interrupt (RTI) counter. ICSERCLK or a nominal 1 kHz clock source within the RTI block can clock the RTI counter.

The input clock frequency and the RTIS bits determine the RTI period. The RTI counter is a free-running counter that generates an overflow at the RTI rate determined by the RTIS bits. When the ADC hardware trigger is enabled, a conversion initiates upon an RTI counter overflow.

The RTI can be configured to cause a hardware trigger in MCU run, and stop.

NOTE

To get quick RTI hardware trigger, the RTI clock source must be a high frequency external clock source.

9.1.1.4 Analog Pin Enables

The ADC on SC9RS08MZ8 device contains only two analog pin enable registers, APCTL1 and APCTL2.

The ADC channel assignments for the SC9RS08MZ8 device are shown in the table below. Reserved channels convert to an unknown value. Channels which are connected to an I/O pin have an associated pin control bit as shown.

Table 9-1. ADC Channel Assignment

ADCH	Channel	Input	Pin Control
00000	AD0	PTA0//ADP0	ADPC0
00001	AD1	PTA1/ADP1	ADPC1
00010	AD2	PTA2/ADP2	ADPC2
00011	AD3	PTA3/ADP3	ADPC3
00100	AD4	PTB0/ADP4	ADPC4
00101	AD5	PTB1/ADP5	ADPC5
00110	AD6	PTB2/ADP6	ADPC6
00111	AD7	PTB3/ADP7	ADPC7
01000	AD8	PTC0/ADP8	ADPC8
01001	AD9	PTC1/ADP9	ADPC9
01010	AD10	PTC2/ADP10	ADPC10
01011	AD11	PTC3/ADP11	ADPC11
01100 – 11100	Reserved	N/A	N/A
11101	V_{REFH}	V_{DD}	N/A
11110	V_{REFL}	V_{SS}	N/A
11111	module disabled	None	N/A

9.1.1.5 Low-Power Mode Operation

The ADC can run in stop mode but requires LVDSE and LVDE in SPMSC1 to be set.

9.1.2 Features

Features of the ADC module include:

- Linear successive approximation algorithm with 10 bits resolution.
- Up to 28 analog inputs.
- Output formatted in 10- or 8-bit right-justified format.
- Single or continuous conversion (automatic return to idle after single conversion).
- Configurable sample time and conversion speed/power.
- Conversion complete flag and interrupt.
- Input clock selectable from up to four sources.
- Operation in wait or stop modes for lower noise operation.
- Asynchronous clock source for lower noise operation.
- Selectable asynchronous hardware conversion trigger.
- Automatic compare with interrupt for less-than, or greater-than or equal-to, programmable value.

9.1.3 Block Diagram

Figure 9-2 provides a block diagram of the ADC module.

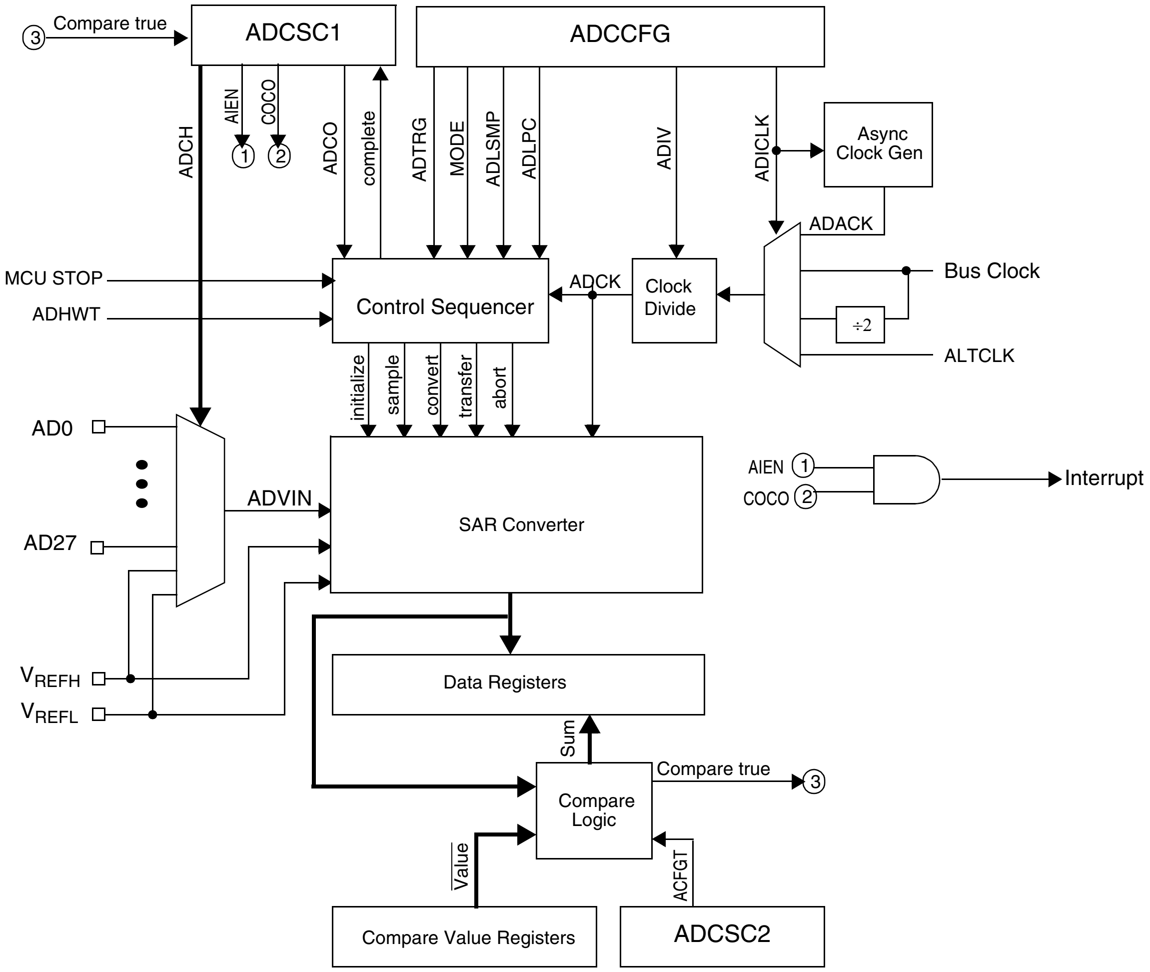


Figure 9-2. ADC Block Diagram

9.2 External Signal Description

The ADC module supports up to 28 separate analog inputs. It also requires four supply/reference/ground connections.

Table 9-2. Signal Properties

Name	Function
AD27–AD0	Analog Channel inputs
V_{REFH}	High reference voltage
V_{REFL}	Low reference voltage
V_{DDAD}	Analog power supply
V_{SSAD}	Analog ground

9.2.1 Analog Power (V_{DDAD})

The ADC analog portion uses V_{DDAD} as its power connection. In some packages, V_{DDAD} is connected internally to V_{DD}. If externally available, connect the V_{DDAD} pin to the same voltage potential as V_{DD}. External filtering might be necessary to ensure clean V_{DDAD} for good results.

9.2.2 Analog Ground (V_{SSAD})

The ADC analog portion uses V_{SSAD} as its ground connection. In some packages, V_{SSAD} is connected internally to V_{SS}. If externally available, connect the V_{SSAD} pin to the same voltage potential as V_{SS}.

9.2.3 Voltage Reference High (V_{REFH})

V_{REFH} is the high reference voltage for the converter. In some packages, V_{REFH} is connected internally to V_{DDAD}. If externally available, V_{REFH} can be connected to the same potential as V_{DDAD}, or can be driven by an external source that is between the minimum V_{DDAD} spec and the V_{DDAD} potential (V_{REFH} must never exceed V_{DDAD}).

9.2.4 Voltage Reference Low (V_{REFL})

V_{REFL} is the low reference voltage for the converter. In some packages, V_{REFL} is connected internally to V_{SSAD}. If externally available, connect the V_{REFL} pin to the same voltage potential as V_{SSAD}.

9.2.5 Analog Channel Inputs (ADx)

The ADC module supports up to 28 separate analog inputs. An input is selected for conversion through the ADCH channel select bits.

9.3 Register Definition

These memory mapped registers control and monitor operation of the ADC:

- Status and control register, ADCSC1and ADCSC2
- Data result registers, ADCRH and ADCRL
- Compare value registers, ADCCVH and ADCCVL
- Configuration register, ADCCFG
- Pin enable registers, APCTL1, APCTL2, APCTL3

9.3.1 Status and Control Register 1 (ADCSC1)

This section describes the function of the ADC status and control register 1 (ADCSC1). Writing ADCSC1 aborts the current conversion and initiates a new conversion (if the ADCH bits are equal to a value other than all 1s).

	7	6	5	4	3	2	1	0
R	COCO	AIEN	ADCO	ADCH				
W	▨							
Reset:	0	0	0	1	1	1	1	1

▨ = Unimplemented or Reserved

Figure 9-3. Status and Control Register (ADCSC1)

Table 9-3. ADCSC1 Register Field Descriptions

Field	Description
7 COCO	**Conversion Complete Flag** — The COCO flag is a read-only bit which is set each time a conversion is completed when the compare function is disabled (ACFE = 0). When the compare function is enabled (ACFE = 1) the COCO flag is set upon completion of a conversion only if the compare result is true. This bit is cleared whenever ADCSC1 is written or whenever ADCRL is read. 0 Conversion not completed 1 Conversion completed
6 AIEN	**Interrupt Enable** — AIEN enables conversion complete interrupts. When COCO becomes set while AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt disabled 1 Conversion complete interrupt enabled
5 ADCO	**Continuous Conversion Enable** — ADCO is used to enable continuous conversions. 0 One conversion following a triggered operation[1] 1 Continuous conversions initiated following a triggered operation is selected.
4:0 ADCH	**Input Channel Select** — The ADCH bits form a 5-bit field which selects one of the input channels. The input channels are detailed in Figure 9-4. The successive approximation converter subsystem is turned off when the channel select bits are all set to 1. This feature allows for explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional, single conversion from being performed. It is not necessary to set the channel select bits to all 1s to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes.

[1] See Table 9-4 for how to specify a hardware or software trigger type (ADTRG).

Figure 9-4. Input Channel Select

ADCH	Input Select	ADCH	Input Select
00000	AD0	10000	AD16
00001	AD1	10001	AD17
00010	AD2	10010	AD18
00011	AD3	10011	AD19
00100	AD4	10100	AD20
00101	AD5	10101	AD21
00110	AD6	10110	AD22
00111	AD7	10111	AD23

Figure 9-4. Input Channel Select (continued)

ADCH	Input Select	ADCH	Input Select
01000	AD8	11000	AD24
01001	AD9	11001	AD25
01010	AD10	11010	AD26
01011	AD11	11011	AD27
01100	AD12	11100	Reserved
01101	AD13	11101	V_{REFH}
01110	AD14	11110	V_{REFL}
01111	AD15	11111	Module disabled

9.3.2 Status and Control Register 2 (ADCSC2)

The ADCSC2 register is used to control the compare function, conversion trigger and conversion active of the ADC module.

	7	6	5	4	3	2	1	0
R	ADACT	ADTRG	ACFE	ACFGT	0	0	R[1]	R[1]
W	▨				▨	▨		
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

[1] Bits 1 and 0 are reserved bits that must always be written to 0.

Figure 9-5. Status and Control Register 2 (ADCSC2)

Table 9-4. ADCSC2 Register Field Descriptions

Field	Description
7 ADACT	**Conversion Active** — ADACT indicates that a conversion is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted. 0 Conversion not in progress 1 Conversion in progress
6 ADTRG	**Conversion Trigger Select** — ADTRG selects the type of trigger to be used for initiating a conversion. 0 Software trigger selected.(initiates a conversion following a write to ADCSC1). 1 Hardware trigger selected.(initiates a conversion following the assertion of the ADHWT input).
5 ACFE	**Compare Function Enable** — ACFE is used to enable the compare function. 0 Compare function disabled 1 Compare function enabled
4 ACFGT	**Compare Function Greater Than Enable** — ACFGT configures the compare function to trigger upon conversion of the input being monitored. 0 Compare triggered when input is less than compare level 1 Compare triggered when input is greater than or equal to compare level

9.3.3 Data Result High Register (ADCRH)

ADCRH contains the upper two bits of the result of a 10-bit conversion. When configured for 8-bit conversions both ADR8 and ADR9 are equal to zero. ADCRH is updated each time a conversion completes except when automatic compare is enabled and the compare condition is not met. In 10-bit MODE, reading ADCRH prevents the ADC from transferring subsequent conversion results into the result registers until ADCRL is read. If ADCRL is not read until after the next conversion is completed, then the intermediate conversion result will be lost. In 8-bit mode there is no interlocking with ADCRL. In the case that the MODE bits are changed, any data in ADCRH becomes invalid.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	ADR9	ADR8
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-6. Data Result High Register (ADCRH)

9.3.4 Data Result Low Register (ADCRL)

ADCRL contains the lower eight bits of the result of a 10-bit conversion, and all eight bits of an 8-bit conversion. This register updates each time a conversion completes except when an automatic compare is enabled and the compare condition is not met. In 10-bit mode, reading ADCRH prevents the ADC from transferring subsequent conversion results into the result registers until ADCRL is read. If ADCRL is not read until after the next conversion completes, then the intermediate conversion results are lost. In 8-bit mode, there is no interlocking with ADCRH. When MODE bits are changed, data in ADCRL becomes invalid.

	7	6	5	4	3	2	1	0
R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
W	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-7. Data Result Low Register (ADCRL)

9.3.5 Compare Value High Register (ADCCVH)

This register holds the upper two bits of the 10-bit compare value. These bits are compared to the upper two bits of the result following a conversion in 10-bit mode when the compare function is enabled.In 8-bit operation, ADCCVH is not used during compare.

	7	6	5	4	3	2	1	0
R	0	0	0	0	▨	▨	ADCV9	ADCV8
W	▨	▨	▨	▨				
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 9-8. Compare Value High Register (ADCCVH)

9.3.6 Compare Value Low Register (ADCCVL)

This register holds the lower 8 bits of the 10-bit compare value, or all 8 bits of the 8-bit compare value. Bits ADCV7:ADCV0 are compared to the lower 8 bits of the result following a conversion in 10-bit or 8-bit mode.

	7	6	5	4	3	2	1	0
R W	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
Reset:	0	0	0	0	0	0	0	0

Figure 9-9. Compare Value Low Register(ADCCVL)

9.3.7 Configuration Register (ADCCFG)

ADCCFG is used to select the mode of operation, clock source, clock divide, and configure for low power or long sample time.

	7	6	5	4	3	2	1	0
R W	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
Reset:	0	0	0	0	0	0	0	0

Figure 9-10. Configuration Register (ADCCFG)

Table 9-5. ADCCFG Register Field Descriptions

Field	Description
7 ADLPC	**Low Power Configuration** — ADLPC controls the speed and power configuration of the successive approximation converter. This optimize power consumption when higher sample rates are not required. 0 High speed configuration 1 Low power configuration: {FC31} power is reduced at the expense of maximum clock speed.
6:5 ADIV	**Clock Divide Select** — ADIV selects the divide ratio used by the ADC to generate the internal clock ADCK. Table 9-6 shows the available clock configurations.

Table 9-5. ADCCFG Register Field Descriptions (continued)

Field	Description
4 ADLSMP	**Long Sample Time Configuration** — ADLSMP selects between long and short sample periods. This adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. 0 Short sample time 1 Long sample time
3:2 MODE	**Conversion Mode Selection** — MODE bits are used to select between 10- or 8-bit operation. See Table 9-7.
1:0 ADICLK	**Input Clock Select** — ADICLK bits select the input clock source to generate the internal clock ADCK. See Table 9-8.

Table 9-6. Clock Divide Select

ADIV	Divide Ratio	Clock Rate
00	1	Input clock
01	2	Input clock ÷ 2
10	4	Input clock ÷ 4
11	8	Input clock ÷ 8

Table 9-7. Conversion Modes

MODE	Mode Description
00	8-bit conversion (N=8)
01	Reserved
10	10-bit conversion (N=10)
11	Reserved

Table 9-8. Input Clock Select

ADICLK	Selected Clock Source
00	Bus clock
01	Bus clock divided by 2
10	Alternate clock (ALTCLK)
11	Asynchronous clock (ADACK)

9.3.8 Pin Control 1 Register (APCTL1)

The pin control registers disable the I/O port control of MCU pins used as analog inputs. APCTL1 controls the pins associated with channels 0–7 of the ADC module.

	7	6	5	4	3	2	1	0
R W	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
Reset:	0	0	0	0	0	0	0	0

Figure 9-11. Pin Control 1 Register (APCTL1)

Table 9-9. APCTL1 Register Field Descriptions

Field	Description
7 ADPC7	**ADC Pin Control 7** — ADPC7 is used to control the pin associated with channel AD7. 0 AD7 pin I/O control enabled 1 AD7 pin I/O control disabled
6 ADPC6	**ADC Pin Control 6** — ADPC6 is used to control the pin associated with channel AD6. 0 AD6 pin I/O control enabled 1 AD6 pin I/O control disabled
5 ADPC5	**ADC Pin Control 5** — ADPC5 is used to control the pin associated with channel AD5. 0 AD5 pin I/O control enabled 1 AD5 pin I/O control disabled
4 ADPC4	**ADC Pin Control 4** — ADPC4 is used to control the pin associated with channel AD4. 0 AD4 pin I/O control enabled 1 AD4 pin I/O control disabled
3 ADPC3	**ADC Pin Control 3** — ADPC3 is used to control the pin associated with channel AD3. 0 AD3 pin I/O control enabled 1 AD3 pin I/O control disabled
2 ADPC2	**ADC Pin Control 2** — ADPC2 is used to control the pin associated with channel AD2. 0 AD2 pin I/O control enabled 1 AD2 pin I/O control disabled
1 ADPC1	**ADC Pin Control 1** — ADPC1 is used to control the pin associated with channel AD1. 0 AD1 pin I/O control enabled 1 AD1 pin I/O control disabled
0 ADPC0	**ADC Pin Control 0** — ADPC0 is used to control the pin associated with channel AD0. 0 AD0 pin I/O control enabled 1 AD0 pin I/O control disabled

9.3.9 Pin Control 2 Register (APCTL2)

APCTL2 is used to control channels 8–15 of the ADC module.

	7	6	5	4	3	2	1	0
R W	ADPC15	ADPC14	ADPC13	ADPC12	ADPC11	ADPC10	ADPC9	ADPC8
Reset:	0	0	0	0	0	0	0	0

Figure 9-12. Pin Control 2 Register (APCTL2)

Table 9-10. APCTL2 Register Field Descriptions

Field	Description
7 ADPC15	**ADC Pin Control 15** — ADPC15 is used to control the pin associated with channel AD15. 0 AD15 pin I/O control enabled 1 AD15 pin I/O control disabled
6 ADPC14	**ADC Pin Control 14** — ADPC14 is used to control the pin associated with channel AD14. 0 AD14 pin I/O control enabled 1 AD14 pin I/O control disabled
5 ADPC13	**ADC Pin Control 13** — ADPC13 is used to control the pin associated with channel AD13. 0 AD13 pin I/O control enabled 1 AD13 pin I/O control disabled
4 ADPC12	**ADC Pin Control 12** — ADPC12 is used to control the pin associated with channel AD12. 0 AD12 pin I/O control enabled 1 AD12 pin I/O control disabled
3 ADPC11	**ADC Pin Control 11** — ADPC11 is used to control the pin associated with channel AD11. 0 AD11 pin I/O control enabled 1 AD11 pin I/O control disabled
2 ADPC10	**ADC Pin Control 10** — ADPC10 is used to control the pin associated with channel AD10. 0 AD10 pin I/O control enabled 1 AD10 pin I/O control disabled
1 ADPC9	**ADC Pin Control 9** — ADPC9 is used to control the pin associated with channel AD9. 0 AD9 pin I/O control enabled 1 AD9 pin I/O control disabled
0 ADPC8	**ADC Pin Control 8** — ADPC8 is used to control the pin associated with channel AD8. 0 AD8 pin I/O control enabled 1 AD8 pin I/O control disabled

9.3.10 Pin Control 3 Register (APCTL3)

APCTL3 is used to control channels 16–23 of the ADC module.

	7	6	5	4	3	2	1	0
R W	ADPC23	ADPC22	ADPC21	ADPC20	ADPC19	ADPC18	ADPC17	ADPC16
Reset:	0	0	0	0	0	0	0	0

Figure 9-13. Pin Control 3 Register (APCTL3)

Table 9-11. APCTL3 Register Field Descriptions

Field	Description
7 ADPC23	**ADC Pin Control 23** — ADPC23 is used to control the pin associated with channel AD23. 0 AD23 pin I/O control enabled 1 AD23 pin I/O control disabled
6 ADPC22	**ADC Pin Control 22** — ADPC22 is used to control the pin associated with channel AD22. 0 AD22 pin I/O control enabled 1 AD22 pin I/O control disabled
5 ADPC21	**ADC Pin Control 21** — ADPC21 is used to control the pin associated with channel AD21. 0 AD21 pin I/O control enabled 1 AD21 pin I/O control disabled
4 ADPC20	**ADC Pin Control 20** — ADPC20 is used to control the pin associated with channel AD20. 0 AD20 pin I/O control enabled 1 AD20 pin I/O control disabled
3 ADPC19	**ADC Pin Control 19** — ADPC19 is used to control the pin associated with channel AD19. 0 AD19 pin I/O control enabled 1 AD19 pin I/O control disabled
2 ADPC18	**ADC Pin Control 18** — ADPC18 is used to control the pin associated with channel AD18. 0 AD18 pin I/O control enabled 1 AD18 pin I/O control disabled
1 ADPC17	**ADC Pin Control 17** — ADPC17 is used to control the pin associated with channel AD17. 0 AD17 pin I/O control enabled 1 AD17 pin I/O control disabled
0 ADPC16	**ADC Pin Control 16** — ADPC16 is used to control the pin associated with channel AD16. 0 AD16 pin I/O control enabled 1 AD16 pin I/O control disabled

9.4 Functional Description

The ADC module is disabled during reset or when the ADCH bits are all high. The module is idle when a conversion has completed and another conversion has not been initiated. When idle, the module is in its lowest power state.

The ADC can perform an analog-to-digital conversion on any of the software selectable channels. The selected channel voltage is converted by a successive approximation algorithm into an 11-bit digital result. In 8-bit mode, the selected channel voltage is converted into a 9-bit digital result by a successive approximation algorithm.

When the conversion completes, the result is placed in the data registers (ADCRH and ADCRL).In 10-bit mode, the result is rounded to 10 bits and placed in ADCRH and ADCRL. In 8-bit mode, the result is rounded to 8 bits and placed in ADCRL. The conversion complete flag (COCO) is then set, and an interrupt is generated if the conversion complete interrupt has been enabled (AIEN = 1).

The ADC module can automatically compare a conversion result with the contents of its compare registers. Set the ACFE bit to enable the compare function and operate in conjunction with any of the conversion modes and configurations.

9.4.1 Clock Select and Divide Control

One of four clock sources can be selected as the clock source for the ADC module. This clock source is then divided by a configurable value to generate the input clock to the converter (ADCK). The clock is selected from one of the following sources by means of the ADICLK bits.

- The bus clock, equal to the frequency at which the software is executed. This is the default selection following reset.
- The bus clock divided by 2. For higher bus clock rates, this allows a maximum divide by 16 of the bus clock.
- ALTCLK, as defined for this MCU (See module section introduction).
- The asynchronous clock (ADACK) – This clock is generated from a clock source within the ADC module. When selected as the clock source this clock remains active while the MCU is in wait or stop mode and allows conversions in these modes for lower noise operation.

The selected clock's frequency must fall within the range specified for ADCK. If the available clocks are too slow, the ADC will not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by the ADIV bits and can be divided by 1, 2, 4, or 8.

9.4.2 Input Select and Pin Control

The pin control registers (APCTL3, APCTL2, and APCTL1) disable the I/O port control of the pins used as analog inputs.When a pin control register bit is set, the following conditions are forced for the associated MCU pin:

- The output buffer is forced to its high impedance state.
- The input buffer is disabled. A read of the I/O port returns a zero for any pin with its input buffer disabled.
- The pullup is disabled.

9.4.3 Hardware Trigger

The ADC module enables ADHWT, a selectable asynchronous hardware conversion trigger, when the ADTRG bit is set. This source is not available on all MCUs. Consult the module introduction for information on the ADHWT source specific to this MCU.

When ADHWT source is available and hardware trigger is enabled (ADTRG=1), a conversion initiates on the rising edge of ADHWT. If a conversion is in progress when a rising edge occurs, the rising edge is ignored. In continuous convert configuration, only the initial rising edge to launch continuous conversions is observed. The hardware trigger function operates with any of the conversion modes and configurations.

9.4.4 Conversion Control

Conversions can be performed in 10-bit mode or 8-bit mode as determined by the MODE bits. Conversions can be initiated by either a software or hardware trigger. In addition, the ADC module can be configured for low power operation, long sample time, continuous conversion, and automatic compare of the conversion result to a software determined compare value.

9.4.4.1 Initiating Conversions

A conversion is initiated:

- A write to ADCSC1 (with ADCH bits not all 1s) if software triggered operation is selected.
- A hardware trigger (ADHWT) event if hardware triggered operation is selected.
- The transfer of the result to the data registers when continuous conversion is enabled.

If continuous conversions are enabled a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation, continuous conversions begin after ADCSC1 is written and continue until aborted. In hardware triggered operation, continuous conversions begin after a hardware trigger event and continue until aborted.

9.4.4.2 Completing Conversions

A conversion completes when the result transferred into the data-result registers, ADCRH and ADCRL. The setting of COCO. An interrupt is generated if AIEN is high when COCO is set.

A blocking mechanism prevents a new result from overwriting previous data in ADCRH and ADCRL if the previous data is in the process of being read while in 10-bit MODE (the ADCRH register has been read; the ADCRL register has not). When blocking is active, the data transfer is blocked, COCO is not set, and the new result is lost. In the case of single conversions with the compare function enabled and the compare condition false, blocking has no effect and ADC operation terminates. In all other cases, when a data transfer is blocked, another conversion initiates regardless of the state of ADCO (single or continuous conversions enabled).

If single conversions are enabled, the blocking mechanism could result in several discarded conversions and excess power consumption. To avoid this, the data registers must not be read after initiating a single conversion until the conversion completes.

9.4.4.3 Aborting Conversions

Any conversion in progress aborts when:

- A write to ADCSC1 occurs (the current conversion is aborted and a new conversion is initiated, if ADCH are not all 1s).
- A write to ADCSC2, ADCCFG, ADCCVH, or ADCCVL occurs. This indicates a mode of operation change has occurred and the current conversion is therefore invalid.
- The MCU is reset.
- The MCU enters stop mode with ADACK not enabled.

When a conversion is aborted, the contents of ADCRH and ADCRL, data registers is not altered but continues to be the values transferred after the completion of the last successful conversion. If a reset aborts the conversion, ADCRH and ADCRL return to their reset states.

9.4.4.4 Power Control

The ADC module remains in idle until a conversion is initiated. If ADACK is selected as the conversion clock source, the ADACK clock generator is also enabled.

Setting ADLPC can reduce power consumption. resulting in a lower maximum value for f_{ADCK} (see the electrical specifications).

9.4.4.5 Total Conversion Time

The total conversion time depends on the sample time (as determined by ADLSMP), the MCU bus frequency, the conversion mode (8-bit or 10-bit), and the frequency of the conversion clock (f_{ADCK}). After the module becomes active, sampling of the input begins. ADLSMP is used to select between short and long sample times.When sampling completes, the converter is isolated from the input channel and a successive approximation algorithm is performed to determine the digital value of the analog signal. The conversion result transfers to ADCRH and ADCRL upon completion of the conversion algorithm.

- If the bus frequency is less than the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled (ADLSMP=0).
- If the bus frequency is less than 1/11th of the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when long sample is enabled (ADLSMP=1).

The maximum total conversion time for different conditions is summarized in Table 9-12.

Table 9-12. Total Conversion Time vs. Control Conditions

Conversion Type	ADICLK	ADLSMP	Max Total Conversion Time
Single or first continuous 8-bit	0x, 10	0	20 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit	0x, 10	0	23 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	0x, 10	1	40 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit	0x, 10	1	43 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	11	0	5 μs + 20 ADCK + 5 bus clock cycles
Single or first continuous 10-bit	11	0	5 μs + 23 ADCK + 5 bus clock cycles
Single or first continuous 8-bit	11	1	5 μs + 40 ADCK + 5 bus clock cycles
Single or first continuous 10-bit	11	1	5 μs + 43 ADCK + 5 bus clock cycles
Subsequent continuous 8-bit; $f_{BUS} \geq f_{ADCK}$	xx	0	17 ADCK cycles
Subsequent continuous 10-bit; $f_{BUS} \geq f_{ADCK}$	xx	0	20 ADCK cycles
Subsequent continuous 8-bit; $f_{BUS} \geq f_{ADCK}/11$	xx	1	37 ADCK cycles
Subsequent continuous 10-bit; $f_{BUS} \geq f_{ADCK}/11$	xx	1	40 ADCK cycles

The chosen clock source and the divide ratio determine the maximum total conversion time. The clock source is selectable by the ADICLK bits, and the divide ratio is specified by the ADIV bits.

For example, you could use the following equation conversion time for a single conversion in 10-bit mode:

$$\text{Conversion time} = \frac{\text{23 ADCK cyc}}{\text{8 MHz/1}} + \frac{\text{5 bus cyc}}{\text{8 MHz}} = 3.5\ \mu\text{s}$$

Number of bus cycles = 3.5 μs x 8 MHz = 28 cycles

where:

- **Mode**: 10-Bit
- **Input Clock Source:** Bus Clock
- **Input Clock Ratio:** Divide by 1
- **Bus Frequency:** 8 MHz
- **Conversion Time:** 3.5 μs

NOTE

The ADCK frequency must be between f_{ADCK} minimum and f_{ADCK} maximum to meet ADC specifications.

9.4.5 Automatic Compare Function

The compare function can be configured to check for an upper or lower limit. After the input is sampled and converted, the result is added to the two's complement of the compare value (ADCCVH and ADCCVL). When comparing to an upper limit (ACFGT = 1), if the result is greater-than or equal-to the compare value, COCO is set. When comparing to a lower limit (ACFGT = 0), if the result is less than the compare value, COCO is set. The value generated by the addition of the conversion result and the two's complement of the compare value is transferred to ADCRH and ADCRL.

Following a conversion where the compare function is enabled, and the compare condition is not true, COCO is not set and no data is transferred to the result registers. An ADC interrupt is generated upon the setting of COCO if the ADC interrupt is enabled (AIEN = 1).

NOTE

The compare function can be used to monitor the voltage on a channel while the MCU is in wait or stop mode. The ADC interrupt wakes the MCU when the compare condition is met.

9.4.6 MCU Wait Mode Operation

The WAIT instruction puts the MCU in a lower power-consumption standby mode. Recovery is very fast because the clock sources remain active. If a conversion is in progress when the MCU enters wait mode, it continues until completion. Conversions can be initiated while the MCU is in wait mode by means of the hardware trigger or if continuous conversions are enabled.

While in wait mode, the bus clock, bus clock divided by two, and ADACK are available as conversion clock sources. Using ALTCLK as the conversion clock source in wait is dependent on the definition of ALTCLK for this MCU. Consult the module introduction for information on ALTCLK specific to this MCU.

A conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from wait mode if the ADC interrupt is enabled (AIEN = 1).

9.4.7 MCU Stop Mode Operation

The STOP instruction puts the MCU in a low power-consumption standby mode during which most or all clock sources on the MCU are disabled.

9.4.7.1 Stop Mode With ADACK Disabled

If the asynchronous clock, ADACK, is not the conversion clock, executing a STOP instruction aborts the current conversion and places the ADC in its idle state. The contents of ADCRH and ADCRL are unaffected by stop mode.After exiting from stop mode, a software or hardware trigger is required to resume conversions.

9.4.7.2 Stop Mode With ADACK Enabled

If ADACK is the conversion clock, the ADC continues operation during stop mode. For guaranteed ADC operation, the MCU's voltage regulator must remain active during stop mode. Consult the module introduction for configuration information for this MCU.

If a conversion is in progress when the MCU enters stop mode, it continues until completion. Conversions can be initiated while the MCU is in stop mode by means of the hardware trigger or if continuous conversions are enabled.

A conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from stop mode if the ADC interrupt is enabled (AIEN = 1).

NOTE

The ADC module to wake the system from low power stop and cause the MCU to begin consuming run-level currents without generating a system level interrupt. To prevent this, your application must ensure that the data transfer blocking mechanism (discussed in Section 9.4.4.2, "Completing Conversions.") is cleared when entering stop and continuing ADC conversions.

9.5 Initialization Information

This section gives basic direction on how to initialize and configure the ADC module. Among other options, you can configure the module for:

- 8-bit or 10-bit resolution
- single or continuous conversion

- polled or interrupt approach

Refer to Table 9-6, Table 9-7, andTable 9-8 for information used in the following example.

NOTE

Hexadecimal values designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

9.5.1 ADC Module Initialization Example

Before the ADC module completes conversions, an initialization must be performed.

9.5.1.1 Initialization Sequence

A typical initialization sequence is as follows:

1. Update the configuration register (ADCCFG) to select the input clock source and the divide ratio used to generate the internal clock, ADCK. Also used for selecting sample time and low-power configuration.
2. Update status and control register 2 (ADCSC2) to select the conversion trigger (hardware or software) and compare function options, if enabled.
3. Update status and control register 1 (ADCSC1) to select continuous or once-only conversions and to enable or disable conversion complete interrupts. Also, use this register to select the input channel on which conversions are to be performed.

9.5.1.2 Pseudo-Code Example

In this example, the ADC module is configured with interrupts enabled to perform a single 10-bit conversion at low power with a long sample time on input channel 1, where the internal ADCK clock is derived from the bus clock divided by 1.

ADCCFG = 0x98 (%10011000)

Bit 7	ADLPC	1	Configures for low power (lowers maximum clock speed)
Bit 6:5	ADIV	00	Sets ADCK to input clock ÷ 1
Bit 4	ADLSMP	1	Configures for long sample time
Bit 3:2	MODE	10	Sets mode at 10-bit conversions
Bit 1:0	ADICLK	00	Selects bus clock as input clock source

ADCSC2 = 0x00 (%00000000)

Bit 7	ADACT	0	Indicates if a conversion is in progress
Bit 6	ADTRG	0	Selects software trigger
Bit 5	ACFE	0	Disables compare function
Bit 4	ACFGT	0	Not used in this example
Bit 3:2		00	Unimplemented or reserved, always reads zero
Bit 1:0		00	Reserved for Freescale internal use; always write zero

ADCSC1 = 0x41 (%01000001)

Bit 7	COCO	0	Indicates when a conversion completes
Bit 6	AIEN	1	Enable conversion complete interrupt
Bit 5	ADCO	0	Specifies one conversion only (continuous conversions disabled)
Bit 4:0	ADCH	00001	Selects input channel 1 selected as ADC input channel

ADCRH/L = 0xxx

Holds results of conversion. Read high byte (ADCRH) before low byte (ADCRL) so that conversion data cannot be overwritten with data from the next conversion.

ADCCVH/L = 0xxx

Holds compare value when compare function enabled

APCTL1=0x02

AD1 pin I/O control disabled. All other AD pins remain general purpose I/O pins

APCTL2=0x00

All other AD pins remain general purpose I/O pins

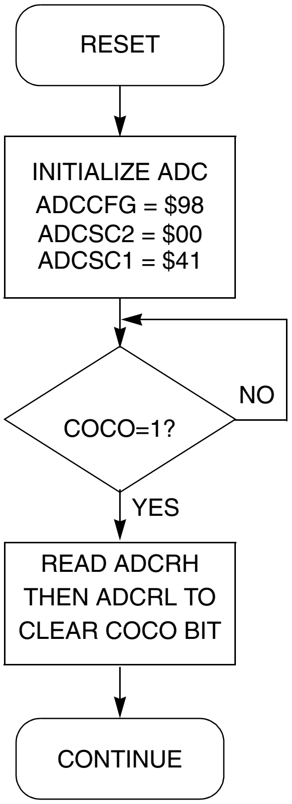


Figure 9-14. Initialization Flowchart Example

9.6 Application Information

This section contains information for using the ADC module in applications. The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an A/D converter.

9.6.1 External Pins and Routing

The following sections discuss the external pins associated with the ADC module and how to use them for best results.

9.6.1.1 Analog Supply Pins

The ADC module has analog power and ground supplies (V_{DDAD} and V_{SSAD}) available as separate pins on some devices. On other devices, V_{SSAD} is shared on the same pin as the MCU digital V_{SS}; on others, both V_{SSAD} and V_{DDAD} are shared with the MCU digital supply pins. In these cases, separate pads for the analog supplies are bonded to the same pin as the corresponding digital supply so some degree of isolation between the supplies is maintained.

When available on a separate pin, V_{DDAD} and V_{SSAD} must be connected to the same voltage potential as their corresponding MCU digital supply (V_{DD} and V_{SS}) and must be routed carefully for maximum noise immunity with bypass capacitors placed as near as possible to the package.

In cases where separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the V_{SSAD} pin. Typically this must be the only ground connection between these supplies if possible. The V_{SSAD} pin makes a good, single-point ground location.

9.6.1.2 Analog Reference Pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs. The high reference is V_{REFH}, which can be shared on the same pin as V_{DDAD} on some devices. The low reference is V_{REFL}, which can be shared on the same pin as V_{SSAD} on some devices.

When available on a separate pin, V_{REFH} can be connected to the same potential as V_{DDAD}, or can be driven by an external source that is between the minimum V_{DDAD} spec and the V_{DDAD} potential (V_{REFH} must never exceed V_{DDAD}). When available on a separate pin, V_{REFL} must be connected to the same voltage potential as V_{SSAD}. Both V_{REFH} and V_{REFL} must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the V_{REFH} and V_{REFL} loop. The best external component to meet this current demand is a 0.1 μF capacitor with good high frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current will cause a voltage drop which could result in conversion errors. Inductance in this path must be minimum (parasitic only).

9.6.1.3 Analog Input Pins

The external analog inputs are typically shared with digital I/O pins on MCU devices. The pin I/O control is disabled by setting the appropriate control bit in one of the pin control registers. Conversions can be performed on inputs without the associated pin control register bit set. It is recommended that the pin control register bit always be set when using a pin as an analog input. This avoids problems with contention because the output buffer will be in its high impedance state and the pullup is disabled. Also, the input buffer draws dc current when its input is not at either V_{DD} or V_{SS}. Setting the pin control register bits for all pins used as analog inputs must be done to achieve lowest operating current.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01 μF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used they must be placed as near as possible to the package pins and be referenced to V_{SSA}.

For proper conversion, the input voltage must fall between V_{REFH} and V_{REFL}. If the input is equal to or exceeds V_{REFH}, the converter circuit converts the signal to $3FF (full scale 10-bit representation) or $FF (full scale 8-bit representation). If the input is equal to or less than V_{REFL}, the converter circuit converts it to $000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There will be a brief current associated with V_{REFL} when the sampling capacitor is charging. The input is sampled for 3.5 cycles of the ADCK source when ADLSMP is low, or 23.5 cycles when ADLSMP is high.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins must not be transitioning during conversions.

9.6.2 Sources of Error

Several sources of error exist for A/D conversions. These are discussed in the following sections.

9.6.2.1 Sampling Error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy. Given the maximum input resistance of approximately 7kΩ and input capacitance of approximately 5.5 pF, sampling to within 1/4LSB (at 10-bit resolution) can be achieved within the minimum sample window (3.5 cycles at 8 MHz maximum ADCK frequency) provided the resistance of the external analog source (R_{AS}) is below 5 kΩ.

Higher source resistances or higher-accuracy sampling is possible by setting ADLSMP (to increase the sample window to 23.5 cycles) or decreasing ADCK frequency to increase sample time.

9.6.2.2 Pin Leakage Error

Leakage on the I/O pins can cause a conversion error if the external analog source resistance (R_{AS}) is high. If this error cannot be tolerated by the application, keep R_{AS} lower than $V_{DDAD} / (2^N \times I_{LEAK})$ for less than 1/4LSB leakage error (N = 8 in 8-bit mode or 10 in 10-bit mode).

9.6.2.3 Noise-Induced Errors

System noise during the sample or conversion process can affect the conversion accuracy. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 μF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 μF low-ESR capacitor from V_{DDAD} to V_{SSAD}.
- If inductive isolation is used from the primary supply, an additional 1 μF capacitor is placed from V_{DDAD} to V_{SSAD}.

- V_{SSAD} (and V_{REFL}, if connected) is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in wait or stop mode before initiating (hardware triggered conversions) or immediately after initiating (hardware or software triggered conversions) the ADC conversion.
  - For software triggered conversions, immediately follow the write to the ADCSC1 with a WAIT instruction or STOP instruction.
  - For stop mode operation, select ADACK as the clock source. Operation in stop reduces V_{DD} noise but increases effective conversion time due to stop recovery.
- There is no I/O switching, input or output, on the MCU during the conversion.

In some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in wait or stop or I/O activity can not be halted, try the following recommended actions to reduce the effect of noise on the accuracy:

- Place a 0.01 μF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSAD} (this improves noise issues, but can affect sample rate based upon the external analog source resistance).
- 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:

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

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, the code width of the first ($000) conversion is only 1/2LSB and the code width of the last ($FF or $3FF) is 1.5LSB.

9.6.2.5 Linearity Errors

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

- Zero-scale error (E_{ZS}) (sometimes called offset) — The difference between the actual code width of the first conversion and the ideal code width (1/2LSB). Note, if the first conversion is $001, then the difference between the actual $001 code width and its ideal (1LSB) is used.
- Full-scale error (E_{FS}) — The difference between the actual code width of the last conversion and the ideal code width (1.5LSB). Note, if the last conversion is $3FE, then the difference between the actual $3FE code width and its ideal (1LSB) is used.

- Differential non-linearity (DNL) — The worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL) — The highest-value (the absolute value) of the running sum 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) — The difference between the actual transfer function and the ideal straight-line transfer function, and therefore 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;

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 converter yields the lower code (and vice-versa). However, even very small amounts of system noise can cause the converter to be indeterminate (between two codes) for a range of input voltages around the transition voltage. This range is normally around ±1/2 LSB and increases with noise. Repeatedly sampling the input and averaging the result. Additionally the techniques discussed in Section 9.6.2.3 reduce this error.

Non-monotonicity —is defined as when, except for code jitter, the converter converts to a lower code for a higher input voltage.

Missing codes —are values never converted for any input value.

In 8-bit or 10-bit mode, the ADC is monotonic and has no missing codes.

Chapter 10
Internal Clock Source (RS08ICSOSCV1)

10.1 Introduction

The internal clock source (ICS) module provides clock source choices for the MCU. The module contains a frequency-locked loop (FLL) as a clock source controllable by an internal reference clock or an external reference clock. The module can provide FLL clock, internal reference clock, or external reference clock as a source for the MCU system clock, ICSOUT.

Whichever clock source is chosen, ICSOUT is passed through a bus clock divider (BDIV), which allows a lower final output clock frequency to be derived. ICSOUT is two times the bus frequency.

Figure 10-1 shows the SC9RS08MZ8 block diagram with the ICS module and pins highlighted.

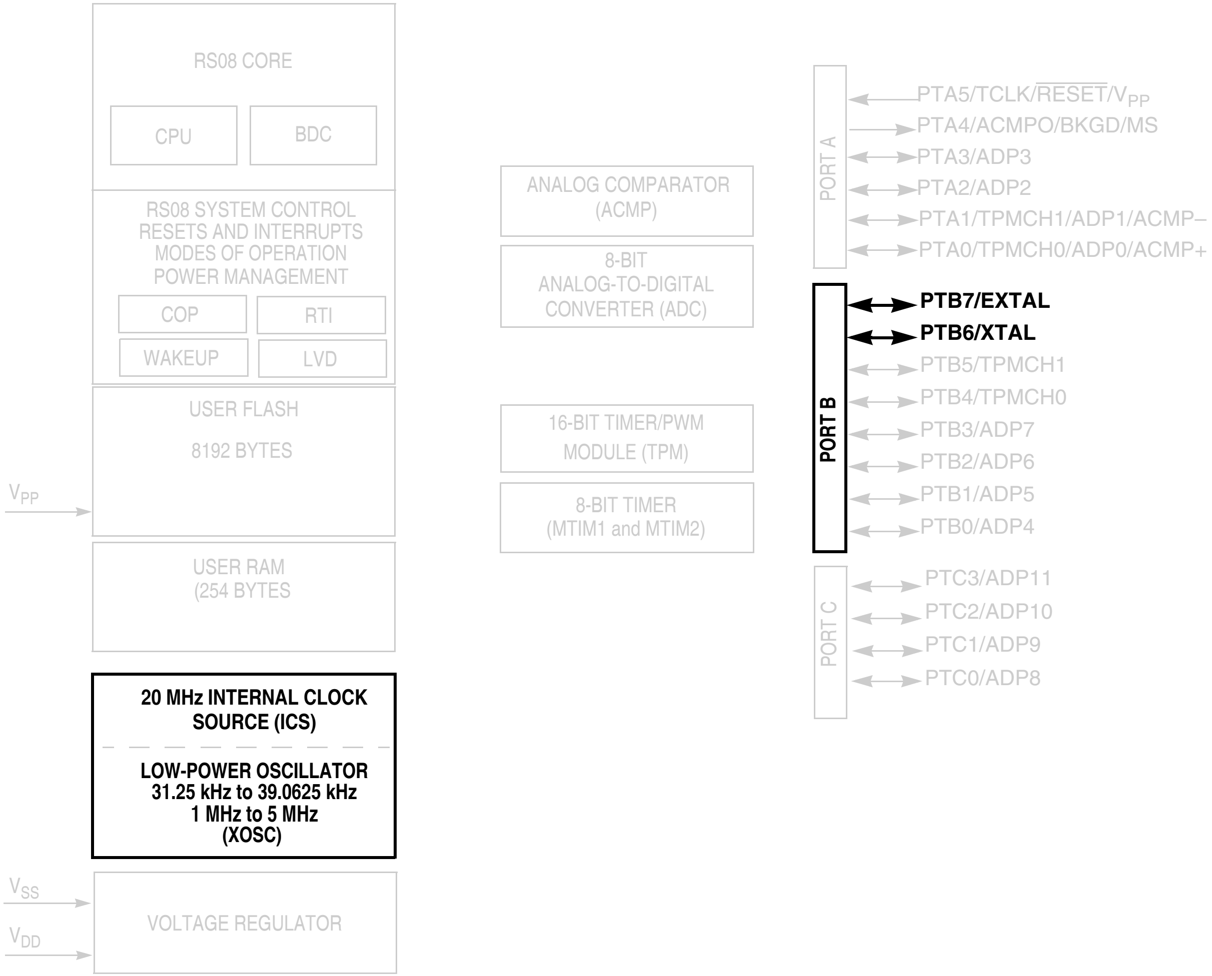


Figure 10-1. SC9RS08MZ8 Block Diagram Highlighting ICS Block and Pins

10.1.1 Features

Key features of the ICS module are:

- Frequency-locked loop (FLL) is trimmable for accuracy
  - 0.2% resolution using internal 32 kHz reference
  - 2% deviation over voltage and temperature using internal 32 kHz reference
- Internal or external reference clocks up to 5 MHz can be used to control the FLL
  - 3 bit select for reference divider is provided
- Internal reference clock has 9 trim bits available
- Internal or external reference clocks can be selected as the clock source for the MCU
- Whichever clock is selected as the source can be divided down
  - 2 bit select for clock divider is provided
    - Allowable dividers are: 1, 2, 4, 8
- Control signals for a low power oscillator as the external reference clock are provided
  - HGO, RANGE, EREFS, ERCLKEN, EREFSTEN
- FLL engaged internal mode is automatically selected out of reset

10.1.2 Modes of Operation

There are seven modes of operation for the ICS: FEI, FEE, FBI, FBILP, FBE, FBELP, and stop.

10.1.2.1 FLL Engaged Internal (FEI)

In FLL engaged internal mode, which is the default mode, the ICS supplies a clock derived from the FLL which is controlled by the internal reference clock.

10.1.2.2 FLL Engaged External (FEE)

In FLL engaged external mode, the ICS supplies a clock derived from the FLL which is controlled by an external reference clock.

10.1.2.3 FLL Bypassed Internal (FBI)

In FLL bypassed internal mode, the FLL is enabled and controlled by the internal reference clock, but is bypassed. The ICS supplies a clock derived from the internal reference clock.

10.1.2.4 FLL Bypassed Internal Low Power (FBILP)

In FLL bypassed internal low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the internal reference clock.

10.1.2.5 FLL Bypassed External (FBE)

In FLL bypassed external mode, the FLL is enabled and controlled by an external reference clock, but is bypassed. The ICS supplies a clock derived from the external reference clock. The external reference clock can be an external crystal/resonator supplied by an OSC controlled by the ICS, or it can be another external clock source.

10.1.2.6 FLL Bypassed External Low Power (FBELP)

In FLL bypassed external low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the external reference clock. The external reference clock can be an external crystal/resonator supplied by an OSC controlled by the ICS, or it can be another external clock source.

10.1.2.7 Stop (STOP)

In stop mode, the FLL is disabled and the internal or external reference clocks can be selected to be enabled or disabled. The ICS does not provide an MCU clock source.

10.1.3 Block Diagram

Figure 10-2 is the ICS block diagram.

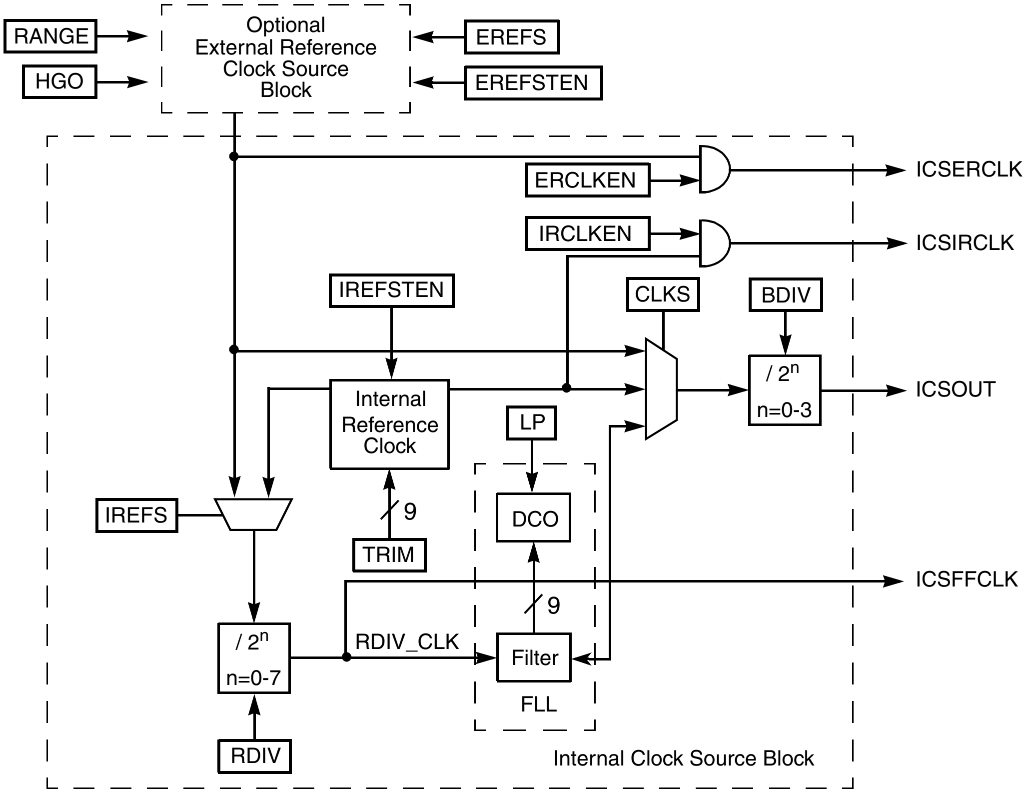


Figure 10-2. Internal Clock Source (ICS) Block Diagram

10.2 External Signal Description

There are no ICS signals that connect off chip.

10.3 Register Definition

Figure 10-1 is a summary of ICS registers.

Table 10-1. ICS Register Summary

Name		7	6	5	4	3	2	1	0
ICSC1	R	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
	W								
ICSC2	R	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
	W								
ICSTRM	R	TRIM							
	W								
ICSSC	R	0	0	0	0	CLKST		OSCINIT	FTRIM
	W								

10.3.1 ICS Control Register 1 (ICSC1)

	7	6	5	4	3	2	1	0
R W	CLKS		RDIV			IREFS	IRCLKEN	IREFSTEN
Reset:	0	0	0	0	0	1	0	0

Figure 10-3. ICS Control Register 1 (ICSC1)

Table 10-2. ICS Control Register 1 Field Descriptions

Field	Description
7:6 CLKS	**Clock Source Select** — Selects the clock source that controls the bus frequency. The actual bus frequency depends on the value of the BDIV bits. 00 Output of FLL is selected 01 Internal reference clock is selected 10 External reference clock is selected 11 Reserved, defaults to 00
5:3 RDIV	**Reference Divider** — Selects the amount to divide down the FLL reference clock selected by the IREFS bit. Resulting frequency must be in the range 31.25 kHz to 39.0625 kHz. 000 Encoding 0 — Divides reference clock by 1 (reset default) 001 Encoding 1 — Divides reference clock by 2 010 Encoding 2 — Divides reference clock by 4 011 Encoding 3 — Divides reference clock by 8 100 Encoding 4 — Divides reference clock by 16 101 Encoding 5 — Divides reference clock by 32 110 Encoding 6 — Divides reference clock by 64 111 Encoding 7 — Divides reference clock by 128
2 IREFS	**Internal Reference Select** — The IREFS bit selects the reference clock source for the FLL. 0 External reference clock selected 1 Internal reference clock selected
1 IRCLKEN	**Internal Reference Clock Enable** — The IRCLKEN bit enables the internal reference clock for use as ICSIRCLK. 0 ICSIRCLK inactive 1 ICSIRCLK active
0 IREFSTEN	**Internal Reference Stop Enable** — The IREFSTEN bit controls whether or not the internal reference clock remains enabled when the ICS enters stop mode. 0 Internal reference clock is disabled in stop 1 Internal reference clock stays enabled in stop if IRCLKEN is set or if ICS is in FEI, FBI, or FBILP mode before entering stop

10.3.2 ICS Control Register 2 (ICSC2)

	7	6	5	4	3	2	1	0
R W	BDIV		RANGE	HGO	LP	EREFS	ERCLKEN	EREFSTEN
Reset:	0	1	0	0	0	0	0	0

Figure 10-4. ICS Control Register 2 (ICSC2)

Table 10-3. ICS Control Register 2 Field Descriptions

Field	Description
7:6 BDIV	**Bus Frequency Divider** — Selects the amount to divide down the clock source selected by the CLKS bits. This controls the bus frequency. 00 Encoding 0 — Divides selected clock by 1 01 Encoding 1 — Divides selected clock by 2 (reset default) 10 Encoding 2 — Divides selected clock by 4 11 Encoding 3 — Divides selected clock by 8
5 RANGE	**Frequency Range Select** — Selects the frequency range for the external oscillator. 0 Low frequency range selected for the external oscillator 1 High frequency range selected for the external oscillator
4 HGO	**High Gain Oscillator Select** — The HGO bit controls the external oscillator mode of operation. 0 Configuring external oscillator for low power operation 1 Configuring external oscillator for high gain operation
3 LP	**Low Power Select** — The LP bit controls whether the FLL is disabled in FLL bypassed mode. 0 FLL is not disabled in bypass mode 1 FLL is disabled in bypass mode
2 EREFS	**External Reference Select** — The EREFS bit selects the source for the external reference clock. 0 External Clock Source requested 1 Oscillator requested
1 ERCLKEN	**External Reference Enable** — The ERCLKEN bit enables the external reference clock for use as ICSERCLK. 0 ICSERCLK inactive 1 ICSERCLK active
0 EREFSTEN	**External Reference Stop Enable** — The EREFSTEN bit controls whether or not the external reference clock remains enabled when the ICS enters stop mode. 0 External reference clock is disabled in stop 1 External reference clock stays enabled in stop if ERCLKEN is set or if ICS is in FEE, FBE, or FBELP mode before entering stop

10.3.3 ICS Trim Register (ICSTRM)

	7	6	5	4	3	2	1	0
R W	TRIM							
POR:	1	0	0	0	0	0	0	0
Reset:	U	U	U	U	U	U	U	U

Figure 10-5. ICS Trim Register (ICSTRM)

Table 10-4. ICS Trim Register Field Descriptions

Field	Description
7:0 TRIM	**ICS Trim Setting** — The TRIM bits control the internal reference clock frequency by controlling the internal reference clock period. The bits' effect is binary weighted (i.e., bit 1 will adjust twice as much as bit 0). Increasing the binary value in TRIM will increase the period, and decreasing the value will decrease the period. An additional fine trim bit is available in ICSSC as the FTRIM bit.

10.3.4 ICS Status and Control (ICSSC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	CLKST		OSCINIT	FTRIM
W								
POR:	0	0	0	0	0	0	0	0
Reset:	0	0	0	0	0	0	0	U

Figure 10-6. ICS Status and Control Register (ICSSC)

Table 10-5. ICS Status and Control Register Field Descriptions

Field	Description
7:4	Reserved, must be cleared.
3-2 CLKST	**Clock Mode Status** — The CLKST bits indicate the current clock mode. The CLKST bits don't update immediately after a write to the CLKS bits due to internal synchronization between clock domains. 00 Output of FLL is selected. 01 FLL bypassed, internal reference clock is selected. 10 FLL bypassed, external reference clock is selected. 11 Reserved.
1 OSCINIT	**OSC Initialization** — If the external reference clock is selected by ERCLKEN or by the ICS being in FEE, FBE, or FBELP mode, and if EREFS is set, then this bit is set after the initialization cycles of the external oscillator clock have completed. This bit is only cleared when ERCLKEN or EREFS is cleared.
0 FTRIM	**ICS Fine Trim** — The FTRIM bit controls the smallest adjustment of the internal reference clock frequency. Setting FTRIM will increase the period and clearing FTRIM will decrease the period by the smallest amount possible.

10.4 Functional Description

10.4.1 Operational Modes

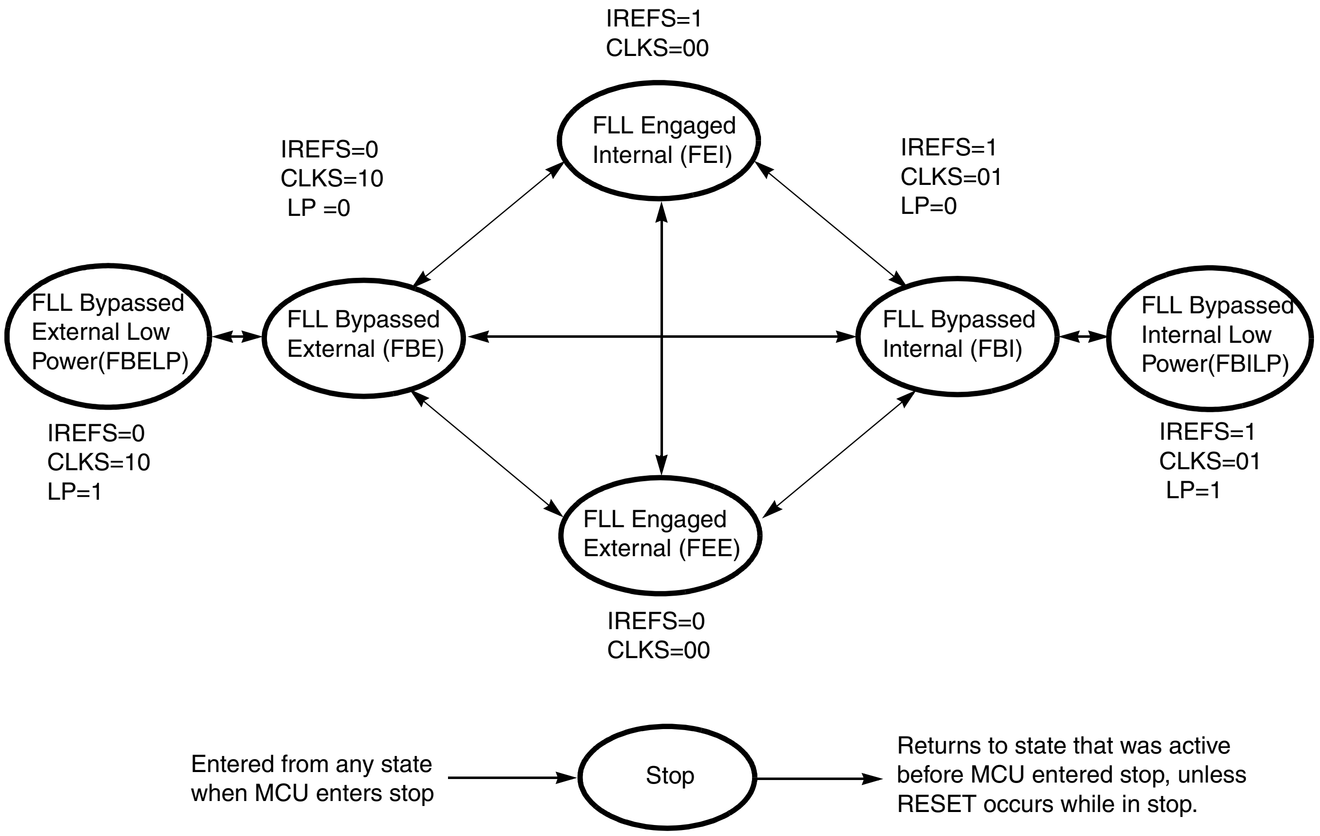


Figure 10-7. Clock Switching Modes

The seven states of the ICS are shown as a state diagram and are described below. The arrows indicate the allowed movements between the states.

10.4.1.1 FLL Engaged Internal (FEI)

FLL engaged internal (FEI) is the default mode of operation and is entered when all the following conditions occur:

- CLKS bits are written to 00
- IREFS bit is written to 1
- RDIV bits are written to divide reference clock to be within the range of 31.25 kHz to 39.0625 kHz.

In FLL engaged internal mode, the ICSOUT clock is derived from the FLL clock, which is controlled by the internal reference clock. The FLL loop will lock the frequency to 512 times the filter frequency, as selected by the RDIV bits. The internal reference clock is enabled.

10.4.1.2 FLL Engaged External (FEE)

The FLL engaged external (FEE) mode is entered when all the following conditions occur:

- CLKS bits are written to 00
- IREFS bit is written to 0
- RDIV bits are written to divide reference clock to be within the range of 31.25 kHz to 39.0625 kHz

In FLL engaged external mode, the ICSOUT clock is derived from the FLL clock which is controlled by the external reference clock. The FLL loop will lock the frequency to 512 times the filter frequency, as selected by the RDIV bits. The external reference clock is enabled.

10.4.1.3 FLL Bypassed Internal (FBI)

The FLL bypassed internal (FBI) mode is entered when all the following conditions occur:

- CLKS bits are written to 01
- IREFS bit is written to 1.
- LP bit is written to 0

In FLL bypassed internal mode, the ICSOUT clock is derived from the internal reference clock. The FLL clock is controlled by the internal reference clock, and the FLL loop will lock the FLL frequency to 512 times the filter frequency, as selected by the RDIV bits. The internal reference clock is enabled.

10.4.1.4 FLL Bypassed Internal Low Power (FBILP)

The FLL bypassed internal low power (FBILP) mode is entered when all the following conditions occur:

- CLKS bits are written to 01
- IREFS bit is written to 1.
- LP bit is written to 1

In FLL bypassed internal low power mode, the ICSOUT clock is derived from the internal reference clock and the FLL is disabled. The internal reference clock is enabled.

10.4.1.5 FLL Bypassed External (FBE)

The FLL bypassed external (FBE) mode is entered when all the following conditions occur:

- CLKS bits are written to 10.
- IREFS bit is written to 0.
- LP bit is written to 0.

In FLL bypassed external mode, the ICSOUT clock is derived from the external reference clock. The FLL clock is controlled by the external reference clock, and the FLL loop will lock the FLL frequency to 512 times the filter frequency, as selected by the RDIV bits. The external reference clock is enabled.

10.4.1.6 FLL Bypassed External Low Power (FBELP)

The FLL bypassed external low power (FBELP) mode is entered when all the following conditions occur:

- CLKS bits are written to 10.
- IREFS bit is written to 0.
- LP bit is written to 1.

In FLL bypassed external low power mode, the ICSOUT clock is derived from the external reference clock and the FLL is disabled. The external reference clock is enabled.

10.4.1.7 Stop

Stop mode is entered whenever the MCU enters a STOP state. In this mode, all ICS clock signals are static except in the following cases:

ICSIRCLK will be active in stop mode when both of the following conditions occur:

- IRCLKEN bit is written to 1
- IREFSTEN bit is written to 1

ICSERCLK will be active in stop mode when both of the following conditions occur:

- ERCLKEN bit is written to 1
- EREFSTEN bit is written to 1

10.4.2 Mode Switching

When switching between FLL engaged internal (FEI) and FLL engaged external (FEE) modes, the IREFS bit can be changed at anytime, but the RDIV bits must be changed simultaneously so that the resulting frequency stays in the range of 31.25 kHz to 39.0625 kHz. After a change in the IREFS value, the FLL will begin locking again after a few full cycles of the resulting divided reference frequency.

The CLKS bits can also be changed at anytime, but the RDIV bits must be changed simultaneously so that the resulting frequency stays in the range of 31.25 kHz to 39.0625 kHz. The actual switch to the newly selected clock will not occur until after a few full cycles of the new clock. If the newly selected clock is not available, the previous clock will remain selected.

10.4.3 Bus Frequency Divider

The BDIV bits can be changed at anytime and the actual switch to the new frequency will occur immediately.

10.4.4 Low Power Bit Usage

The low power bit (LP) is provided to allow the FLL to be disabled and thus conserve power when it is not being used. However, in some applications it may be desirable to enable the FLL and allow it to lock for maximum accuracy before switching to an FLL engaged mode. Do this by writing the LP bit to 0.

10.4.5 Internal Reference Clock

When IRCLKEN is set, the internal reference clock signal will be presented as ICSIRCLK, which can be used as an additional clock source. The ICSIRCLK frequency can be re-targeted by trimming the period of the internal reference clock. This can be done by writing a new value to the TRIM bits in the ICSTRM register. Writing a larger value will slow down the ICSIRCLK frequency, and writing a smaller value to the ICSTRM register will speed up the ICSIRCLK frequency. The TRIM bits will effect the ICSOUT frequency if the ICS is in FLL engaged internal (FEI), FLL bypassed internal (FBI), or FLL bypassed internal low power (FBILP) mode. The TRIM and FTRIM value will not be affected by a reset.

Until ICSIRCLK is trimmed, programming low reference divider (RDIV) factors may result in ICSOUT frequencies exceeding the maximum chip-level frequency and violating the chip-level clock timing specifications (see the Device Overview chapter).

If IREFSTEN is set and the IRCLKEN bit is written to 1, the ICSIRCLK will keep running during stop mode in order to provide a fast recovery upon exiting stop.

All MCU devices are factory programmed with a trim value in a reserved memory location. This value can be copied to the ICSTRM register during reset initialization. The factory trim value does not include the FTRIM bit. For finer precision, the user can trim the internal oscillator in the application and set the FTRIM bit accordingly.

10.4.6 Optional External Reference Clock

The ICS module can support an external reference clock with frequencies between 31.25 kHz to 5 MHz in all modes. When the ERCLKEN is set, the external reference clock signal will be presented as ICSERCLK, which can be used as an additional clock source. When IREFS = 1, the external reference clock will not be used by the FLL and will only be used as ICSERCLK. In these modes, the frequency can be equal to the maximum frequency the chip-level timing specifications support (see the Device Overview chapter).

If EREFSTEN is set and the ERCLKEN bit is written to 1, the ICSERCLK will keep running during stop mode in order to provide a fast recovery upon exiting stop.

10.4.7 Fixed Frequency Clock

The ICS presents the divided FLL reference clock as ICSFFCLK for use as an additional clock source for peripheral modules. The ICS provides an output signal (ICSFFE) which indicates when the ICS is providing ICSOUT frequencies four times or greater than the divided FLL reference clock (ICSFFCLK). In FLL engaged mode (FEI and FEE) this is always true and ICSFFE is always high. In ICS bypass mode, ICSFFE will get asserted for the following combinations of BDIV and RDIV values:

- BDIV = 00 (divide by 1), RDIV ≥ 010
- BDIV = 01 (divide by 2), RDIV ≥ 011
- BDIV = 10 (divide by 4), RDIV ≥ 100
- BDIV = 11 (divide by 8), RDIV ≥ 101

Chapter 11
Modulo Timer (RS08MTIMV1)

11.1 Introduction

The MTIM is a simple 8-bit timer with several software-selectable clock sources and a programmable interrupt. For MCUs containing more than one MTIM, MTIMs are collectively called MTIMs. For example, MTIMx for an MCU with two MTIMs refer to MTIM1 and MTIM2.

The central component of the MTIM is the 8-bit counter that can operate as a free-running counter or a modulo counter. A timer overflow interrupt can be enabled to generate periodic interrupts for time-based software loops.

MTIM has three optional reference clocks in SC9RS08MZ8. They are TCLK clock, ICSFFCLK clock, and bus clock from ICS module.

Figure 11-1 shows the SC9RS08MZ8 block diagram with the MTIM modules and pin highlighted.

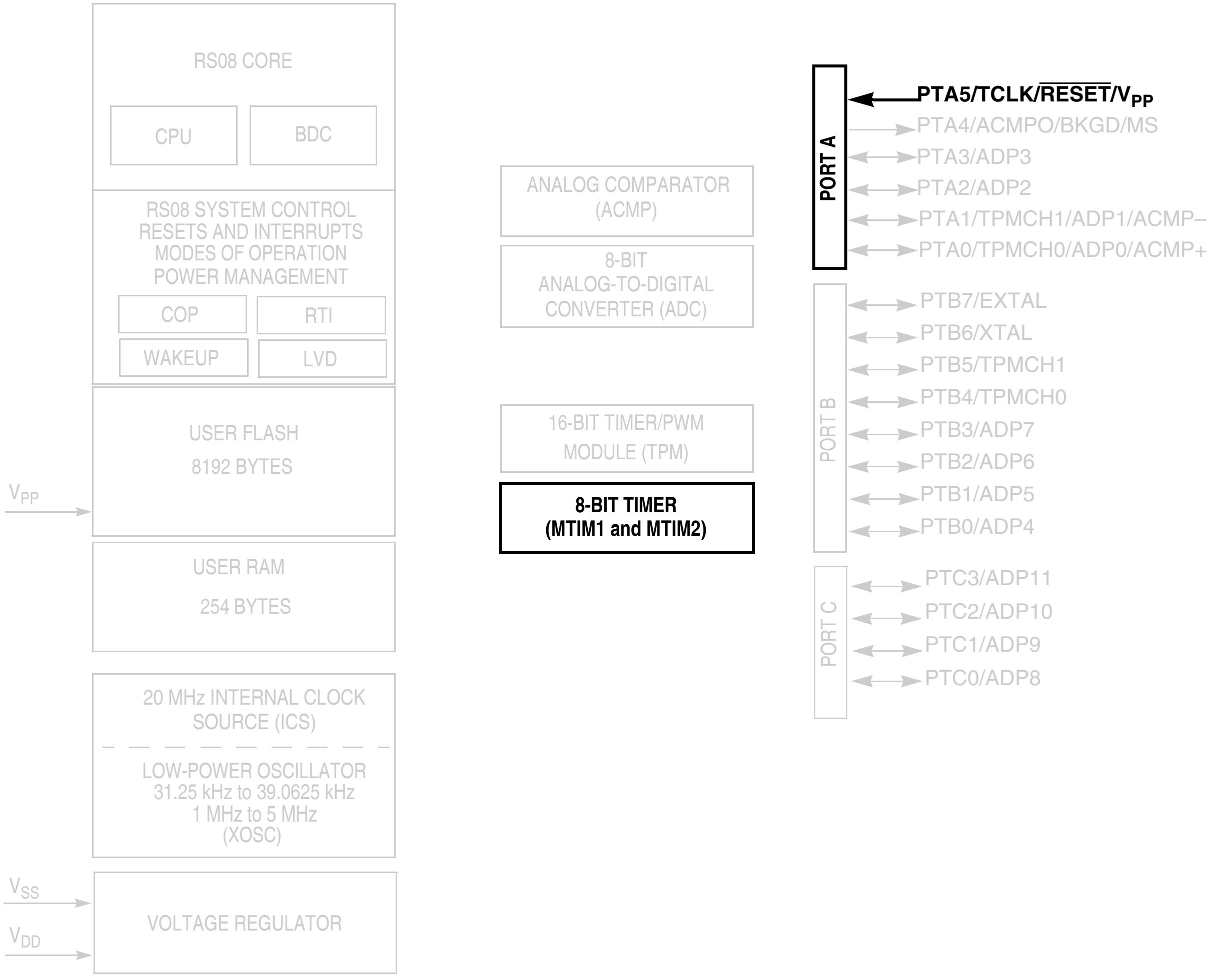


Figure 11-1. SC9RS08MZ8 Block Diagram Highlighting MTIM Blocks and Pin

11.1.1 Features

Timer system features include:

- 8-bit up-counter
  - Free-running or 8-bit modulo limit
  - Software controllable interrupt on overflow
  - Counter reset bit (TRST)
  - Counter stop bit (TSTP)
- Four software selectable clock sources for input to prescaler:
  - System bus clock — rising edge
  - Fixed frequency clock (XCLK) — rising edge
  - External clock source on the TCLK pin — rising edge
  - External clock source on the TCLK pin — falling edge
- Nine selectable clock prescale values:
  - Clock source divide by 1, 2, 4, 8, 16, 32, 64, 128, or 256

11.1.2 Modes of Operation

This section defines the MTIM's operation in stop, wait and background debug modes.

11.1.2.1 Operation in Wait Mode

The MTIM continues to run in wait mode if enabled before executing the WAIT instruction. Therefore, the MTIM can be used to bring the MCU out of wait mode if the timer overflow interrupt is enabled. For lowest possible current consumption, the MTIM must be disabled by software if not needed as an interrupt source during wait mode.

11.1.2.2 Operation in Stop Modes

The MTIM is disabled in all stop modes, regardless of the settings before executing the STOP instruction. Therefore, the MTIM cannot be used as a wake up source from stop mode.

If stop is exited with a reset, the MTIM will be put into its reset state. If stop is exited with an interrupt, the MTIM continues from the state it was in when stop was entered. If the counter was active upon entering stop, the count will resume from the current value.

11.1.2.3 Operation in Active Background Mode

The MTIM suspends all counting until the MCU returns to normal user operating mode. Counting resumes from the suspended value as long as an MTIM reset did not occur (TRST written to a 1 or any value is written to the MTIMMOD register).

11.1.3 Block Diagram

The block diagram for the modulo timer module is shown Figure 11-2.

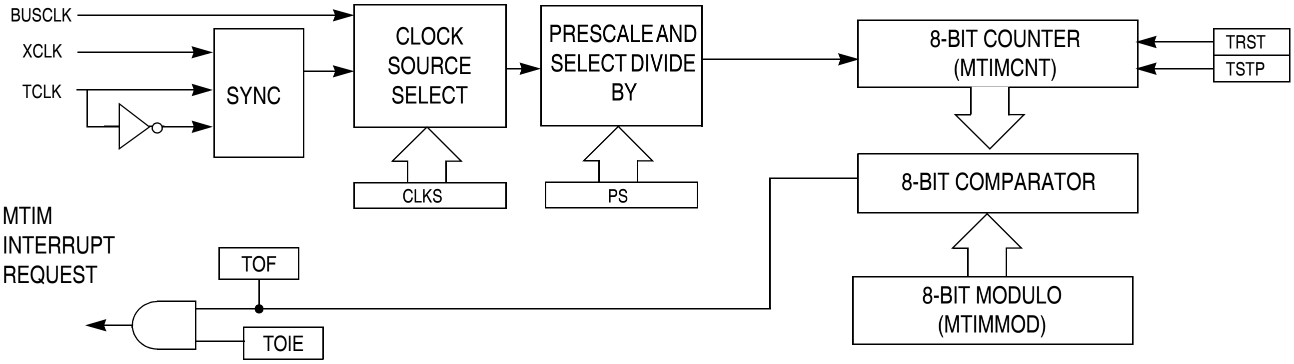


Figure 11-2. Modulo Timer (MTIM) Block Diagram

11.2 External Signal Description

The MTIM includes one external signal, TCLK, used to input an external clock when selected as the MTIM clock source. The signal properties of TCLK are shown in Table 11-1.

Table 11-1. Signal Properties

Signal	Function	I/O
TCLK	External clock source input into MTIM	I

The TCLK input must be synchronized by the bus clock. Also, variations in duty cycle and clock jitter must be accommodated. Therefore, the TCLK signal must be limited to one-fourth of the bus frequency.

The TCLK pin can be muxed with a general-purpose port pin. See the Pins and Connections chapter for the pin location and priority of this function.

11.3 Register Definition

Each MTIM includes four registers, which are summarized in Table 11-2:

- An 8-bit status and control register
- An 8-bit clock configuration register
- An 8-bit counter register
- An 8-bit modulo register

Refer to the direct-page register summary in the memory section of this data sheet for the absolute address assignments for all MTIM registers. This section refers to registers and control bits only by their names.

Table 11-2. MTIM Register Summary

Name		7	6	5	4	3	2	1	0
MTIMSC	R	TOF	TOIE	0	TSTP	0	0	0	0
	W			TRST					
MTIMCLK	R	0	0	CLKS		PS			
	W								
MTIMCNT	R	COUNT							
	W								
MTIMMOD	R	MOD							
	W								

11.3.1 MTIM Status and Control Register (MTIMSC)

MTIMSC contains the overflow status flag and control bits which are used to configure the interrupt enable, reset the counter, and stop the counter.

	7	6	5	4	3	2	1	0
R	TOF	TOIE	0	TSTP	0	0	0	0
W			TRST					
Reset:	0	0	0	1	0	0	0	0

Figure 11-3. MTIM Status and Control Register (MTIMSC)

Table 11-3. MTIMSC Field Descriptions

Field	Description
7 TOF	**MTIM Overflow Flag** — This read-only bit is set when the MTIM counter register overflows to \$00 after reaching the value in the MTIM modulo register. Clear TOF by reading the MTIMSC register while TOF is set, then writing a 0 to TOF. TOF is also cleared when TRST is written to a 1 or when any value is written to the MTIMMOD register. 0 MTIM counter has not reached the overflow value in the MTIM modulo register. 1 MTIM counter has reached the overflow value in the MTIM modulo register.
6 TOIE	**MTIM Overflow Interrupt Enable** — This read/write bit enables MTIM overflow interrupts. If TOIE is set, then an interrupt is generated when TOF = 1. Reset clears TOIE. Do not set TOIE if TOF = 1. Clear TOF first, then set TOIE. 0 TOF interrupts are disabled. Use software polling. 1 TOF interrupts are enabled.
5 TRST	**MTIM Counter Reset** — When a 1 is written to this write-only bit, the MTIM counter register resets to \$00 and TOF is cleared. Reading this bit always returns 0. 0 No effect. MTIM counter remains at current state. 1 MTIM counter is reset to \$00.
4 TSTP	**MTIM Counter Stop** — When set, this read/write bit stops the MTIM counter at its current value. Counting resumes from the current value when TSTP is cleared. Reset sets TSTP to prevent the MTIM from counting. 0 MTIM counter is active. 1 MTIM counter is stopped.

11.3.2 MTIM Clock Configuration Register (MTIMCLK)

MTIMCLK contains the clock select bits (CLKS) and the prescaler select bits (PS).

	7	6	5	4	3	2	1	0
R	0	0	CLKS		PS			
W								
Reset:	0	0	0	0	0	0	0	0

Figure 11-4. MTIM Clock Configuration Register (MTIMCLK)

Table 11-4. MTIMCLK Field Description

Field	Description
5:4 CLKS	**Clock Source Select** — These two read/write bits select one of four different clock sources as the input to the MTIM prescaler. Changing the clock source while the counter is active does not clear the counter. The count continues with the new clock source. Reset clears CLKS to 00. 00 Encoding 0 — Bus clock (BUSCLK). 01 Encoding 1 — Fixed-frequency clock (XCLK). 10 Encoding 3 — External source (TCLK pin), falling edge. 11 Encoding 4 — External source (TCLK pin), rising edge.
3:0 PS	**Clock Source Prescaler** — These four read/write bits select one of nine outputs from the 8-bit prescaler. Changing the prescaler value while the counter is active does not clear the counter. The count continues with the new prescaler value. Reset clears PS to 0000. 0000 Encoding 0 — MTIM clock source ÷ 1. 0001 Encoding 1 — MTIM clock source ÷ 2. 0010 Encoding 2 — MTIM clock source ÷ 4. 0011 Encoding 3 — MTIM clock source ÷ 8. 0100 Encoding 4 — MTIM clock source ÷ 16. 0101 Encoding 5 — MTIM clock source ÷ 32. 0110 Encoding 6 — MTIM clock source ÷ 64. 0111 Encoding 7 — MTIM clock source ÷ 128. 1000 Encoding 8 — MTIM clock source ÷ 256. All other encodings default to MTIM clock source ÷ 256.

11.3.3 MTIM Counter Register (MTIMCNT)

MTIMCNT is the read-only value of the current MTIM count of the 8-bit counter.

	7	6	5	4	3	2	1	0
R	COUNT							
W								
Reset:	0	0	0	0	0	0	0	0

Figure 11-5. MTIM Counter Register (MTIMCNT)

Table 11-5. MTIMCNT Field Description

Field	Description
7:0 COUNT	**MTIM Count** — These eight read-only bits contain the current value of the 8-bit counter. Writes have no effect to this register. Reset clears the count to $00.

11.3.4 MTIM Modulo Register (MTIMMOD)

	7	6	5	4	3	2	1	0
R W	MOD							
Reset:	0	0	0	0	0	0	0	0

Figure 11-6. MTIM Modulo Register (MTIMMOD)

Table 11-6. MTIMMOD Descriptions

Field	Description
7:0 MOD	**MTIM Modulo** — These eight read/write bits contain the modulo value used to reset the count and set TOF. A value of $00 puts the MTIM in free-running mode. Writing to MTIMMOD resets the COUNT to $00 and clears TOF. Reset sets the modulo to $00.

11.4 Functional Description

The MTIM is composed of a main 8-bit up-counter with an 8-bit modulo register, a clock source selector, and a prescaler block with nine selectable values. The module also contains software selectable interrupt logic.

The MTIM counter (MTIMCNT) has three modes of operation: stopped, free-running, and modulo. Out of reset, the counter is stopped. If the counter is started without writing a new value to the modulo register, then the counter will be in free-running mode. The counter is in modulo mode when a value other than $00 is in the modulo register while the counter is running.

After any MCU reset, the counter is stopped and reset to $00, and the modulus is set to $00. The bus clock is selected as the default clock source and the prescale value is divide by 1. To start the MTIM in free-running mode, simply write to the MTIM status and control register (MTIMSC) and clear the MTIM stop bit (TSTP).

Four clock sources are software selectable: the internal bus clock, the fixed frequency clock (XCLK), and an external clock on the TCLK pin, selectable as incrementing on either rising or falling edges. The MTIM clock select bits (CLKS) in MTIMCLK are used to select the desired clock source. If the counter is active (TSTP = 0) when a new clock source is selected, the counter will continue counting from the previous value using the new clock source.

Nine prescale values are software selectable: clock source divided by 1, 2, 4, 8, 16, 32, 64, 128, or 256. The prescaler select bits (PS) in MTIMCLK select the desired prescale value. If the counter is active (TSTP = 0) when a new prescaler value is selected, the counter will continue counting from the previous value using the new prescaler value.

The MTIM modulo register (MTIMMOD) allows the overflow compare value to be set to any value from $01 to $FF. Reset clears the modulo value to $00, which results in a free running counter.

When the counter is active (TSTP = 0), the counter increments at the selected rate until the count matches the modulo value. When these values match, the counter overflows to $00 and continues counting. The MTIM overflow flag (TOF) is set whenever the counter overflows. The flag sets on the transition from the modulo value to $00. Writing to MTIMMOD while the counter is active resets the counter to $00 and clears TOF.

Clearing TOF is a two-step process. The first step is to read the MTIMSC register while TOF is set. The second step is to write a 0 to TOF. If another overflow occurs between the first and second steps, the clearing process is reset and TOF will remain set after the second step is performed. This will prevent the second occurrence from being missed. TOF is also cleared when a 1 is written to TRST or when any value is written to the MTIMMOD register.

The MTIM allows for an optional interrupt to be generated whenever TOF is set. To enable the MTIM overflow interrupt, set the MTIM overflow interrupt enable bit (TOIE) in MTIMSC. TOIE must never be written to a 1 while TOF = 1. Instead, TOF must be cleared first, then the TOIE can be set to 1.

11.4.1 MTIM Operation Example

This section shows an example of the MTIM operation as the counter reaches a matching value from the modulo register.

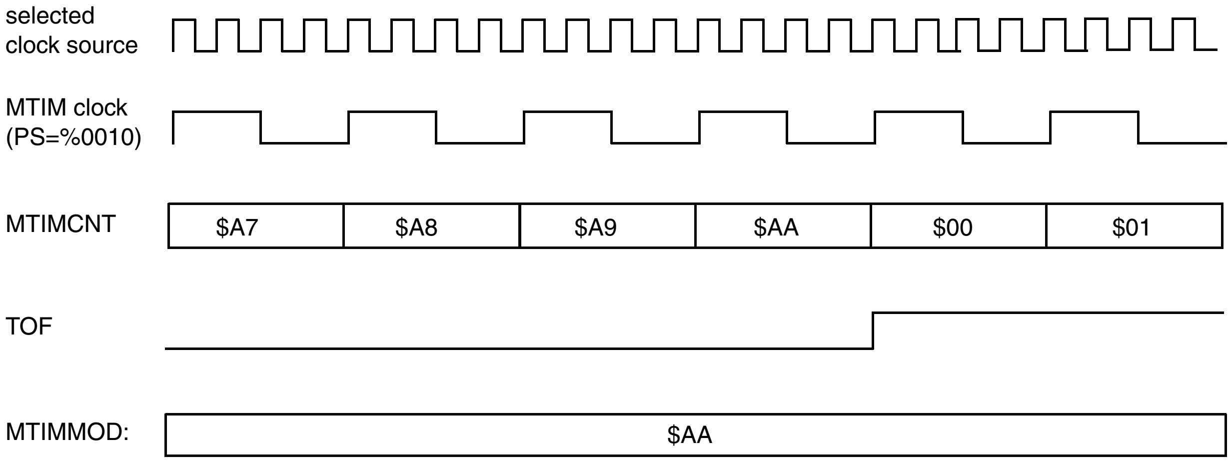


Figure 11-7. MTIM Counter Overflow Example

In the example of Figure 11-7, the selected clock source could be any of the four possible choices. The prescaler is set to PS = %0010 or divide-by-4. The modulo value in the MTIMMOD register is set to $AA. When the counter, MTIMCNT, reaches the modulo value of $AA, the counter overflows to $00 and continues counting. The timer overflow flag, TOF, sets when the counter value changes from $AA to $00. An MTIM overflow interrupt is generated when TOF is set, if TOIE = 1.

Chapter 12
16-Bit Timer/PWM (RS08TPMV2)

12.1 Introduction

The TPM uses one input/output (I/O) pin per channel, TPMCHn where x is the TPM number (for example, 1 or 2), and n is the channel number (for example, 0–4). The TPM shares its I/O pins with general-purpose I/O port pins (refer to the Chapter 2, “Pins and Connections.”)

The TPM module pins, TPMCH0 and TPMCH1 can be repositioned under software control using TPMCH0PS in SPT (Table 12-1). TPMCH0PS and TPMCH1PS in SOPT select which general-purpose I/O ports are associated with TPM.

Table 12-1. TPM Position Options

TPMCH0PS in SOPT	Port Pin for TPMCH0
0 (default)	PTA0
1	PTB4

TPMCH1PS in SOPT	Port Pin for TPMCH1
0 (default)	PTA1
1	PTB5

Figure 12-1 shows the SC9RS08MZ8 block diagram with the TPM module and pins highlighted.

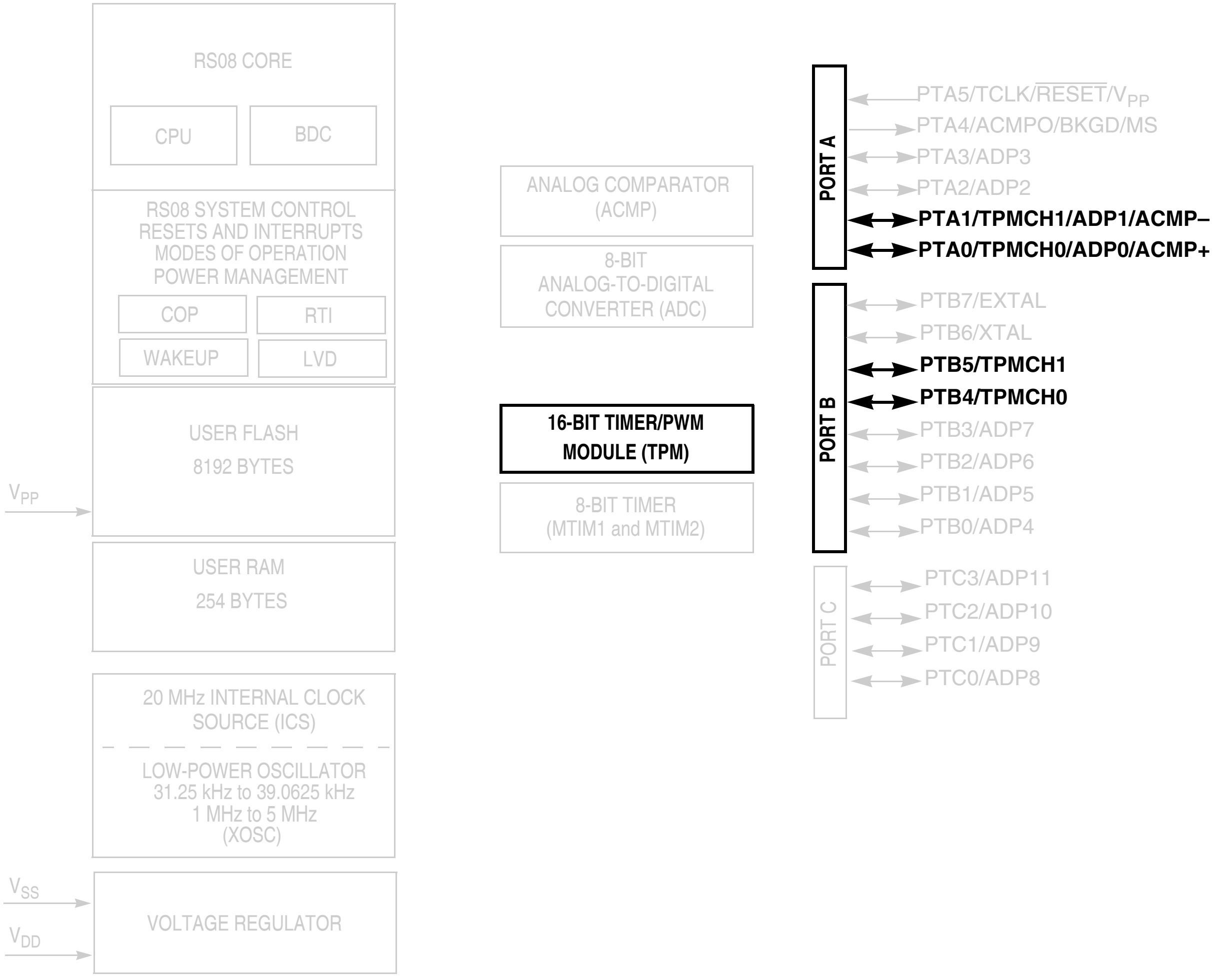


Figure 12-1. SC9RS08MZ8 Block Diagram Highlighting TPM Block and Pins

12.1.1 Features

The TPM has the following features:

- Each TPM may be configured for buffered, center-aligned pulse-width modulation (CPWM) on all channels
- Clock sources independently selectable per TPM (multiple TPMs device)
- Selectable clock sources (device dependent): bus clock, fixed system clock, external pin
- Clock prescaler taps for divide by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit free-running or up/down (CPWM) count operation
- 16-bit modulus register to control counter range
- Timer system enable
- One interrupt per channel plus a terminal count interrupt for each TPM module (multiple TPMs device)
- Channel features:
  - Each channel may be input capture, output compare, or buffered edge-aligned PWM
  - Rising-edge, falling-edge, or any-edge input capture trigger
  - Set, clear, or toggle output compare action
  - Selectable polarity on PWM outputs

12.1.2 Block Diagram

Figure 12-2 shows the structure of a TPM. Some MCUs include more than one TPM, with various numbers of channels.

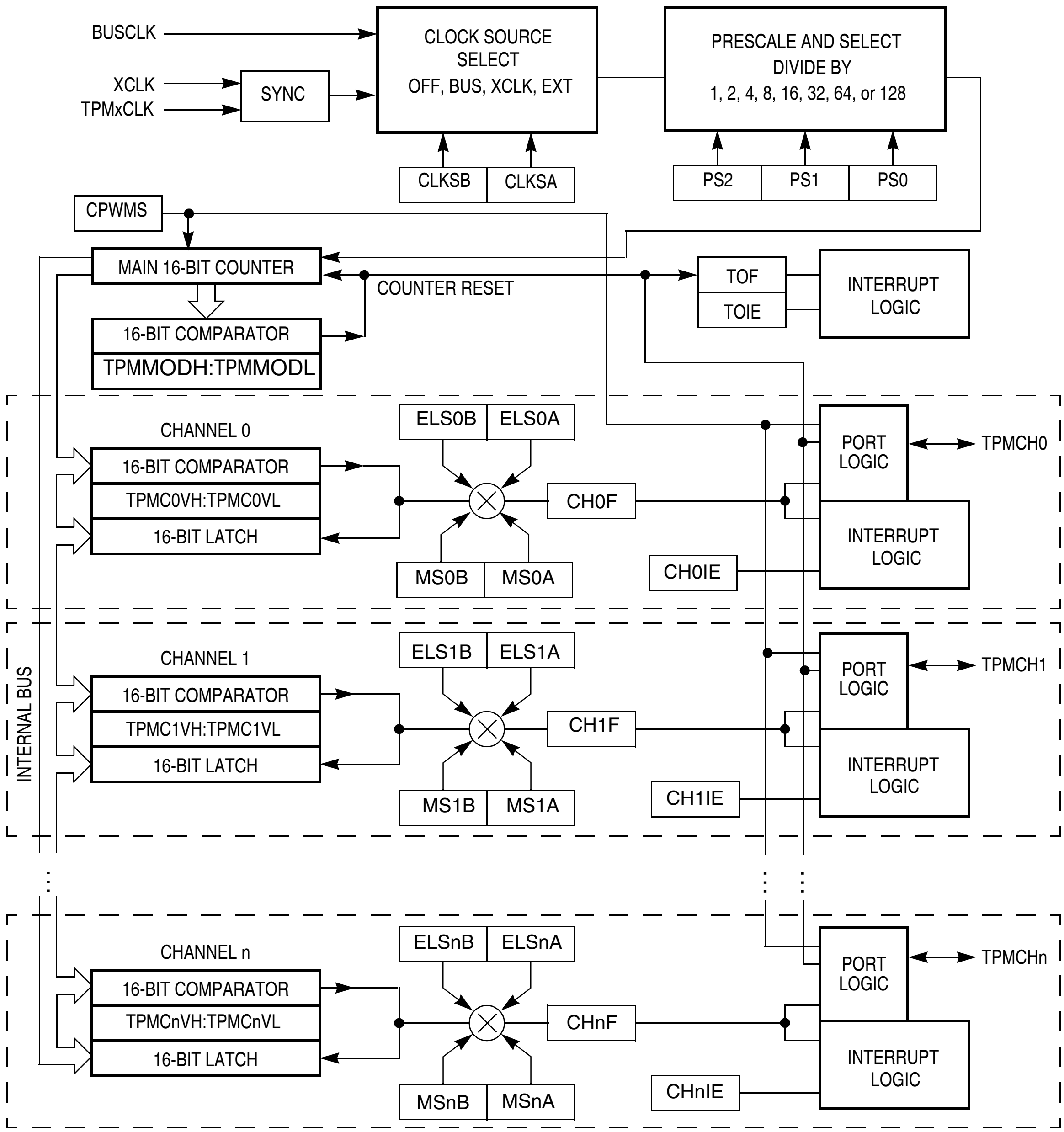


Figure 12-2. TPM Block Diagram

The central component of the TPM is the 16-bit counter that can operate as a free-running counter, a modulo counter, or an up-/down-counter when the TPM is configured for center-aligned PWM. The TPM counter (when operating in normal up-counting mode) provides the timing reference for the input capture, output compare, and edge-aligned PWM functions. The timer counter modulo registers, TPMMODH:TPMMODL, control the modulo value of the counter. (The values 0x0000 or 0xFFFF effectively make the counter free running.) Software can read the counter value at any time without affecting the counting sequence. Any write to either byte of the TPMCNT counter resets the counter regardless of the data value written.

All TPM channels are programmable independently as input capture, output compare, or buffered edge-aligned PWM channels.

12.2 External Signal Description

When any pin associated with the timer is configured as a timer input, a passive pullup can be enabled. After reset, the TPM modules are disabled and all pins default to general-purpose inputs with the passive pullups disabled.

12.2.1 External TPM Clock Sources

When control bits CLKSB:CLKSA in the timer status and control register are set to 1:1, the prescaler and consequently the 16-bit counter for TPM are driven by an external clock source, TPMxCLK, connected to an I/O pin. A synchronizer is needed between the external clock and the rest of the TPM. This synchronizer is clocked by the bus clock so the frequency of the external source must be less than one-half the frequency of the bus rate clock. The upper frequency limit for this external clock source is specified to be one-fourth the bus frequency to conservatively accommodate duty cycle and phase-locked loop (PLL) or frequency-locked loop (FLL) frequency jitter effects.

On some devices the external clock input is shared with one of the TPM channels. When a TPM channel is shared as the external clock input, the associated TPM channel cannot use the pin. (The channel can still be used in output compare mode as a software timer.) Also, if one of the TPM channels is used as the external clock input, the corresponding ELSnB:ELSnA control bits must be set to 0:0 so the channel is not trying to use the same pin.

12.2.2 TPMCHn — TPM Channel n I/O Pins

Each TPM channel is associated with an I/O pin on the MCU. The function of this pin depends on the configuration of the channel. In some cases, no pin function is needed so the pin reverts to being controlled by general-purpose I/O controls. When a timer has control of a port pin, the port data and data direction registers do not affect the related pin(s). See the Pins and Connections chapter for additional information about shared pin functions.

12.3 Register Definition

The TPM includes:

- An 8-bit status and control register (TPMSC)
- A 16-bit counter (TPMCNTH:TPMCNTL)
- A 16-bit modulo register (TPMMODH:TPMMODL)

Each timer channel has:

- An 8-bit status and control register (TPMCnSC)
- A 16-bit channel value register (TPMCnVH:TPMCnVL)

Refer to the direct-page register summary in the Memory chapter of this data sheet for the absolute address assignments for all TPM registers. This section refers to registers and control bits only by their names. A

Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

12.3.1 Timer Status and Control Register (TPMSC)

TPMSC contains the overflow status flag and control bits that are used to configure the interrupt enable, TPM configuration, clock source, and prescale divisor. These controls relate to all channels within this timer module.

	7	6	5	4	3	2	1	0
R	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
W	▨							
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 12-3. Timer Status and Control Register (TPMSC)

Table 12-2. TPMSC Register Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — This flag is set when the TPM counter changes to 0x0000 after reaching the modulo value programmed in the TPM counter modulo registers. When the TPM is configured for CPWM, TOF is set after the counter has reached the value in the modulo register, at the transition to the next lower count value. Clear TOF by reading the TPM status and control register when TOF is set and then writing a 0 to TOF. If another TPM overflow occurs before the clearing sequence is complete, the sequence is reset so TOF would remain set after the clear sequence was completed for the earlier TOF. Reset clears TOF. Writing a 1 to TOF has no effect. 0 TPM counter has not reached modulo value or overflow 1 TPM counter has overflowed
6 TOIE	**Timer Overflow Interrupt Enable** — This read/write bit enables TPM overflow interrupts. If TOIE is set, an interrupt is generated when TOF equals 1. Reset clears TOIE. 0 TOF interrupts inhibited (use software polling) 1 TOF interrupts enabled
5 CPWMS	**Center-Aligned PWM Select** — This read/write bit selects CPWM operating mode. Reset clears this bit so the TPM operates in up-counting mode for input capture, output compare, and edge-aligned PWM functions. Setting CPWMS reconfigures the TPM to operate in up-/down-counting mode for CPWM functions. Reset clears CPWMS. 0 All TPM channels operate as input capture, output compare, or edge-aligned PWM mode as selected by the MSnB:MSnA control bits in each channel's status and control register 1 All TPM channels operate in center-aligned PWM mode
4:3 CLKS[B:A]	**Clock Source Select** — As shown in Table 12-3, this 2-bit field is used to disable the TPM system or select one of three clock sources to drive the counter prescaler. The external source and the XCLK are synchronized to the bus clock by an on-chip synchronization circuit.
2:0 PS[2:0]	**Prescale Divisor Select** — This 3-bit field selects one of eight divisors for the TPM clock input as shown in Table 12-4. This prescaler is located after any clock source synchronization or clock source selection, so it affects whatever clock source is selected to drive the TPM system.

Table 12-3. TPM Clock Source Selection

CLKSB:CLKSA	TPM Clock Source to Prescaler Input
0:0	No clock selected (TPM disabled)
0:1	Bus rate clock (BUSCLK)
1:0	Fixed system clock (XCLK)
1:1	External source (TPMCLK)[1,2]

[1] The maximum frequency that is allowed as an external clock is one-fourth of the bus frequency.

[2] If the external clock input is shared with channel n and is selected as the TPM clock source, the corresponding ELSnB:ELSnA control bits must be set to 0:0 so channel n does not try to use the same pin for a conflicting function.

Table 12-4. Prescale Divisor Selection

PS2:PS1:PS0	TPM Clock Source Divided-By
0:0:0	1
0:0:1	2
0:1:0	4
0:1:1	8
1:0:0	16
1:0:1	32
1:1:0	64
1:1:1	128

12.3.2 Timer Counter Registers (TPMCNTH:TPMCNTL)

The two read-only TPM counter registers contain the high and low bytes of the value in the TPM counter. Reading either byte (TPMCNTH or TPMCNTL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. This allows coherent 16-bit reads in either order. The coherency mechanism is automatically restarted by an MCU reset, a write of any value to TPMCNTH or TPMCNTL, or any write to the timer status/control register (TPMSC).

Reset clears the TPM counter registers.

	7	6	5	4	3	2	1	0
R	Bit 15	14	13	12	11	10	9	Bit 8
W	Any write to TPMCNTH clears the 16-bit counter.							
Reset	0	0	0	0	0	0	0	0

Figure 12-4. Timer Counter Register High (TPMCNTH)

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W	Any write to TPMCNTL clears the 16-bit counter.							
Reset	0	0	0	0	0	0	0	0

Figure 12-5. Timer Counter Register Low (TPMCNTL)

When background mode is active, the timer counter and the coherency mechanism are frozen such that the buffer latches remain in the state they were in when the background mode became active even if one or both bytes of the counter are read while background mode is active.

12.3.3 Timer Counter Modulo Registers (TPMMODH:TPMMODL)

The read/write TPM modulo registers contain the modulo value for the TPM counter. After the TPM counter reaches the modulo value, the TPM counter resumes counting from 0x0000 at the next clock (CPWMS = 0) or starts counting down (CPWMS = 1), and the overflow flag (TOF) becomes set. Writing to TPMMODH or TPMMODL inhibits TOF and overflow interrupts until the other byte is written. Reset sets the TPM counter modulo registers to 0x0000, which results in a free-running timer counter (modulo disabled).

	7	6	5	4	3	2	1	0
R W	Bit 15	14	13	12	11	10	9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 12-6. Timer Counter Modulo Register High (TPMMODH)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 12-7. Timer Counter Modulo Register Low (TPMMODL)

It is good practice to wait for an overflow interrupt so both bytes of the modulo register can be written well before a new overflow. An alternative approach is to reset the TPM counter before writing to the TPM modulo registers to avoid confusion about when the first counter overflow will occur.

12.3.4 Timer Channel n Status and Control Register (TPMCnSC)

TPMCnSC contains the channel interrupt status flag and control bits that are used to configure the interrupt enable, channel configuration, and pin function.

	7	6	5	4	3	2	1	0
R	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
W							▨	▨
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 12-8. Timer Channel n Status and Control Register (TPMCnSC)

Table 12-5. TPMCnSC Register Field Descriptions

Field	Description
7 CHnF	**Channel n Flag** — When channel n is configured for input capture, this flag bit is set when an active edge occurs on the channel n pin. When channel n is an output compare or edge-aligned PWM channel, CHnF is set when the value in the TPM counter registers matches the value in the TPM channel n value registers. This flag is seldom used with center-aligned PWMs because it is set every time the counter matches the channel value register, which correspond to both edges of the active duty cycle period. A corresponding interrupt is requested when CHnF is set and interrupts are enabled (CHnIE = 1). Clear CHnF by reading TPMCnSC while CHnF is set and then writing a 0 to CHnF. If another interrupt request occurs before the clearing sequence is complete, the sequence is reset so CHnF would remain set after the clear sequence was completed for the earlier CHnF. This is done so a CHnF interrupt request cannot be lost by clearing a previous CHnF. Reset clears CHnF. Writing a 1 to CHnF has no effect. 0 No input capture or output compare event occurred on channel n 1 Input capture or output compare event occurred on channel n
6 CHnIE	**Channel n Interrupt Enable** — This read/write bit enables interrupts from channel n. Reset clears CHnIE. 0 Channel n interrupt requests disabled (use software polling) 1 Channel n interrupt requests enabled
5 MSnB	**Mode Select B for TPM Channel n** — When CPWMS = 0, MSnB = 1 configures TPM channel n for edge-aligned PWM mode. For a summary of channel mode and setup controls, refer to Table 12-6.
4 MSnA	**Mode Select A for TPM Channel n** — When CPWMS = 0 and MSnB = 0, MSnA configures TPM channel n for input capture mode or output compare mode. Refer to Table 12-6 for a summary of channel mode and setup controls.
3:2 ELSn[B:A]	**Edge/Level Select Bits** — Depending on the operating mode for the timer channel as set by CPWMS:MSnB:MSnA and shown in Table 12-6, these bits select the polarity of the input edge that triggers an input capture event, select the level that will be driven in response to an output compare match, or select the polarity of the PWM output. Setting ELSnB:ELSnA to 0:0 configures the related timer pin as a general-purpose I/O pin unrelated to any timer channel functions. This function is typically used to temporarily disable an input capture channel or to make the timer pin available as a general-purpose I/O pin when the associated timer channel is set up as a software timer that does not require the use of a pin.

Table 12-6. Mode, Edge, and Level Selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	XX	00	Pin not used for TPM channel; use as an external clock for the TPM or revert to general-purpose I/O	
0	00	01	Input capture	Capture on rising edge only
		10		Capture on falling edge only
		11		Capture on rising or falling edge
	01	00	Output compare	Software compare only
		01		Toggle output on compare
		10		Clear output on compare
		11		Set output on compare
	1X	10	Edge-aligned PWM	High-true pulses (clear output on compare)
		X1		Low-true pulses (set output on compare)
1	XX	10	Center-aligned PWM	High-true pulses (clear output on compare-up)
		X1		Low-true pulses (set output on compare-up)

If the associated port pin is not stable for at least two bus clock cycles before changing to input capture mode, it is possible to get an unexpected indication of an edge trigger. Typically, a program would clear status flags after changing channel configuration bits and before enabling channel interrupts or using the status flags to avoid any unexpected behavior.

12.3.5 Timer Channel Value Registers (TPMCnVH:TPMCnVL)

These read/write registers contain the captured TPM counter value of the input capture function or the output compare value for the output compare or PWM functions. The channel value registers are cleared by reset.

	7	6	5	4	3	2	1	0
R W	Bit 15	14	13	12	11	10	9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 12-9. Timer Channel Value Register High (TPMCnVH)

	7	6	5	4	3	2	1	0
R W	Bit 7	6	5	4	3	2	1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 12-10. Timer Channel Value Register Low (TPMCnVL)

In input capture mode, reading either byte (TPMCnVH or TPMCnVL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. This latching mechanism also resets (becomes unlatched) when the TPMCnSC register is written.

In output compare or PWM modes, writing to either byte (TPMCnVH or TPMCnVL) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the timer channel value registers. This latching mechanism may be manually reset by writing to the TPMCnSC register.

This latching mechanism allows coherent 16-bit writes in either order, which is friendly to various compiler implementations.

12.4 Functional Description

All TPM functions are associated with a main 16-bit counter that allows flexible selection of the clock source and prescale divisor. A 16-bit modulo register also is associated with the main 16-bit counter in the TPM. Each TPM channel is optionally associated with an MCU pin and a maskable interrupt function.

The TPM has center-aligned PWM capabilities controlled by the CPWMS control bit in TPMSC. When CPWMS is set to 1, timer counter TPMCNT changes to an up-/down-counter and all channels in the associated TPM act as center-aligned PWM channels. When CPWMS = 0, each channel can independently be configured to operate in input capture, output compare, or buffered edge-aligned PWM mode.

The following sections describe the main 16-bit counter and each of the timer operating modes (input capture, output compare, edge-aligned PWM, and center-aligned PWM). Because details of pin operation and interrupt activity depend on the operating mode, these topics are covered in the associated mode sections.

12.4.1 Counter

All timer functions are based on the main 16-bit counter (TPMCNTH:TPMCNTL). This section discusses selection of the clock source, up-counting vs. up-/down-counting, end-of-count overflow, and manual counter reset.

After any MCU reset, CLKSB:CLKSA = 0:0 so no clock source is selected and the TPM is inactive. Normally, CLKSB:CLKSA would be set to 0:1 so the bus clock drives the timer counter. The clock source for the TPM can be selected to be off, the bus clock (BUSCLK), the fixed system clock (XCLK), or an external input. The maximum frequency allowed for the external clock option is one-fourth the bus rate. Refer to Section 12.3.1, “Timer Status and Control Register (TPMSC)” and Table 12-3 for more information about clock source selection.

When the microcontroller is in active background mode, the TPM temporarily suspends all counting until the microcontroller returns to normal user operating mode. During stop mode, all TPM clocks are stopped; therefore, the TPM is effectively disabled until clocks resume. During wait mode, the TPM continues to operate normally.

The main 16-bit counter has two counting modes. When center-aligned PWM is selected (CPWMS = 1), the counter operates in up-/down-counting mode. Otherwise, the counter operates as a simple up-counter. As an up-counter, the main 16-bit counter counts from 0x0000 through its terminal count and then continues with 0x0000. The terminal count is 0xFFFF or a modulus value in TPMMODH:TPMMODL.

When center-aligned PWM operation is specified, the counter counts upward from 0x0000 through its terminal count and then counts downward to 0x0000 where it returns to up-counting. Both 0x0000 and the terminal count value (value in TPMMODH:TPMMODL) are normal length counts (one timer clock period long).

An interrupt flag and enable are associated with the main 16-bit counter. The timer overflow flag (TOF) is a software-accessible indication that the timer counter has overflowed. The enable signal selects between software polling (TOIE = 0) where no hardware interrupt is generated, or interrupt-driven operation (TOIE = 1) where a static hardware interrupt is automatically generated whenever the TOF flag is 1.

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the main 16-bit counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF becomes set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to 0x0000. When the main 16-bit counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and the next lower count value. This corresponds to the end of a PWM period. (The 0x0000 count value corresponds to the center of a period.)

Because the HCS08 MCU is an 8-bit architecture, a coherency mechanism is built into the timer counter for read operations. Whenever either byte of the counter is read (TPMCNTH or TPMCNTL), both bytes are captured into a buffer so when the other byte is read, the value will represent the other byte of the count at the time the first byte was read. The counter continues to count normally, but no new value can be read from either byte until both bytes of the old count have been read.

The main timer counter can be reset manually at any time by writing any value to either byte of the timer count TPMCNTH or TPMCNTL. Resetting the counter in this manner also resets the coherency mechanism in case only one byte of the counter was read before resetting the count.

12.4.2 Channel Mode Selection

Provided CPWMS = 0 (center-aligned PWM operation is not specified), the MSnB and MSnA control bits in the channel n status and control registers determine the basic mode of operation for the corresponding channel. Choices include input capture, output compare, and buffered edge-aligned PWM.

12.4.2.1 Input Capture Mode

With the input capture function, the TPM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TPM latches the contents of the TPM counter into the channel value registers (TPMCnVH:TPMCnVL). Rising edges, falling edges, or any edge may be chosen as the active edge that triggers an input capture.

When either byte of the 16-bit capture register is read, both bytes are latched into a buffer to support coherent 16-bit accesses regardless of order. The coherency sequence can be manually reset by writing to the channel status/control register (TPMCnSC).

An input capture event sets a flag bit (CHnF) that can optionally generate a CPU interrupt request.

12.4.2.2 Output Compare Mode

With the output compare function, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter reaches the value in the channel value registers of an output compare channel, the TPM can set, clear, or toggle the channel pin.

In output compare mode, values are transferred to the corresponding timer channel value registers only after both 8-bit bytes of a 16-bit register have been written. This coherency sequence can be manually reset by writing to the channel status/control register (TPMCnSC).

An output compare event sets a flag bit (CHnF) that can optionally generate a CPU interrupt request.

12.4.2.3 Edge-Aligned PWM Mode

This type of PWM output uses the normal up-counting mode of the timer counter (CPWMS = 0) and can be used when other channels in the same TPM are configured for input capture or output compare functions. The period of this PWM signal is determined by the setting in the modulus register (TPMMODH:TPMMODL). The duty cycle is determined by the setting in the timer channel value register (TPMCnVH:TPMCnVL). The polarity of this PWM signal is determined by the setting in the ELSnA control bit. Duty cycle cases of 0 percent and 100 percent are possible.

As Figure 12-11 shows, the output compare value in the TPM channel registers determines the pulse width (duty cycle) of the PWM signal. The time between the modulus overflow and the output compare is the pulse width. If ELSnA = 0, the counter overflow forces the PWM signal high and the output compare forces the PWM signal low. If ELSnA = 1, the counter overflow forces the PWM signal low and the output compare forces the PWM signal high.

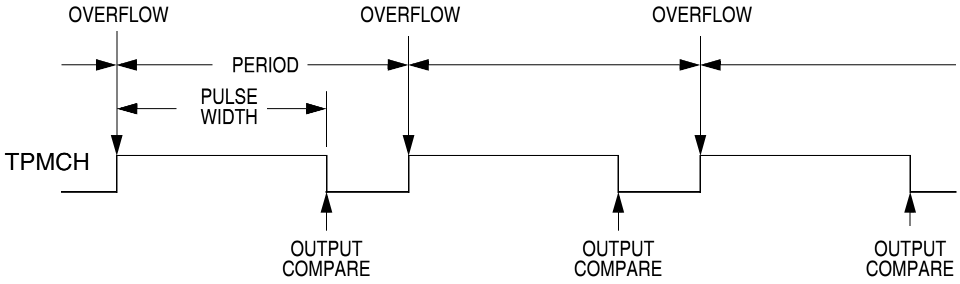


Figure 12-11. PWM Period and Pulse Width (ELSnA = 0)

When the channel value register is set to 0x0000, the duty cycle is 0 percent. By setting the timer channel value register (TPMCnVH:TPMCnVL) to a value greater than the modulus setting, 100% duty cycle can be achieved. This implies that the modulus setting must be less than 0xFFFF to get 100% duty cycle.

Because the HCS08 is a family of 8-bit MCUs, the settings in the timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to either register, TPMCnVH or TPMCnVL, write to buffer registers. In edge-PWM mode, values are transferred to the corresponding timer channel registers only after both 8-bit bytes of a 16-bit register have been written and the value in the TPMCNTH:TPMCNTL counter is 0x0000. (The new duty cycle does not take effect until the next full period.)

12.4.3 Center-Aligned PWM Mode

This type of PWM output uses the up-/down-counting mode of the timer counter (CPWMS = 1). The output compare value in TPMCnVH:TPMCnVL determines the pulse width (duty cycle) of the PWM signal and the period is determined by the value in TPMMODH:TPMMODL. TPMMODH:TPMMODL must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results. ELSnA will determine the polarity of the CPWM output.

pulse width = 2 x (TPMCnVH:TPMCnVL) ***Eqn. 12-1***

period = 2 x (TPMMODH:TPMMODL);
for TPMMODH:TPMMODL = 0x0001–0x7FFF ***Eqn. 12-2***

If the channel value register TPMCnVH:TPMCnVL is zero or negative (bit 15 set), the duty cycle will be 0%. If TPMCnVH:TPMCnVL is a positive value (bit 15 clear) and is greater than the (nonzero) modulus setting, the duty cycle will be 100% because the duty cycle compare will never occur. This implies the usable range of periods set by the modulus register is 0x0001 through 0x7FFE (0x7FFF if generation of 100% duty cycle is not necessary). This is not a significant limitation because the resulting period is much longer than required for normal applications.

TPMMODH:TPMMODL = 0x0000 is a special case that must not be used with center-aligned PWM mode. When CPWMS = 0, this case corresponds to the counter running free from 0x0000 through 0xFFFF, but when CPWMS = 1 the counter needs a valid match to the modulus register somewhere other than at 0x0000 in order to change directions from up-counting to down-counting.

Figure 12-12 shows the output compare value in the TPM channel registers (multiplied by 2), which determines the pulse width (duty cycle) of the CPWM signal. If ELSnA = 0, the compare match while counting up forces the CPWM output signal low and a compare match while counting down forces the output high. The counter counts up until it reaches the modulo setting in TPMMODH:TPMMODL, then counts down until it reaches zero. This sets the period equal to two times TPMMODH:TPMMODL.

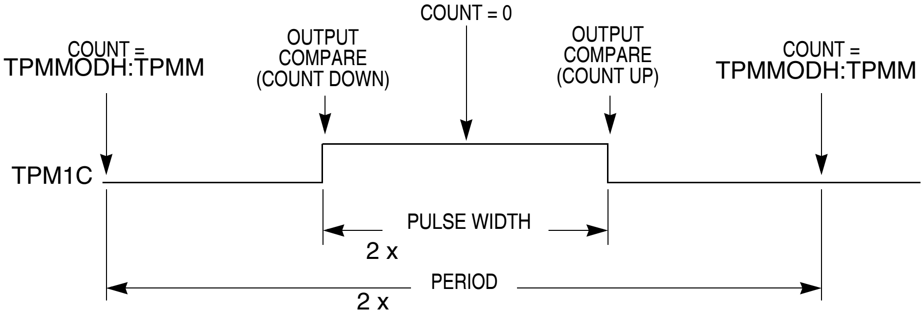


Figure 12-12. CPWM Period and Pulse Width (ELSnA = 0)

Center-aligned PWM outputs typically produce less noise than edge-aligned PWMs because fewer I/O pin transitions are lined up at the same system clock edge. This type of PWM is also required for some types of motor drives.

Because the HCS08 is a family of 8-bit MCUs, the settings in the timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to any of the registers, TPMMODH, TPMMODL, TPMCnVH, and TPMCnVL, actually write to buffer registers. Values are transferred to the corresponding timer channel registers only after both 8-bit bytes of a 16-bit register have been written and the timer counter overflows (reverses direction from up-counting to down-counting at the end of the terminal count in the modulus register). This TPMCNT overflow requirement only applies to PWM channels, not output compares.

Optionally, when TPMCNTH:TPMCNTL = TPMMODH:TPMMODL, the TPM can generate a TOF interrupt at the end of this count. The user can choose to reload any number of the PWM buffers, and they will all update simultaneously at the start of a new period.

Writing to TPMSC cancels any values written to TPMMODH and/or TPMMODL and resets the coherency mechanism for the modulo registers. Writing to TPMCnSC cancels any values written to the channel value registers and resets the coherency mechanism for TPMCnVH:TPMCnVL.

12.5 TPM Interrupts

The TPM generates an optional interrupt for the main counter overflow and an interrupt for each channel. The meaning of channel interrupts depends on the mode of operation for each channel. If the channel is configured for input capture, the interrupt flag is set each time the selected input capture edge is recognized. If the channel is configured for output compare or PWM modes, the interrupt flag is set each time the main timer counter matches the value in the 16-bit channel value register. See the Resets, Interrupts, and System Configuration chapter for absolute interrupt vector addresses, priority, and local interrupt mask control bits.

For each interrupt source in the TPM, a flag bit is set on recognition of the interrupt condition such as timer overflow, channel input capture, or output compare events. This flag may be read (polled) by software to verify that the action has occurred, or an associated enable bit (TOIE or CHnIE) can be set to enable hardware interrupt generation. While the interrupt enable bit is set, a static interrupt will be generated whenever the associated interrupt flag equals 1. It is the responsibility of user software to perform a sequence of steps to clear the interrupt flag before returning from the interrupt service routine.

12.5.1 Clearing Timer Interrupt Flags

TPM interrupt flags are cleared by a 2-step process that includes a read of the flag bit while it is set (1) followed by a write of 0 to the bit. If a new event is detected between these two steps, the sequence is reset and the interrupt flag remains set after the second step to avoid the possibility of missing the new event.

12.5.2 Timer Overflow Interrupt Description

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the 16-bit timer counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF becomes set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to 0x0000. When the counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and the next lower count value. This corresponds to the end of a PWM period. (The 0x0000 count value corresponds to the center of a period.)

12.5.3 Channel Event Interrupt Description

The meaning of channel interrupts depends on the current mode of the channel (input capture, output compare, edge-aligned PWM, or center-aligned PWM).

When a channel is configured as an input capture channel, the ELSnB:ELSnA control bits select rising edges, falling edges, any edge, or no edge (off) as the edge that triggers an input capture event. When the selected edge is detected, the interrupt flag is set. The flag is cleared by the 2-step sequence described in Section 12.5.1, “Clearing Timer Interrupt Flags.”

When a channel is configured as an output compare channel, the interrupt flag is set each time the main timer counter matches the 16-bit value in the channel value register. The flag is cleared by the 2-step sequence described in Section 12.5.1, “Clearing Timer Interrupt Flags.”

12.5.4 PWM End-of-Duty-Cycle Events

For channels that are configured for PWM operation, there are two possibilities:

- When the channel is configured for edge-aligned PWM, the channel flag is set when the timer counter matches the channel value register that marks the end of the active duty cycle period.
- When the channel is configured for center-aligned PWM, the timer count matches the channel value register twice during each PWM cycle. In this CPWM case, the channel flag is set at the start and at the end of the active duty cycle, which are the times when the timer counter matches the channel value register.

The flag is cleared by the 2-step sequence described in Section 12.5.1, “Clearing Timer Interrupt Flags.”

Chapter 13
Development Support

13.1 Introduction

Development support systems in the RS08 family include the RS08 background debug controller (BDC).

The BDC provides a single-wire debug interface to the target MCU. This interface provides a convenient means for programming the on-chip FLASH and other nonvolatile memories. Also, the BDC is the primary debug interface for development and allows non-intrusive access to memory data and traditional debug features such as CPU register modify, breakpoint, and single-instruction trace commands.

In the RS08 Family, address and data bus signals are not available on external pins. Debug is done through commands fed into the target MCU via the single-wire background debug interface, including resetting the device without using a reset pin.

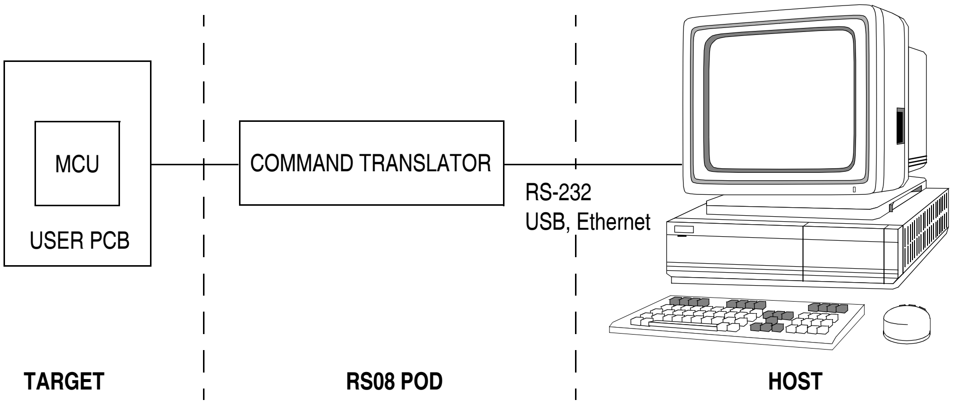


Figure 13-1. Connecting MCU to Host for Debugging

13.2 Features

Features of the RS08 background debug controller (BDC) include:

- Uses a single pin for background debug serial communications
- Non-intrusive of user memory resources; BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands allow access to memory resources while CPU is running user code without stopping applications
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from wait or stop modes

- BDC_RESET command allows host to reset MCU without using a reset pin
- One hardware address breakpoint built into BDC
- RS08 clock source runs in stop mode if BDM enabled to allow debugging when CPU is in stop mode
- COP watchdog suspended while in active background mode

13.3 RS08 Background Debug Controller (BDC)

All MCUs in the RS08 Family contain a single-wire background debug interface which supports in-circuit programming of on-chip non-volatile memory and sophisticated debug capabilities. Unlike debug interfaces on earlier 8-bit MCUs, this debug system provides for minimal interference with normal application resources. It does not use any user memory or locations in the memory map. It requires use of only the output-only BKGD pin. This pin will be shared with simple user output-only functions (typically port, comparator outputs, etc.), which can be easily debugged in normal user mode.

RS08 BDM commands are divided into two groups:

- Active background mode commands require that the target MCU is in active background mode (the user program is not running). The BACKGROUND command causes the target MCU to enter active background mode. Active background mode commands allow the CPU registers to be read or written and allow the user to trace one (TRACE1) user instruction at a time or GO to the user program from active background mode.
- Non-intrusive commands can be executed at any time even while the user program is running. Non-intrusive commands allow a user to read or write MCU memory locations or access status and control registers within the background debug controller (BDC).

Typically, a relatively simple interface pod is used to translate commands from a host computer into commands for the custom serial interface to the single-wire background debug system. Depending on the development tool vendor, this interface pod may use a standard RS-232 serial port, a parallel printer port, or some other type of communication such as Ethernet or a universal serial bus (USB) to communicate between the host PC and the pod.

Figure 13-2 shows the standard header for connection of a RS08 BDM pod. A pod is a small interface device that connects a host computer such as a personal computer to a target RS08 system. BKGD and GND are the minimum connections required to communicate with a target MCU. The pseudo-open-drain $\overline{\text{RESET}}$ signal is included in the connector to allow a direct hardware method for the host to force or monitor (if $\overline{\text{RESET}}$ is available as output) a target system reset.

The RS08 BDM pods supply the V_{PP} voltage to the RS08 MCU when in-circuit programming is required. The V_{PP} connection from the pod is shared with $\overline{\text{RESET}}$ as shown in Figure 13-2. For V_{PP} requirements see the FLASH specifications in the electricals appendix.

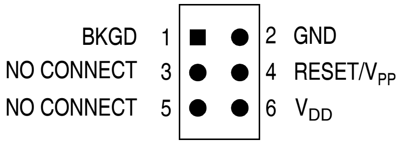


Figure 13-2. Standard RS08 BDM Tool Connector

Background debug controller (BDC) serial communications use a custom serial protocol that was first introduced on the M68HC12 Family of microcontrollers. This protocol requires that the host knows the communication clock rate, which is determined by the target BDC clock rate. If a host is attempting to communicate with a target MCU that has an unknown BDC clock rate, a SYNC command may be sent to the target MCU to request a timed sync response signal from which the host can determine the correct communication speed.

For RS08 MCUs, the BDC clock is the same frequency as the MCU bus clock. For a detailed description of the communications protocol, refer to Section 13.3.2, “Communication Details."

13.3.1 BKGD Pin Description

BKGD is the single-wire background debug interface pin. BKGD is a pseudo-open-drain pin that contains an on-chip pullup, therefore it requires no external pullup resistor. Unlike typical open-drain pins, the external resistor capacitor (RC) time constant on this pin, which is influenced by external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speedup pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 13.3.2, “Communication Details," for more detail.

The primary function of this pin is bidirectional serial communication of background debug commands and data. During reset, this pin selects between starting in active background mode and normal user mode running an application program. This pin is also used to request a timed sync response pulse to allow a host development tool to determine the target BDC clock frequency.

By controlling the BKGD pin and forcing an MCU reset (issuing a BDC_RESET command, or through a power-on reset (POR)), the host can force the target system to reset into active background mode rather than start the user application program. This is useful to gain control of a target MCU whose FLASH program memory has not yet been programmed with a user application program.

When no debugger pod is connected to the 6-pin BDM interface connector, the internal pullup on BKGD determines the normal operating mode.

On some RS08 devices, the BKGD pin may be shared with an alternative output-only function. To support BDM debugging, the user must disable this alternative function. Debugging of the alternative function must be done in normal user mode without using BDM.

13.3.2 Communication Details

The BDC serial interface requires the host to generate a falling edge on the BKGD pin to indicate the start of each bit time. The host provides this falling edge whether data is transmitted or received.

The BDC serial communication protocol requires the host to know the target BDC clock speed. Commands and data are sent most significant bit first (MSB-first) at 16 BDC clock cycles per bit. The interface times out if 512 BDC clock cycles occur between falling edges from the host. Any BDC command that was in progress when this timeout occurs is aborted without affecting the memory or operating mode of the target MCU system.

Figure 13-3 shows an external host transmitting a logic 1 or 0 to the BKGD pin of a target MCU. The host is asynchronous to the target so there is a 0-to-1 cycle delay from the host-generated falling edge to where the target perceives the beginning of the bit time. Ten target BDC clock cycles later, the target senses the bit level on the BKGD pin. Typically, the host actively drives the pseudo-open-drain BKGD pin during host-to-target transmissions to speed up rising edges. Because the target does not drive the BKGD pin during the host-to-target period, there is no need to treat the line as an open-drain signal during this period.

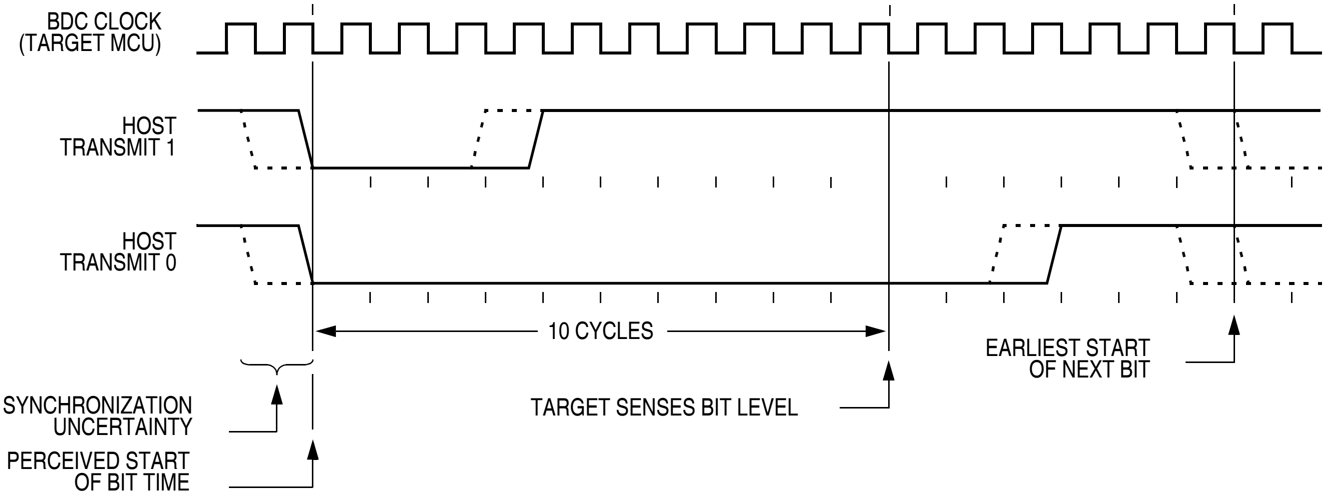


Figure 13-3. BDC Host-to-Target Serial Bit Timing

Figure 13-4 shows the host receiving a logic 1 from the target MCU. Because the host is asynchronous to the target, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target. The host holds the BKGD pin low long enough for the target to recognize it (at least two target BDC cycles). The host must release the low drive before the target drives a brief active-high speedup pulse seven cycles after the perceived start of the bit time. The host must sample the bit level approximately 10 cycles after it started the bit time.

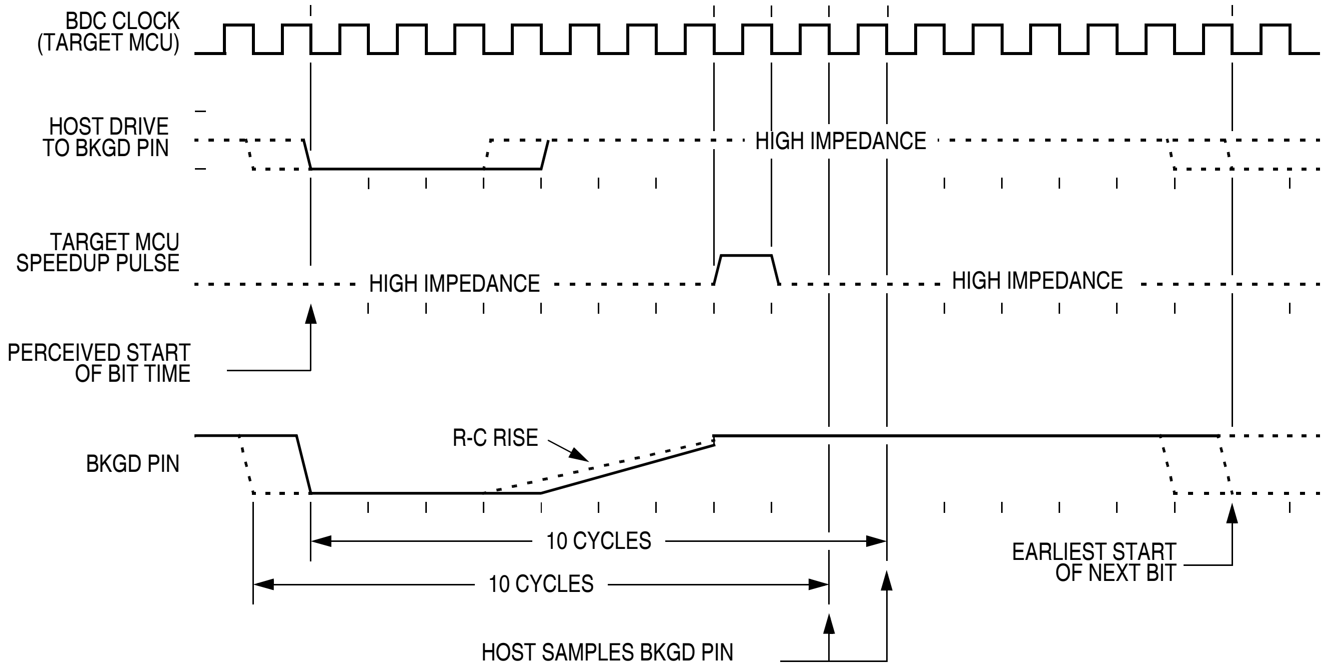


Figure 13-4. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 13-5 shows the host receiving a logic 0 from the target MCU. Because the host is asynchronous to the target, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target. The host initiates the bit time but the target finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 BDC clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level approximately 10 cycles after starting the bit time.

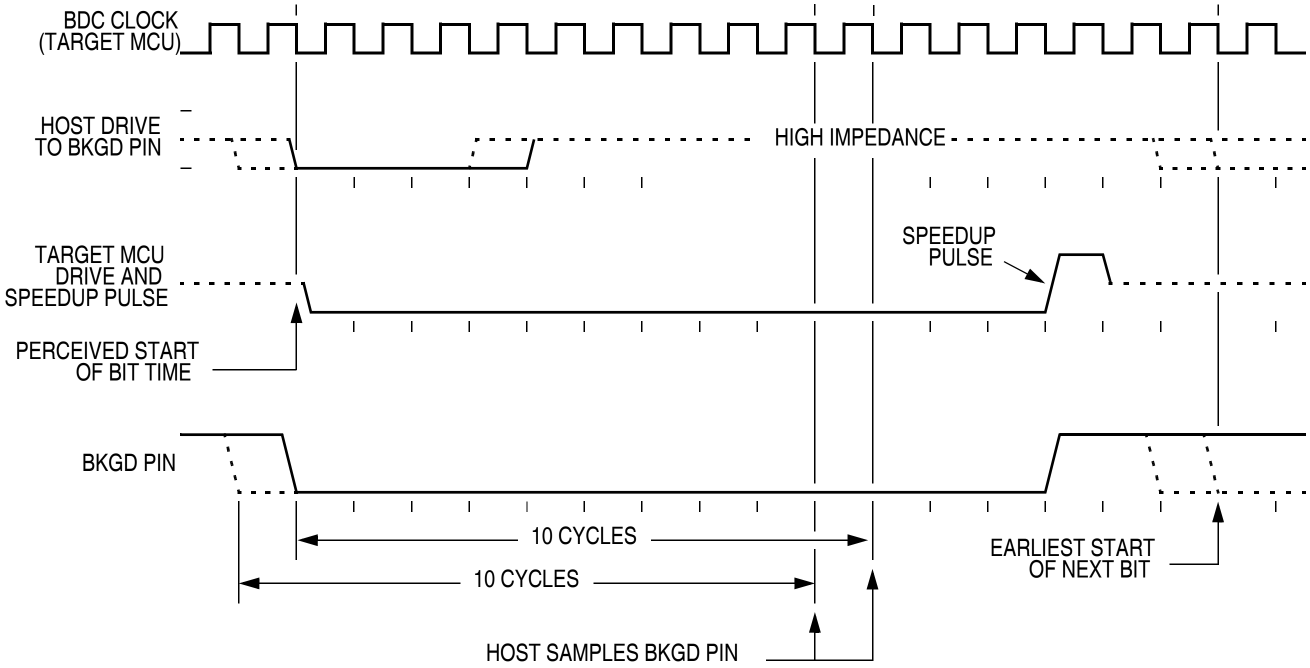


Figure 13-5. BDM Target-to-Host Serial Bit Timing (Logic 0)

13.3.3 SYNC and Serial Communication Timeout

The host initiates a host-to-target serial transmission by generating a falling edge on the BKGD pin. If BKGD is kept low for more than 128 target clock cycles, the target understands that a SYNC command was issued. In this case, the target will keep waiting for a rising edge on BKGD to answer the SYNC request pulse. If the rising edge is not detected, the target will keep waiting indefinitely, without any timeout limit. When a rising edge on BKGD occurs after a valid SYNC request, the BDC will drive the BKGD pin low for exactly 128 BDC cycles.

Consider now the case where the host returns BKGD to logic 1 before 128 cycles. This is interpreted as a valid bit transmission, and not as a SYNC request. The target will keep waiting for another falling edge marking the start of a new bit. If, however, a new falling edge is not detected by the target within 512 clock cycles since the last falling edge, a timeout occurs and the current command is discarded without affecting memory or the operating mode of the MCU. This is referred as a soft-reset to the BDC.

If a read command is issued but the data is not retrieved within 512 serial clock cycles, a soft-reset will occur causing the command to be disregarded. The data is not available for retrieving after the timeout has occurred. A soft-reset is also used to end a READ_BLOCK or WRITE_BLOCK command.

The following describes the actual bit-time requirements for a host to guarantee logic 1 or 0 bit transmission without the target timing out or interpreting the bit as a SYNC command:

- To send a logic 0, BKGD must be kept low for a minimum of 12 BDC cycles and up to 511 BDC cycles except for the first bit of a command sequence, which will be detected as a SYNC request.
- To send a logic 1, BKGD must be held low for at least four BDC cycles, be released by the eighth cycle, and be held high until at least the sixteenth BDC cycle.

- Subsequent bits must occur within 512 BDC cycles of the last bit sent.

13.4 BDC Registers and Control Bits

The BDC contains two non-CPU accessible registers:

- The BDC status and control register (BDCSCR) is an 8-bit register containing control and status bits for the background debug controller.
- The BDC breakpoint register (BDCBKPT) holds a 16-bit breakpoint match address.

These registers are accessed with dedicated serial BDC commands and are not located in the memory space of the target MCU (so they do not have addresses and cannot be accessed by user programs).

Some of the bits in the BDCSCR have write limitations; otherwise, these registers may be read or written at any time. For example, the ENBDM control bit may not be written while the MCU is in active background mode. This prevents the ambiguous condition of the control bit forbidding active background mode while the MCU is already in active background mode. Also, the status bits (BDMACT, WS, and WSF) are read-only status indicators and can never be written by the WRITE_CONTROL serial BDC command.

13.4.1 BDC Status and Control Register (BDCSCR)

This register can be read or written by serial BDC commands (READ_STATUS and WRITE_CONTROL) but is not accessible to user programs because it is not located in the normal memory map of the MCU.

	7	6	5	4	3	2	1	0
R	ENBDM	BDMACT	BKPTEN	FTS	0	WS	WSF	0
W		▨			▨	▨	▨	▨
Normal Reset	0	0	0	0	0	0	0	0
Reset in Active BDM:	1	1	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 13-6. BDC Status and Control Register (BDCSCR)

Table 13-1. BDCSCR Register Field Descriptions

Field	Description
7 ENBDM	**Enable BDM (Permit Active Background Mode)** — Typically, this bit is written to 1 by the debug host shortly after the beginning of a debug session or whenever the debug host resets the target and remains 1 until a normal reset clears it. If the application can go into stop mode, this bit is required to be set if debugging capabilities are required. 0 BDM cannot be made active (non-intrusive commands still allowed). 1 BDM can be made active to allow active background mode commands.
6 BDMACT	**Background Mode Active Status** — This is a read-only status bit. 0 BDM not active (user application program running). 1 BDM active and waiting for serial commands.

Table 13-1. BDCSCR Register Field Descriptions (continued)

Field	Description
5 BKPTEN	**BDC Breakpoint Enable** — If this bit is clear, the BDC breakpoint is disabled and the FTS (force tag select) control bit and BDCBKPT match register are ignored 0 BDC breakpoint disabled. 1 BDC breakpoint enabled.
4 FTS	**Force/Tag Select** — When FTS = 1, a breakpoint is requested whenever the CPU address bus matches the BDCBKPT match register. When FTS = 0, a match between the CPU address bus and the BDCBKPT register causes the fetched opcode to be tagged. If this tagged opcode ever reaches the end of the instruction queue, the CPU enters active background mode rather than executing the tagged opcode. 0 Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction. 1 Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode).
2 WS	**Wait or Stop Status** — When the target CPU is in wait or stop mode, most BDC commands cannot function. However, the BACKGROUND command can be used to force the target CPU out of wait or stop and into active background mode where all BDC commands work. Whenever the host forces the target MCU into active background mode, the host must issue a READ_STATUS command to check that BDMACT = 1 before attempting other BDC commands. 0 Target CPU is running user application code or in active background mode (was not in wait or stop mode when background became active). 1 Target CPU is in wait or stop mode, or a BACKGROUND command was used to change from wait or stop to active background mode.
1 WSF	**Wait or Stop Failure Status** — This status bit is set if a memory access command failed due to the target CPU executing a wait or stop instruction at or about the same time. The usual recovery strategy is to issue a BACKGROUND command to get out of wait or stop mode into active background mode, repeat the command that failed, then return to the user program. (Typically, the host would restore CPU registers and stack values and re-execute the wait or stop instruction.) 0 Memory access did not conflict with a wait or stop instruction. 1 Memory access command failed because the CPU entered wait or stop mode.

13.4.2 BDC Breakpoint Match Register

This 16-bit register holds the 14-bit address for the hardware breakpoint in the BDC. The BKPTEN and FTS control bits in BDCSCR are used to enable and configure the breakpoint logic. Dedicated serial BDC commands (READ_BKPT and WRITE_BKPT) are used to read and write the BDCBKPT register. Breakpoints are normally set while the target MCU is in active background mode before running the user application program. However, because READ_BKPT and WRITE_BKPT are non-intrusive commands, they could be executed even while the user program is running. For additional information about setup and use of the hardware breakpoint logic in the BDC, refer to the RS08 Family Reference Manual.”

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
W	▨	▨														
Any Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 13-7. BDC Breakpoint Match Register (BDCBKPT)

13.5 RS08 BDC Commands

BDC commands are sent serially from a host computer to the BKGD pin of the target MCU. All commands and data are sent MSB-first using a custom BDC communications protocol. Active background mode commands require that the target MCU is currently in the active background mode while non-intrusive commands may be issued at any time whether the target MCU is in active background mode or running a user application program.

Table 13-2 shows all RS08 BDC commands, a shorthand description of their coding structure, and the meaning of each command.

Coding Structure Nomenclature

The following nomenclature is used in Table 13-2 to describe the coding structure of the BDC commands.

Commands begin with an 8-bit command code in the host-to-target direction (most significant bit first)

/ = Separates parts of the command

d = Delay 16 to 511 target BDC clock cycles

soft-reset = Delay of at least 512 BDC clock cycles from last host falling-edge

AAAA = 16-bit address in the host-to-target direction[1]

RD = Eight bits of read data in the target-to-host direction

WD = Eight bits of write data in the host-to-target direction

RD16 = 16 bits of read data in the target-to-host direction

WD16 = 16 bits of write data in the host-to-target direction

SS = the contents of BDCSCR in the target-to-host direction (STATUS)

CC = Eight bits of write data for BDCSCR in the host-to-target direction (CONTROL)

RBKP = 16 bits of read data in the target-to-host direction (from BDCBKPT breakpoint register)

WBKP = 16 bits of write data in the host-to-target direction (for BDCBKPT breakpoint register)

---

[1] The RS08 CPU uses only 14 bits of address and occupies the lower 14 bits of the 16-bit AAAA address field. The values of address bits 15 and 14 in AAAA are truncated and thus do not matter.

Table 13-2. RS08 BDC Command Summary

Command Mnemonic	Active Background Mode/ Non-Intrusive	Coding Structure	Description
SYNC	Non-intrusive	n/a[1]	Request a timed reference pulse to determine target BDC communication speed
BDC_RESET	Any CPU mode	18[2]	Request an MCU reset
BACKGROUND	Non-intrusive	90/d	Enter active background mode if enabled (ignore if ENBDM bit equals 0)
READ_STATUS	Non-intrusive	E4/SS	Read BDC status from BDCSCR
WRITE_CONTROL	Non-intrusive	C4/CC	Write BDC controls in BDCSCR
READ_BYTE	Non-intrusive	E0/AAAA/d/RD	Read a byte from target memory
READ_BYTE_WS	Non-intrusive	E1/AAAA/d/SS/RD	Read a byte and report status
WRITE_BYTE	Non-intrusive	C0/AAAA/WD/d	Write a byte to target memory
WRITE_BYTE_WS	Non-intrusive	C1/AAAA/WD/d/SS	Write a byte and report status
READ_BKPT	Non-intrusive	E2/RBKP	Read BDCBKPT breakpoint register
WRITE_BKPT	Non-intrusive	C2/WBKP	Write BDCBKPT breakpoint register
GO	Active background mode	08/d	Go to execute the user application program starting at the address currently in the PC
TRACE1	Active background mode	10/d	Trace one user instruction at the address in the PC, then return to active background mode
READ_BLOCK	Active background mode	80/AAAA/d/RD[3]	Read a block of data from target memory starting from AAAA continuing until a soft-reset is detected
WRITE_BLOCK	Active background mode	88/AAAA/WD/d[4]	Write a block of data to target memory starting at AAAA continuing until a soft-reset is detected
READ_A	Active background mode	68/d/RD	Read accumulator (A)

Table 13-2. RS08 BDC Command Summary (continued)

Command Mnemonic	Active Background Mode/ Non-Intrusive	Coding Structure	Description
WRITE_A	Active background mode	48/WD/d	Write accumulator (A)
READ_CCR_PC	Active background mode	6B/d/RD16[5]	Read the CCR bits z, c concatenated with the 14-bit program counter (PC) RD16=zc:PC
WRITE_CCR_PC	Active background mode	4B/WD16/d[6]	Write the CCR bits z, c concatenated with the 14-bit program counter (PC) WD16=zc:PC
READ_SPC	Active background mode	6F/d/RD16[7]	Read the 14-bit shadow program counter (SPC) RD16=0:0:SPC
WRITE_SPC	Active background mode	4F/WD16/d[8]	Write 14-bit shadow program counter (SPC) WD16 = x:x:SPC, the two most significant bits shown by “x” are ignored by target

[1] The SYNC command is a special operation which does not have a command code.

[2] 18 was HCS08 BDC command for TAGGO.

[3] Each RD requires a delay between host read data byte and next read, command ends when target detects a soft-reset.

[4] Each WD requires a delay between host write data byte and next byte, command ends when target detects a soft-reset.

[5] HCS08 BDC had separate READ_CCR and READ_PC commands, the RS08 BDC combined this commands.

[6] HCS08 BDC had separate WRITE_CCR and WRITE_PC commands, the RS08 BDC combined this commands.

[7] 6F is READ_SP (read stack pointer) for HCS08 BDC.

[8] 4F is WRITE_SP (write stack pointer) for HCS08 BDC.

Appendix A
Electrical Characteristics

A.1 Introduction

This chapter contains electrical and timing specifications for the SC9RS08MZ8 microcontroller available at the time of publication.

A.2 Parameter Classification

The electrical parameters shown in this supplement are guaranteed by various methods. To give the customer a better understanding the following classification is used and the parameters are tagged accordingly in the tables where appropriate:

Table A-1. Parameter Classifications

P	Those parameters are guaranteed during production testing on each individual device.
C	Those parameters are achieved by the design characterization by measuring a statistically relevant sample size across process variations.
T	Those parameters are achieved by design characterization on a small sample size from typical devices under typical conditions unless otherwise noted. All values shown in the typical column are within this category.
D	Those parameters are derived mainly from simulations.

NOTE

The classification is shown in the column labeled “C” in the parameter tables where appropriate.

A.3 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only, and functional operation at the maxima is not guaranteed. Stress beyond the limits specified in Table A-2 may affect device reliability or cause permanent damage to the device. For functional operating conditions, refer to the remaining tables in this chapter.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (for instance, V_{SS} or V_{DD}) or the programmable pull-up resistor associated with the pin is enabled.

Table A-2. Absolute Maximum Ratings

Rating	Symbol	Value	Unit
Supply voltage	V_{DD}	–0.3 to 5.8	V
Maximum current into V_{DD}	I_{DD}	120	mA
Digital input voltage	V_{In}	–0.3 to V_{DD} + 0.3	V
Instantaneous maximum current Single pin limit (applies to all port pins)[1, 2, 3]	I_D	±25	mA
Storage temperature range	T_{stg}	–55 to 150	°C

[1] Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive (V_{DD}) and negative (V_{SS}) clamp voltages, then use the larger of the two resistance values.

[2] All functional non-supply pins are internally clamped to V_{SS} and V_{DD} except the $\overline{\text{RESET}}/V_{PP}$ pin which is internally clamped to V_{SS} only.

[3] Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{In} > V_{DD}$) is greater than I_{DD}, the injection current may flow out of V_{DD} and could result in external power supply going out of regulation. Ensure external V_{DD} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power. Examples are: if no system clock is present, or if the clock rate is very low which would reduce overall power consumption.

A.4 Thermal Characteristics

This section provides information about operating temperature range, power dissipation, and package thermal resistance. Power dissipation on I/O pins is usually small compared to the power dissipation in on-chip logic and voltage regulator circuits and it is user-determined rather than being controlled by the MCU design. In order to take $P_{I/O}$ into account in power calculations, determine the difference between actual pin voltage and V_{SS} or V_{DD} and multiply by the pin current for each I/O pin. Except in cases of unusually high pin current (heavy loads), the difference between pin voltage and V_{SS} or V_{DD} will be very small.

Table A-3. Thermal Characteristics

Rating	Symbol	Value	Unit
Operating temperature range (packaged)	T_A	T_L to T_H –40 to 85	°C
Maximum junction temperature	T_{JMAX}	105	°C
Thermal resistance 20-pin SOIC	θ_{JA}	96	°C/W

The average chip-junction temperature (TJ) in °C can be obtained from:

$$T_J = T_A + (P_D \times \theta_{JA}) \quad \textit{Eqn. A-1}$$

where:

T_A = Ambient temperature, °C

θ_{JA} = Package thermal resistance, junction-to-ambient, °C /W

$P_D = P_{int} + P_{I/O}$

$P_{int} = I_{DD} \times V_{DD}$, Watts chip internal power

$P_{I/O}$ = Power dissipation on input and output pins user determined

For most applications, $P_{I/O} << P_{int}$ and can be neglected. An approximate relationship between PD and TJ (if $P_{I/O}$ is neglected) is:

$$P_D = K \div (T_J + 273°C) \qquad \textit{Eqn. A-2}$$

Solving Equation A-1 and Equation A-2 for K gives:

$$K = P_D \times (T_A + 273°C) + \theta_{JA} \times (P_D)^2 \qquad \textit{Eqn. A-3}$$

where K is a constant pertaining to the particular part. K can be determined from Equation A-3 by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations 1 and 2 iteratively for any value of T_A.

A.5 ESD Protection and Latch-Up Immunity

Although damage from electrostatic discharge (ESD) is much less common on these devices than on early CMOS circuits, normal handling precautions must be used to avoid exposure to static discharge. Qualification tests are performed to ensure that these devices can withstand exposure to reasonable levels of static without suffering any permanent damage.

All ESD testing is in conformity with AEC-Q100 Stress Test Qualification for Automotive Grade Integrated Circuits. During the device qualification ESD stresses were performed for the human body model (HBM), the machine model (MM) and the charge device model (CDM).

A device is defined as a failure if after exposure to ESD pulses the device no longer meets the device specification. Complete DC parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-4. ESD and Latch-up Test Conditions

Model	Description	Symbol	Value	Unit
Human Body	Series resistance	R1	1500	Ω
	Storage capacitance	C	100	pF
	Number of pulses per pin	—	3	—
Machine	Series resistance	R1	0	Ω
	Storage capacitance	C	200	pF
	Number of pulses per pin	—	3	—
Latch-up	Minimum input voltage limit	—	–2.5	V
	Maximum input voltage limit	—	7.5	V

Table A-5. ESD and Latch-Up Protection Characteristics

No.	Rating[1]	Symbol	Min	Max	Unit
1	Human body model (HBM)	V_{HBM}	±2000	—	V
2	Machine model (MM)	V_{MM}	±200	—	V
3	Charge device model (CDM)	V_{CDM}	±500	—	V
4	Latch-up current at T_A = 85°C (applies to all pins except pin 9 PTC3/ADP11)	I_{LAT}	±100[2]	—	mA
	Latch-up current at T_A = 85°C (applies to pin 9 PTC3/ADP11)	I_{LAT}	±75[3]	—	mA

[1] Parameter is achieved by design characterization on a small sample size from typical devices under typical conditions unless otherwise noted.

[2] These pins meet JESD78A Class II (section 1.2) Level A (section 1.3) requirement of ±100 mA.

[3] This pin meets JESD78A Class II (section 1.2) Level B (section 1.3) characterization to ±75 mA.

A.6 DC Characteristics

This section includes information about power supply requirements, I/O pin characteristics, and power supply current in various operating modes.

Table A-6. DC Characteristics (Temperature Range = –40 to 85°C Ambient)

Parameter	Symbol	Min	Typical	Max	Unit
Supply voltage (run, wait and stop modes.) $0 < f_{Bus} < 10$MHz	V_{DD}	2.7	—	5.5	V
Minimum RAM retention supply voltage applied to V_{DD}	V_{RAM}	0.8[1]	—	—	V
Low-voltage Detection threshold (V_{DD} falling) (V_{DD} rising)	V_{LVD}	— —	1.86 1.94	— —	V
Power on RESET (POR) voltage	V_{POR}[1]	0.9	—	1.7	V
Input high voltage (V_{DD} > 2.7V) (all digital inputs)	V_{IH}	0.70 × V_{DD}	—	—	V
Input low voltage (V_{DD} > 2.7 V) (all digital inputs)	V_{IL}	—	—	0.30 × V_{DD}	V
Input hysteresis (all digital inputs)	V_{hys}[1]	0.06 × V_{DD}	—	—	V
Input leakage current (per pin) $V_{In} = V_{DD}$ or V_{SS}, all input only pins	$\lvert I_{In}\rvert$	—	0.025	—	µA
High impedance (off-state) leakage current (per pin) $V_{In} = V_{DD}$ or V_{SS}, all input/output	$\lvert I_{OZ}\rvert$	—	0.025	—	µA
Internal pullup resistors[2](all port pins)	R_{PU}	—	45	—	kΩ
Internal pulldown resistors[2](all port pins except PTA5)	R_{PD}	—	45	—	kΩ
PTA5 Internal pulldown resistor	—	45	—	95	kΩ
Output high voltage — Low Drive (PTxDSn = 0) 5 V, I_{Load} = 2 mA	V_{OH}	V_{DD} – 0.8	—	—	V
Output high voltage — High Drive (PTxDSn = 1) 5 V, I_{Load} = 5 mA		V_{DD} – 0.8	—	—	
Maximum total I_{OH} for all port pins	$\lvert I_{OHT}\rvert$	—	—	40	mA

Table A-6. DC Characteristics (Temperature Range = –40 to 85°C Ambient) (continued)

Parameter	Symbol	Min	Typical	Max	Unit
Output low voltage — Low Drive (PTxDSn = 0) 5 V, I_{Load} = 2 mA	V_{OL}	—	—	0.8	V
Output low voltage — High Drive (PTxDSn = 1) 5 V, I_{Load} = 5 mA		—	—	0.8	
Maximum total I_{OL} for all port pins	I_{OLT}	—	—	40	mA
DC injection current[3, 4, 5 ,6] $V_{In} < V_{SS}$, $V_{In} > V_{DD}$ Single pin limit Total MCU limit, includes sum of all stressed pins		— —	— —	0.2 0.8	mA mA
Input capacitance (all non-supply pins)	C_{In}	—	—	7	pF

[1] This parameter is characterized and not tested on each device.

[2] Measurement condition for pull resistors: $V_{In} = V_{SS}$ for pullup and $V_{In} = V_{DD}$ for pulldown.

[3] All functional non-supply pins are internally clamped to V_{SS} and V_{DD} except the $\overline{RESET}/V_{PP}$ which is internally clamped to V_{SS} only.

[4] Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.

[5] Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.

[6] This parameter is characterized and not tested on each device.

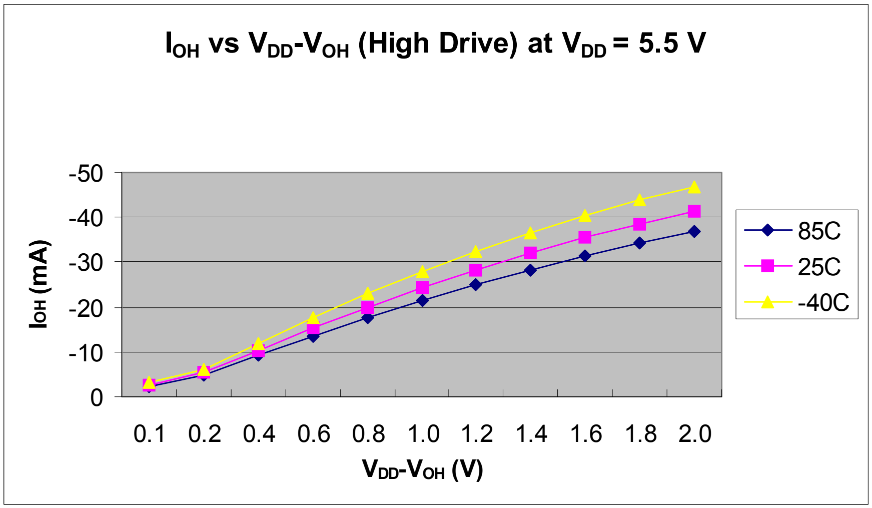


Figure A-1. Typical I_{OH} vs. V_{DD}–V_{OH}
V_{DD} = 5.5 V (High Drive)

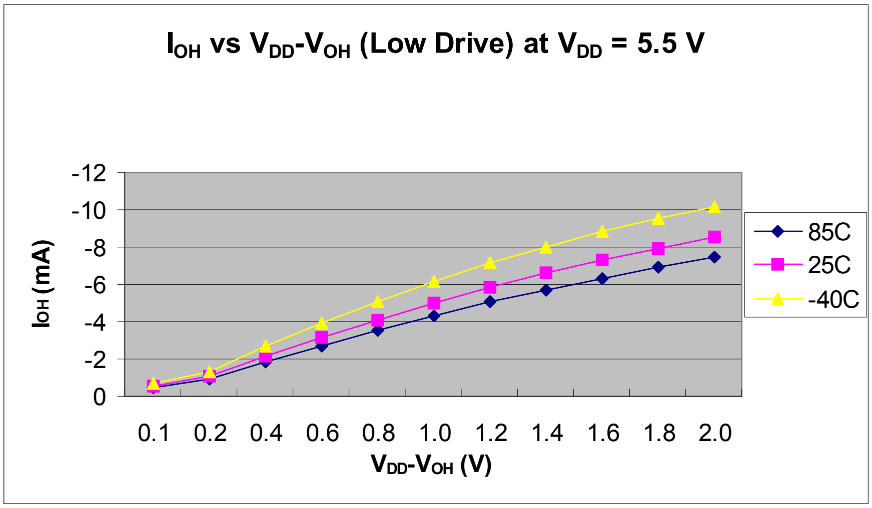


Figure A-2. Typical I_{OH} vs. V_{DD}–V_{OH} V_{DD} = 5.5 V (Low Drive)

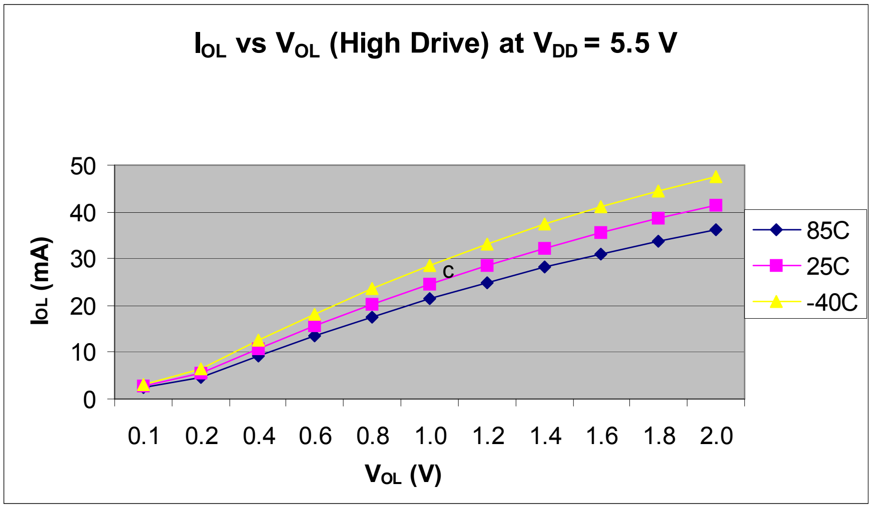


Figure A-3. Typical I_{OL} vs. V_{DD}–V_{OL} V_{DD} = 5.5 V (High Drive)

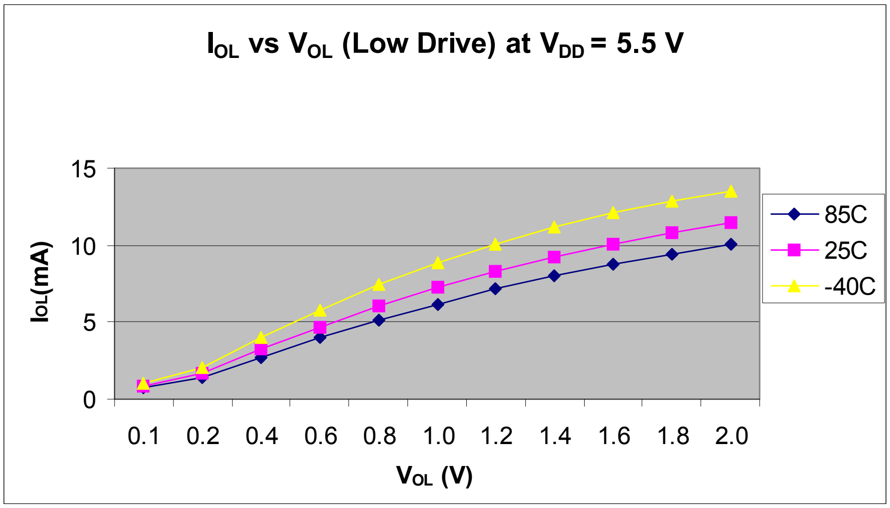


Figure A-4. Typical I_{OL} vs. V_{DD}–V_{OL}
V_{DD} = 5.5 V (Low Drive)

A.7 Supply Current Characteristics

Table A-7. Supply Current Characteristics

Parameter	Symbol	V_{DD} (V)	Typical[1]	Temp. (°C)
Run supply current[2] measured at (f_{Bus} = 10 MHz)	RIDD10	5	2.4 mA	25 85
Run supply current[3] measured at (fBus = 1.25 MHz)	RIDD1	5	0.42 mA	25 85
Stop mode supply current	SIDD	5	2.4 µA	25 85
ADC adder from stop[3]		5	128 µA	25 85
ACMP adder from stop (ACME = 1)		5	21 µA	25 85
RTI adder from stop with 1 kHz clock source enabled[4]		5	2.4 µA	25 85
RTI adder from stop with 1 MHz external clock source reference enabled		5	2.1 µA	25 85
LVI adder from stop (LVDE=1 and LVDSE=1)		5	70 µA	25 85

[1] Typicals are measured at 25 °C.

[2] Not include any DC loads on port pins.

[3] Required asynchronous ADC clock and LVD to be enabled.

[4] Most customers are expected to find that auto-wakeup from stop can be used instead of the higher current wait mode. Wait mode typical is 1.3 mA at 3 V and 1 mA at 2 V with f_{Bus} = 1 MHz.

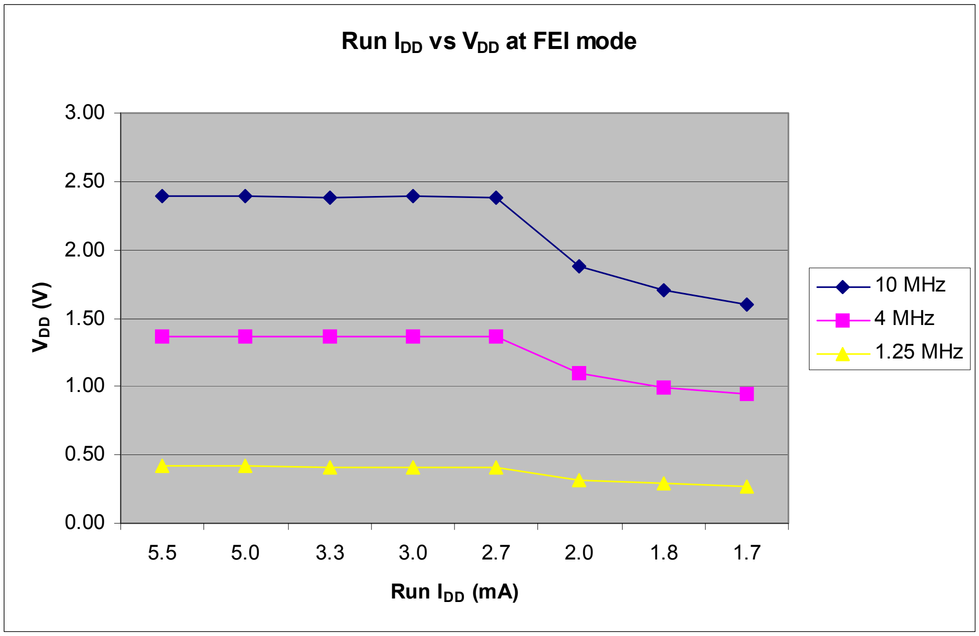


Figure A-5. Typical Run I_{DD} vs. V_{DD} for FEI Mode

A.8 External Oscillator (XOSC) Characteristics

Table A-8. Oscillator Electrical Specifications (Temperature Range = –40 to 85°C Ambient)

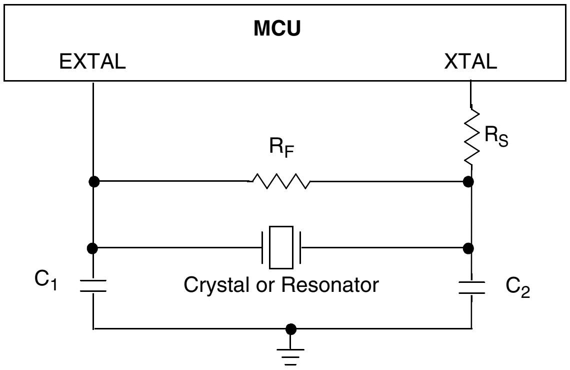
Num	C	Rating	Symbol	Min	Typical[1]	Max	Unit
1	C	Oscillator crystal or resonator (EREFS = 1, ERCLKEN = 1)					
		Low range (RANGE = 0)	f_{lo}	32	—	38.4	kHz
		High range (RANGE = 1) FEE or FBE mode [2]	f_{hi}	1	—	5	MHz
		High range (RANGE = 1, HGO = 1) FBELP mode	$f_{hi\text{-}hgo}$	1	—	16	MHz
		High range (RANGE = 1, HGO = 0) FBELP mode	$f_{hi\text{-}lp}$	1	—	8	MHz
2	D	Load capacitors	C_1, C_2	See crystal or resonator manufacturer's recommendation.			
3	D	Feedback resistor	R_F				MΩ
		Low range (32 kHz to 100 kHz)		—	10	—	
		High range (1 MHz to 16 MHz)		—	1	—	
4	D	Series resistor	R_S				kΩ
		Low range, low gain (RANGE = 0, HGO = 0)		—	0	—	
		Low range, high gain (RANGE = 0, HGO = 1)		—	100	—	
		High range, low gain (RANGE = 1, HGO = 0)		—	0	—	
		High range, high gain (RANGE = 1, HGO = 1)					
		≥ 8 MHz		—	0	—	
		4 MHz		—	0	—	
		1 MHz		—	0	—	
5	C	Crystal start-up time [3]					ms
		Low range, low gain (RANGE = 0, HGO = 0)	$t_{CSTL\text{-}LP}$	—	200	—	
		Low range, high gain (RANGE = 0, HGO = 1)	$t_{CSTL\text{-}HGO}$	—	400	—	
		High range, low gain (RANGE = 1, HGO = 0)[4]	$t_{CSTH\text{-}LP}$	—	5	—	
		High range, high gain (RANGE = 1, HGO = 1)[4]	$t_{CSTH\text{-}HGO}$	—	—	—	
6	D	Square wave input clock frequency (EREFS = 0, ERCLKEN = 1)	f_{extal}				MHz
		FEE or FBE mode [2]		0.03125	—	5	
		FBELP mode		0	—	40	

[1] Typical data was characterized at 5.0 V, 25 °C or is recommended value.

[2] The input clock source must be divided using RDIV to within the range of 31.25 kHz to 39.0625 kHz.

[3] This parameter is characterized and not tested on each device. Proper PC board layout procedures must be followed to achieve specifications.

[4] 4 MHz crystal.



A.9 AC Characteristics

This section describes AC timing characteristics for each peripheral system.

A.9.1 Control Timing

Table A-9. Control Timing

Num	C	Parameter	Symbol	Min	Typical	Max	Unit
1	D	Bus frequency ($t_{cyc} = 1/f_{Bus}$)	f_{Bus}	0	—	10	MHz
2	D	Real time interrupt internal oscillator period	t_{RTI}	—	1000	—	µs
3	D	External $\overline{\text{RESET}}$ pulse width[1]	t_{extrst}	150	—	—	ns
4	D	Port rise and fall time (load = 50 pF)[2] Slew rate control disabled (PTxSE = 0) Slew rate control enabled (PTxSE = 1)	t_{Rise}, t_{Fall}	 — —	 11 35	 — —	ns

[1] This is the shortest pulse guaranteed to pass through the pin input filter circuitry. Shorter pulses may or may not be recognized.

[2] Timing is shown with respect to 20% V_{DD} and 80% V_{DD} levels. Temperature range –40 °C to 85 °C.

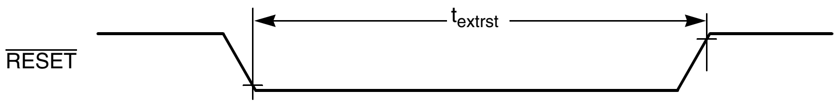


Figure A-6. Reset Timing

A.9.2 TPM/MTIM Module Timing

Synchronizer circuits determine the shortest input pulses that can be recognized or the fastest clock that can be used as the optional external source to the timer counter. These synchronizers operate from the current bus rate clock.

Table A-10. TPM Input Timing

Num	C	Rating	Symbol	Min	Max	Unit
1	D	External clock frequency	f_{TPMext}	DC	$f_{Bus}/4$	MHz
2	D	External clock period	t_{TPMext}	4	—	t_{cyc}
3	D	External clock high time	t_{clkh}	1.5	—	t_{cyc}
4	D	External clock low time	t_{clkl}	1.5	—	t_{cyc}
5	D	Input capture pulse width	t_{ICPW}	1.5	—	t_{cyc}

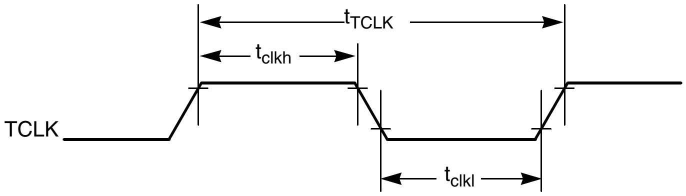


Figure A-7. Timer External Clock

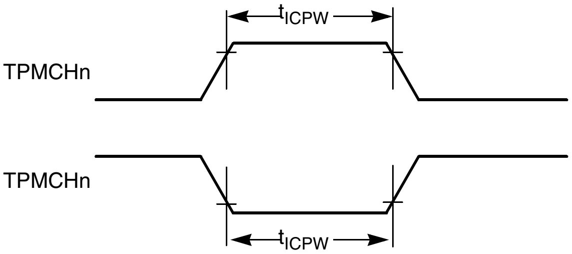


Figure A-8. Timer Input Capture Pulse

A.10 Analog Comparator (ACMP) Electrical

Table A-11. Analog Comparator Electrical Specifications

Num	C	Characteristic	Symbol	Min	Typical	Max	Unit
1	D	Supply voltage	V_{DD}	2.7	—	5.5	V
2	P	Supply current (active)	I_{DDAC}	—	20	35	µA
3	D	Analog input voltage[1]	V_{AIN}	V_{SS} – 0.3	—	V_{DD}	V
4	P	Analog input offset voltage[1]	V_{AIO}		20	40	mV
5	C	Analog Comparator hysteresis[1]	V_H	3.0	9.0	15.0	mV
6	C	Analog source impedance[1]	R_{AS}	—	—	10	kΩ
7	P	Analog input leakage current	I_{ALKG}	—	—	1.0	µA
8	C	Analog Comparator initialization delay	t_{AINIT}	—	—	1.0	µs
9	P	Analog Comparator bandgap reference voltage	V_{BG}	1.1	1.208	1.3	V

[1] These data are characterized but not production tested.

A.11 Internal Clock Source Characteristics

Table A-12. Internal Clock Source Specifications

Num	C	Characteristic	Symbol	Min	Typical[1]	Max	Unit
1	C	Average internal reference frequency — untrimmed	f_{int_ut}	25	31.25	41.66	kHz
2	P	Average internal reference frequency — trimmed	f_{int_t}	31.25	39.06	39.0625	kHz
3	C	DCO output frequency range — untrimmed	f_{dco_ut}	12.8	16	21.33	MHz
4	P	DCO output frequency range — trimmed	f_{dco_t}	16	20	20	MHz
5	C	Resolution of trimmed DCO output frequency at fixed voltage and temperature	$\Delta f_{dco\_res\_t}$	—	—	0.2	%fdco
6	C	Total deviation of trimmed DCO output frequency over voltage and temperature	Δf_{dco_t}	—	—	2	%fdco

Table A-12. Internal Clock Source Specifications

Num	C	Characteristic	Symbol	Min	Typical[1]	Max	Unit
7	C	FLL acquisition time[2,3]	$t_{acquire}$	—	—	1	ms
8	C	Stop recovery time (FLL wakeup to previous acquired frequency) IREFSTEN=0 IREFSTEN=1	t_wakeup	—	100 86	—	μs

[1] Data in typical column was characterized at 5.0 V, 25 °C or is typical recommended value.

[2] This parameter is characterized and not tested on each device.

[3] This specification applies to any time the FLL reference source or reference divider is changed, trim value changed or changing from FLL disabled (FBILP) to FLL enabled (FEI, FBI).

A.12 ADC Characteristics

Table A-13. 8-bit ADC Operating Conditions

C	Characteristic	Conditions	Symbol	Min.	Typical	Max.	Unit
D	Input voltage		V_{ADIN}	V_{SS}	—	V_{DD}	V
C	Accuracy	V_{DD} = 2 V	—	—	8 bit	—	—
C	Input capacitance		C_{ADIN}	—	4.5	—	pF
C	Input resistance		R_{ADIN}	—	3	—	kΩ
C	Analog source resistance external to MCU	8 bit mode (all valid f_{ADCK})	R_{AS}	—	—	10	kΩ
D	ADC conversion clock frequency	High Speed (ADLPC=0)	f_{ADCK}	0.4	—	8.0	MHz
		Low Power (ADLPC=1)		0.4	—	8.0	

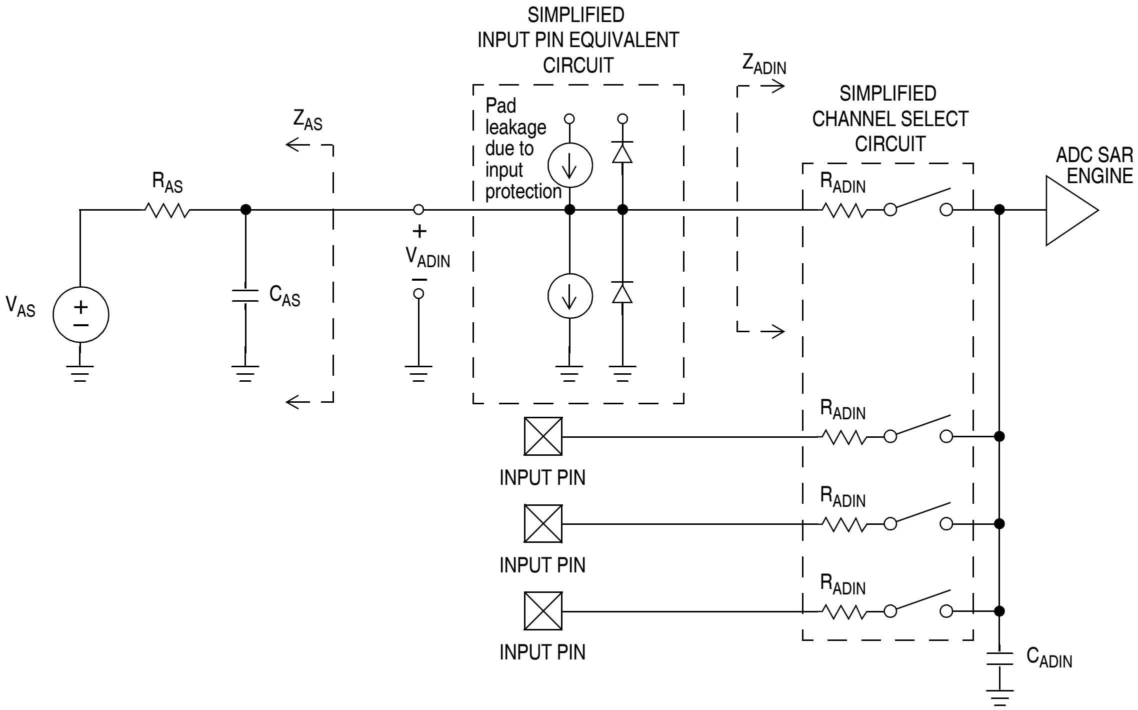


Figure A-9. ADC Input Impedance Equivalency Diagram

Table A-14. 8-bit ADC Characteristics

Characteristic	Conditions	C	Symbol	Min	Typical[1]	Max	Unit
Supply current ADLPC = 1 ADLSMP = 1 ADCO = 1		T	I_{DDAD}	—	133	—	µA
Supply current ADLPC = 1 ADLSMP = 0 ADCO = 1		T	I_{DDAD}	—	218	—	µA
Supply current ADLPC = 0 ADLSMP = 1 ADCO = 1		T	I_{DDAD}	—	327	—	µA
Supply current ADLPC = 0 ADLSMP = 0 ADCO = 1		C	I_{DDAD}	—	0.582	—	mA
Supply current	Stop, reset, module off	T	I_{DDAD}	—	0.011	—	µA
ADC asynchronous clock source	High speed (ADLPC = 0)	T	f_{ADACK}	—	3.3	—	MHz
	Low power (ADLPC = 1)			—	2	—	

Table A-14. 8-bit ADC Characteristics (continued)

Characteristic	Conditions	C	Symbol	Min	Typical[1]	Max	Unit
Conversion time (including sample time)	Short sample (ADLSMP=0)	P	t_{ADC}	—	20	—	ADCK cycles
	Long sample (ADLSMP=1)			—	40	—	
Sample time	Short sample (ADLSMP=0)	P	t_{ADS}	—	3.5	—	ADCK cycles
	Long sample (ADLSMP=1)			—	23.5	—	
Total unadjusted error	8 bit mode	C	E_{TUE}	—	±0.5	—	LSB[2]
Differential non-linearity	8 bit mode	P	DNL	—	±0.3	—	LSB[2]
	Monotonicity and No-Missing-Codes guaranteed						
Integral non-linearity	8 bit mode	C	INL	—	±0.5	—	LSB[2]
Zero-scale error	8 bit mode	P	E_{ZS}	—	±0.5	—	LSB[2]
Full-Scale error $V_{ADIN} = V_{DDA}$	8 bit mode	P	E_{FS}	—	±0.5	—	LSB[2]
Quantization error	8 bit mode	D	E_Q	—	±0.5	—	LSB[2]
Input leakage error pad leakage[3] R_{AS}	8 bit mode	D	E_{IL}	—	±0.2	—	LSB[2]

[1] Typical values assume Temp = 25 °C, f_{ADCK} = 1.0 MHz unless otherwise stated. Typical values are for reference only and are not tested in production.

[2] 1 LSB = $(V_{REFH} - V_{REFL})/2^N$

[3] Based on input pad leakage current. Refer to pad electrical.

A.13 Flash Specifications

This section provides details about program/erase times and program-erase endurance for the flash memory. For detailed information about program/erase operations, see Section 4.6, “Flash.”

Table A-15. Flash Characteristics

Characteristic	Symbol	Min	Typical[1]	Max	Unit
Supply voltage for program/erase	V_{DD}	2.7	—	5.5	V
Program/Erase voltage	V_{PP}	11.8	12	12.2	V
VPP current Program Mass erase	 I_{VPP_prog} I_{VPP_erase}	 — —	 — —	 200 100	 µA µA
Supply voltage for read operation $0 < f_{Bus} < 10$ MHz	V_{Read}	2.7	—	5.5	V
Byte program time	t_{prog}	20	—	40	µs
Mass erase time	t_{me}	500	—	—	ms

Table A-15. Flash Characteristics (continued)

Characteristic	Symbol	Min	Typical[1]	Max	Unit
Cumulative program HV time[2]	t_{hv}	—	—	8	ms
Total cumulative HV time (total of t_{me} & t_{hv} applied to device)	t_{hv_total}	—	—	2	hours
HVEN to program setup time	t_{pgs}	10	—	—	µs
PGM/MASS to HVEN setup time	t_{nvs}	5	—	—	µs
HVEN hold time for PGM	t_{nvh}	5	—	—	µs
HVEN hold time for MASS	t_{nvh1}	100	—	—	µs
V_{PP} to PGM/MASS setup time	t_{vps}	20	—	—	ns
HVEN to V_{PP} hold time	t_{vph}	20	—	—	ns
V_{PP} rise time[3]	t_{vrs}	200	—	—	ns
Recovery time	t_{rcv}	1	—	—	µs
Program/erase endurance T_L to T_H = –40°C to 85°C		1000	—		cycles
Data retention	t_{D_ret}	15	—	—	years

[1] Typicals are measured at 25 °C.

[2] t_{hv} is the cumulative high voltage programming time to the same row before next erase. Same address can not be programmed more than twice before next erase.

[3] Fast V_{PP} rise time may potentially trigger the ESD protection structure, which may result in over current flowing into the pad and cause permanent damage to the pad. External filtering for the V_{PP} power source is recommended. An example V_{PP} filter is shown in Figure A-10.

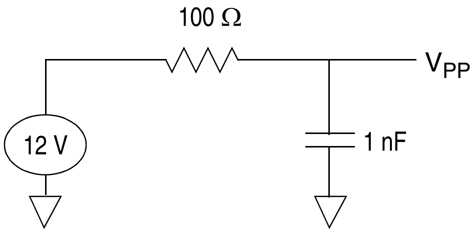
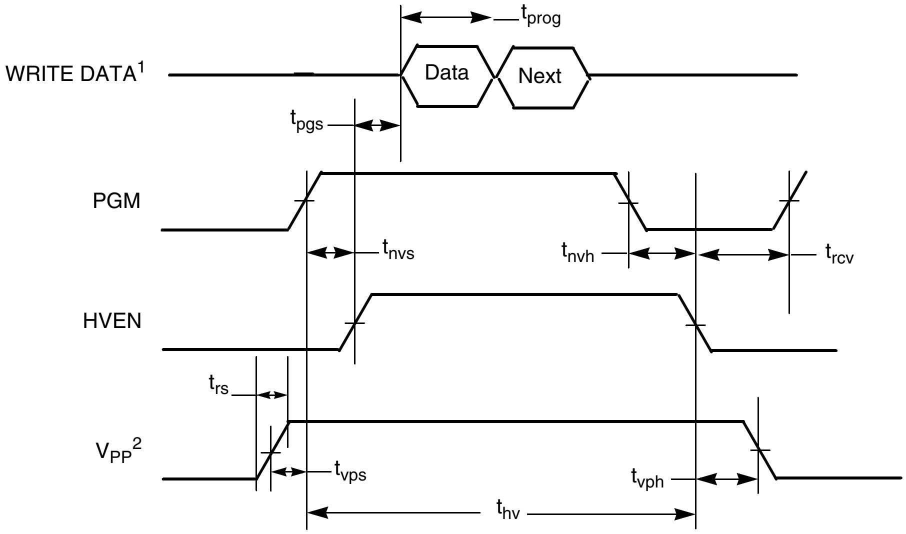


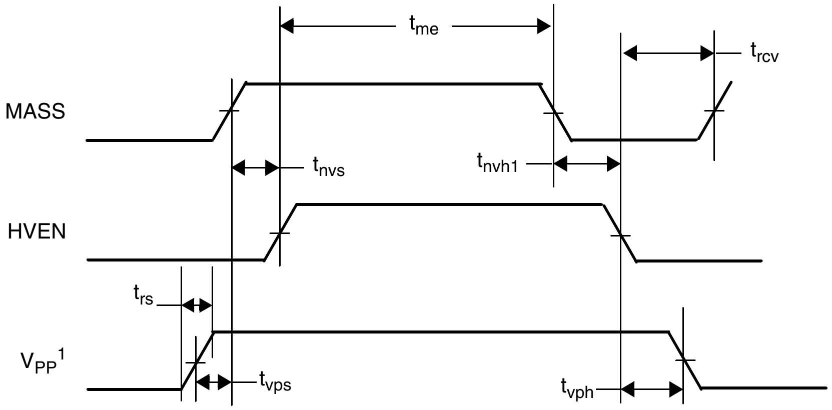
Figure A-10. Example V_{PP} Filtering



1 Next Data applies if programming multiple bytes in a single row, refer to Section 4.6, "Flash."

2 V_{DD} must be at a valid operating voltage before voltage is applied or removed from the V_{PP} pin.

Figure A-11. Flash Program Timing



1 V_{DD} must be at a valid operating voltage before voltage is applied or removed from the V_{PP} pin.

Figure A-12. Flash Mass Erase Timing

A.14 EMC Performance

Electromagnetic compatibility (EMC) performance is highly dependant on the environment in which the MCU resides. Board design and layout, circuit topology choices, location and characteristics of external components as well as MCU software operation all play a significant role in EMC performance. The system designer should consult Freescale applications notes such as AN2321, AN1050, AN1263, AN2764, and AN1259 for advice and guidance specifically targeted at optimizing EMC performance.

A.14.1 Radiated Emissions

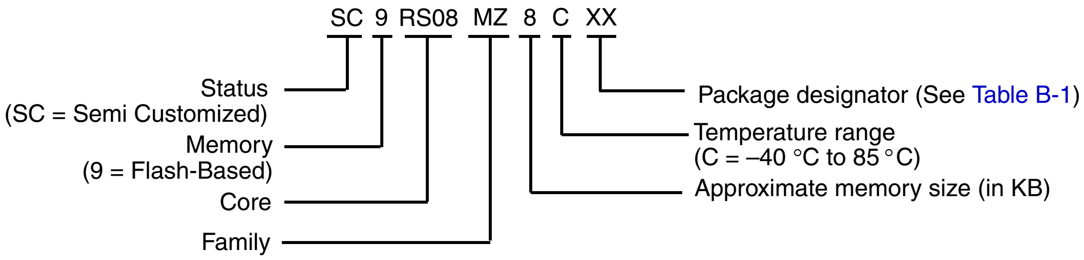
Microcontroller radiated RF emissions are measured from 150 kHz to 1 GHz using the TEM/GTEM Cell method in accordance with the IEC 61967-2 and SAE J1752/3 standards. The measurement is performed with the microcontroller installed on a custom EMC evaluation board while running specialized EMC test software. The radiated emissions from the microcontroller are measured in a TEM cell in two package orientations (North and East).

Appendix B
Ordering Information

This section contains ordering numbers for SC9RS08MZ8 device. See below for an example of the device numbering system.

Table B-1. Device Numbering System

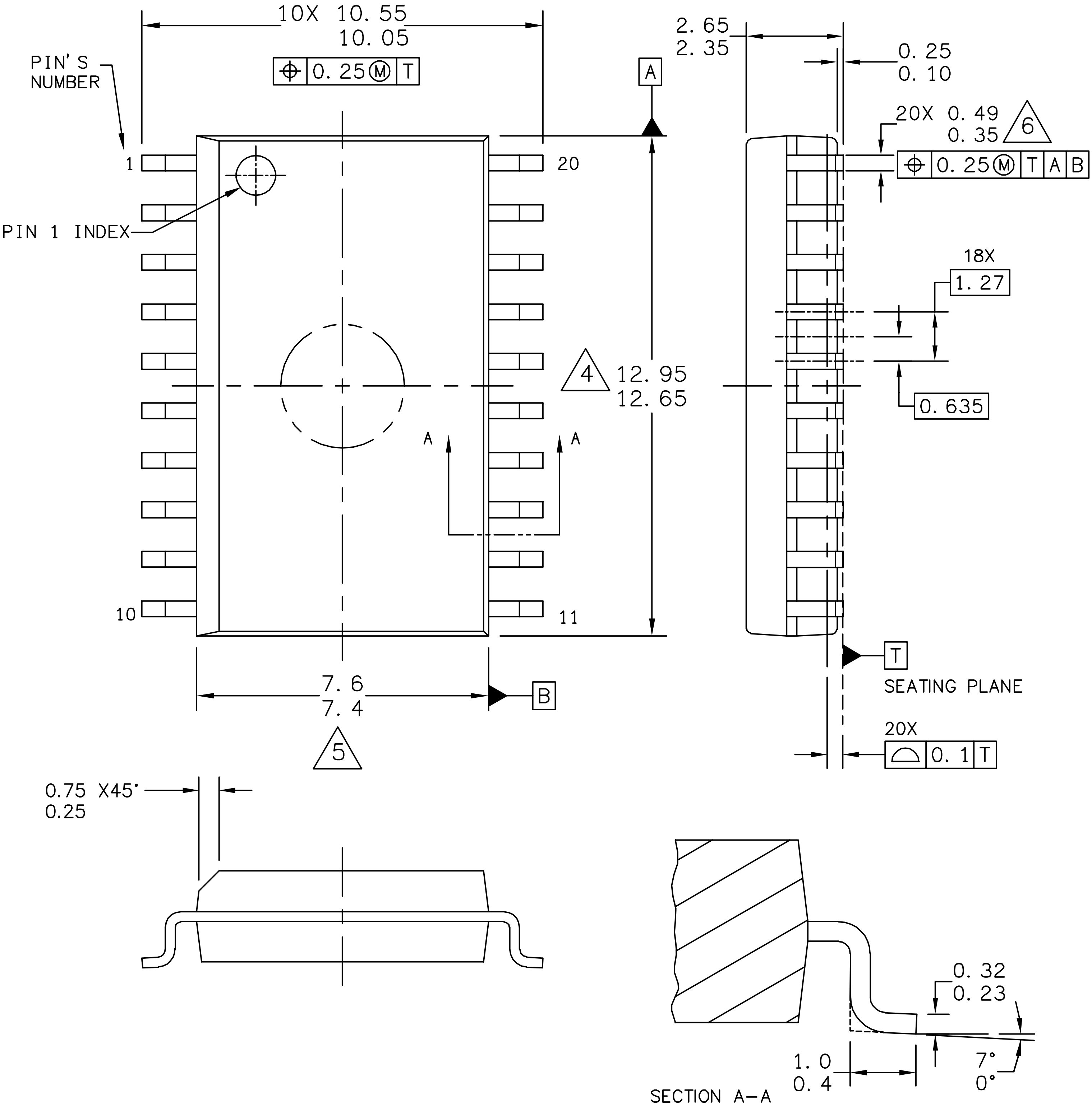
Device Number	Memory		Package		
	Flash	RAM	Type	Designator	Document No.
SC9RS08MZ8	8 Kbytes	254 bytes	20 W-SOIC	WJ	98ASB42343B



B.1 Mechanical Drawings

This following pages contain mechanical specifications for SC9RS08MZ8 package options.

- 20-pin W-SOIC (wide body small outline integrated circuit)



© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE

PRINT VERSION NOT TO SCALE

TITLE:
20LD SOIC W/B, 1.27 PITCH
CASE-OUTLINE

DOCUMENT NO: 98ASB42343B

REV: J

CASE NUMBER: 751D-07

23 MAR 2005

STANDARD: JEDEC MS-013AC

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–1994.

3. DATUMS A AND B TO BE DETERMINED AT THE PLANE WHERE THE BOTTOM OF THE LEADS EXIT THE PLASTIC BODY.

△4. THIS DIMENSION DOES NOT INCLUDE MOLD FLASH, PROTRUSION OR GATE BURRS. MOLD FLASH, PROTRUSION OR GATE BURRS SHALL NOT EXCEED 0.15 MM PER SIDE. THIS DIMENSION IS DETERMINED AT THE PLANE WHERE THE BOTTOM OF THE LEADS EXIT THE PLASTIC BODY.

△5. THIS DIMENSION DOES NOT INCLUDE INTER–LEAD FLASH OR PROTRUSIONS. INTER–LEAD FLASH AND PROTRUSIONS SHALL NOT EXCEED 0.25 MM PER SIDE. THIS DIMENSION IS DETERMINED AT THE PLANE WHERE THE BOTTOM OF THE LEADS EXIT THE PLASTIC BODY.

△6. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.62 mm.

© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE

PRINT VERSION NOT TO SCALE

TITLE:
20LD SOIC W/B, 1.27 PITCH,
CASE OUTLINE

DOCUMENT NO: 98ASB42343B

REV: J

CASE NUMBER: 751D–07

23 MAR 2005

STANDARD: JEDEC MS–013AC

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale’s Environmental Products program, go to http://www.freescale.com/epp.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2008. All rights reserved.

SC9RS08MZ8
Rev. 1, 9/2008