

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

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

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

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

Details

Product Status	Obsolete
Core Processor	RS08
Core Size	8-Bit
Speed	10MHz
Connectivity	-
Peripherals	LVD, POR, WDT
Number of I/O	4
Program Memory Size	2KB (2K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	63 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	8-SOIC (0.209", 5.30mm Width)
Supplier Device Package	8-SO
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/pc9rs08ka2dwe

Section Number	Title	Page
----------------	-------	------

Chapter 11

Modulo Timer (RS08MTIMV1)

11.1	Introduction	89
11.1.1	Features	90
11.1.2	Modes of Operation	90
11.1.2.1	Operation in Wait Mode	90
11.1.2.2	Operation in Stop Modes	90
11.1.2.3	Operation in Active Background Mode	90
11.1.3	Block Diagram	91
11.2	External Signal Description	91
11.3	Register Definition	91
11.3.1	MTIM Status and Control Register (MTIMSC)	92
11.3.2	MTIM Clock Configuration Register (MTIMCLK)	93
11.3.3	MTIM Counter Register (MTIMCNT)	93
11.3.4	MTIM Modulo Register (MTIMMOD)	94
11.4	Functional Description	95
11.4.1	MTIM Operation Example	96

Chapter 12

Development Support

12.1	Introduction	97
12.2	Features	97
12.3	RS08 Background Debug Controller (BDC)	98
12.3.1	BKGD Pin Description	99
12.3.2	Communication Details	99
12.3.3	SYNC and Serial Communication Timeout	102
12.4	BDC Registers and Control Bits	103
12.4.1	BDC Status and Control Register (BDCSCR)	103
12.4.2	BDC Breakpoint Match Register	104
12.5	RS08 BDC Commands	105

Section Number	Title	Page
Appendix A		
Electrical Characteristics		
A.1	Introduction	109
A.2	Absolute Maximum Ratings	109
A.3	Thermal Characteristics	110
A.4	Electrostatic Discharge (ESD) Protection Characteristics	111
A.5	DC Characteristics	111
A.6	Supply Current Characteristics	115
A.7	Analog Comparator (ACMP) Electricals	117
A.8	Internal Clock Source Characteristics	117
A.9	AC Characteristics	118
	A.9.1 Control Timing	118
A.10	FLASH Specifications	119
Appendix B		
Ordering Information and Mechanical Drawings		
B.1	Ordering Information	123
B.2	Mechanical Drawings	123

- When a BDC breakpoint is encountered

After active background mode is entered, the CPU is held 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, defined as 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, which 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 MC9RS08KA2 Series is shipped from the Freescale Semiconductor factory, the Flash program memory is usually erased so there is no program that could 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 has been previously programmed.

For additional information about the active background mode, refer to the [Development Support](#) chapter of this data sheet.

3.5 Wait Mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The program counter (PC) is halted at the position where the WAIT instruction is executed. When an interrupt request occurs:

1. MCU exits wait mode and resumes processing.
2. PC is incremented by one and fetches the next instruction to be processed.

It is the responsibility of the user program to probe the corresponding interrupt source that woke the MCU, because no vector fetching process is involved.

While the MCU is in wait mode, not all background debug commands can be used. Only the BACKGROUND command and memory-access-with-status commands are available when the MCU is in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from wait mode and enter active background mode.

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

Table 3-1. Wait Mode Behavior

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

3.6 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 the modules are halted. If the STOPE bit is not set when the CPU executes a STOP instruction, the MCU will not enter stop mode and an illegal opcode reset is forced.

Table 3-2 summarizes the behavior of the MCU in stop mode.

Table 3-2. Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS ¹	ACMP ²	Regulator	I/O Pins	RTI ³
Stop	Standby	Standby	Optionally on	Optionally on	Standby	States held	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 If the 32-kHz trimmed clock in the ICS module is selected as the clock source for the RTI, LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

Upon entering stop mode, all of the clocks in the MCU are halted. The ICS is turned off by default when the IREFSTEN bit is cleared and the voltage regulator is put in 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 $\overline{\text{RESET}}$, any asynchronous interrupt that has been enabled, or the real-time interrupt. The asynchronous interrupts are the KBI pins, LVD interrupt, or the ACMP interrupt.

If stop is exited by asserting the $\overline{\text{RESET}}$ pin, the MCU will be reset and program execution starts at location \$3FFD. If exited by means of an asynchronous interrupt or real-time interrupt, the next instruction after the location where the STOP instruction was executed will be executed accordingly. It is the responsibility of the user program to 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 and the 1-kHz source are disabled. Power consumption is lower when the 1-kHz source is disabled, but in that case, the real-time interrupt cannot wake the MCU from stop.

The trimmed 32-kHz clock in the ICS module can also be enabled for the real-time interrupt to allow a wakeup from stop mode with no external components. The 32-kHz clock reference is enabled by setting

4.2 Unimplemented Memory

Attempting to access either data or an instruction at an unimplemented memory address will cause reset.

4.3 Indexed/Indirect Addressing

Register D[X] and register X together perform the 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 that is 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 *lda ,x* 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.

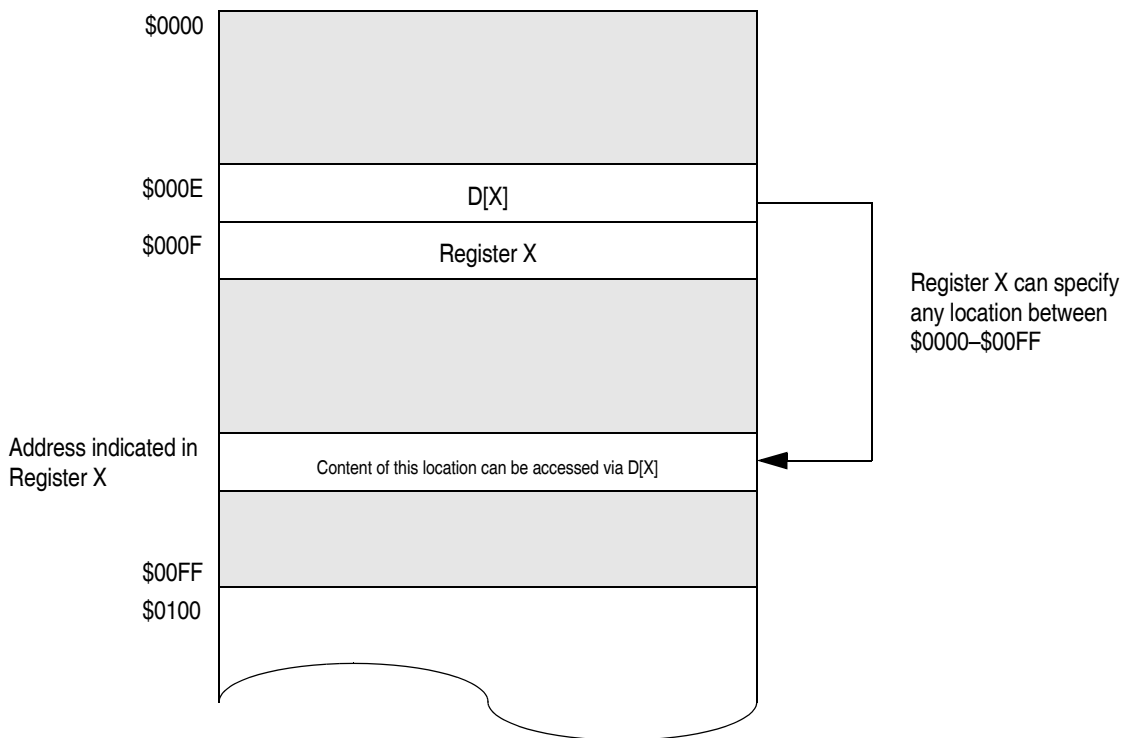


Figure 4-2. Indirect Addressing Registers

4.4 RAM and Register Addresses and Bit Assignments

The fast access RAM area can be accessed by instructions using tiny, short, and direct addressing mode instructions. For tiny addressing mode instructions, the operand is encoded along with the opcode to a single byte.

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 the 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_{nvhl} .
9. Clear the HVEN bit.
10. After time, t_{rcv} , the memory can be accessed in read mode again.
11. Remove external V_{pp} .

NOTE

Flash memory cannot be programmed or erased by software code executed from Flash locations. To program or erase Flash, commands must be executed from RAM or BDC commands. User code should not enter wait or stop during an erase or program sequence.

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

4.6.4 Security

The MC9RS08KA2 Series includes circuitry to help prevent unauthorized access to the contents of Flash memory. When security is engaged, Flash is considered a secure resource. The RAM, direct-page registers, and background debug controller are considered unsecured resources. Attempts to access a secure memory location through the background debug interface, or whenever BKGDP is set, are blocked (reads return all 0s).

Security is engaged or disengaged based on the state of a nonvolatile register bit (SECD) in the FOPT register. During reset, the contents of the nonvolatile location NVOPT are copied from Flash into the working FOPT register in high-page register space. A user engages security by programming the NVOPT location, which can be done at the same time the Flash memory is programmed. Notice the erased state (SECD = 1) makes the MCU unsecure. When SECD in NVOPT is programmed (SECD = 0), next time the device is reset via POR, internal reset, or external reset, security is engaged. In order 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. 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,”](#) via BDM commands.

Security can always be disengaged through the background debug interface by following these steps:

1. Mass erase Flash via background BDM commands or RAM loaded program.
2. Perform reset and the device will boot up with security disengaged.

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 MC9RS08KA2 Series is hard coded to the value \$0800. See also ID bits in Figure 5-4 .

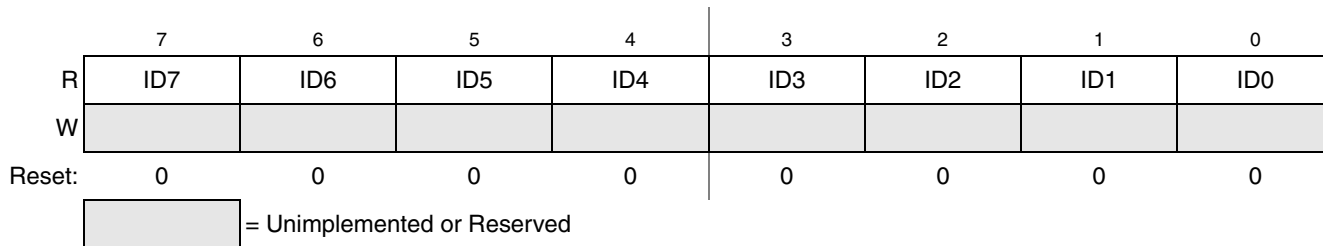


Figure 5-4. System Device Identification Register — Low (SDIDL)

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 MC9RS08KA2 Series is hard coded to the value \$0800. 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.

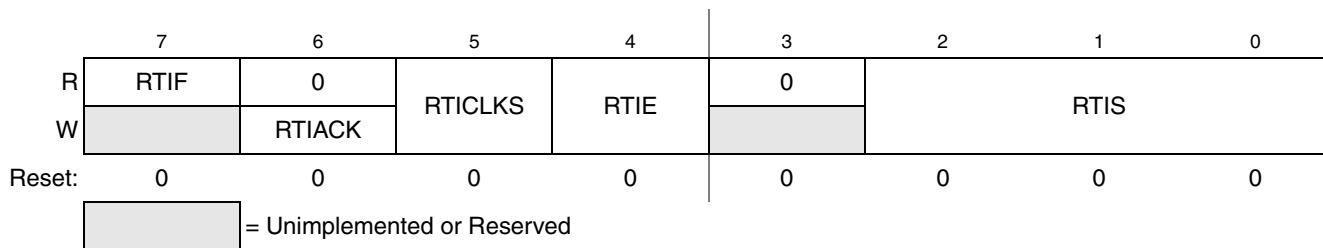


Figure 5-5. System RTI Status and Control Register (SRTISC)

Table 5-6. SRTISC Register Field Descriptions

Field	Description
7 RTIF	Real-Time Interrupt Flag — This 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 — This write-only bit is used to acknowledge 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 — This 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 internal trimmed 32-kHz oscillator (ICS module) and is divided by 32 in RTI logic to produce a trimmed 1-kHz clock source for RTI counter.

Chapter 7

Keyboard Interrupt (RS08KBIV1)

7.1 Introduction

The keyboard interrupt (KBI) module provides independently enabled external interrupt sources.

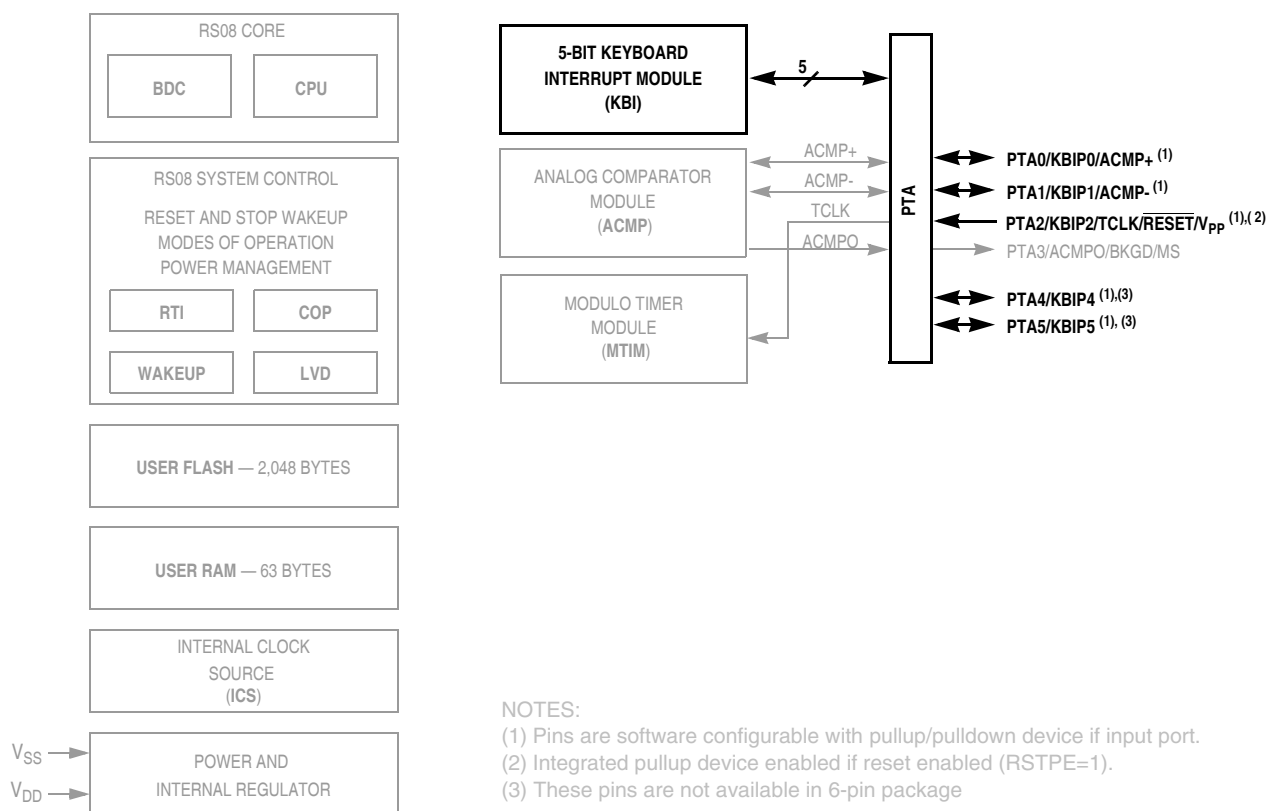


Figure 7-1. MC9RS08KA2 Series Block Diagram with KBI Block and Pins Highlighted

7.1.1 Features

The KBI features include:

- Each keyboard interrupt pin has an individual pin enable bit
- Each keyboard interrupt pin is programmable as falling edge (or rising edge) only, or both falling edge and low level (or both rising edge and high level) interrupt sensitivity
- One software-enabled keyboard interrupt
- Exit from low-power modes

8.2.2 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.

8.2.3 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.

8.2.4 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 8-3](#) identifies the CCR bits and their bit positions.

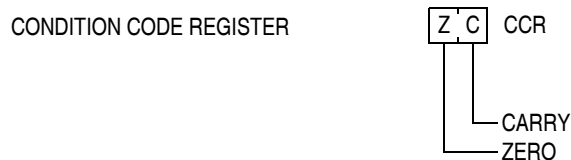


Figure 8-3. Condition Code Register (CCR)

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:

```

8.3.5 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.

8.3.6 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.

8.3.7 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.

8.3.8 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.

8.4 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:

8.5 Summary Instruction Table

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in [Table 8-1](#) through [Table 8-2](#).

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
<i>rel</i>	=	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
↑	=	Bit set or cleared according to results of operation
U	=	Undefined after the operation

Machine coding notation

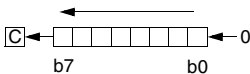
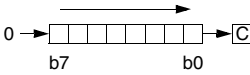
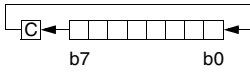
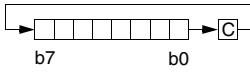
Table 8-1. Instruction Set Summary (Sheet 3 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles	
			Z	C					
BRSET <i>n,opr8a,rel</i>	Branch if Bit <i>n</i> in Memory Set	PC ← (PC) + \$0003 + <i>rel</i> , if (Mn) = 1	—	↑	DIR (b0)	00	dd rr	5	
BRSET <i>n,D[X],rel</i>					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	
					IX (b0)	00	0E rr	5	
BRSET <i>n,X,rel</i>					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	
					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 <i>n,opr8a</i>					Set Bit <i>n</i> in Memory	Mn ← 1	—	—	DIR (b0)
	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					
	IX (b0)	10	0E	5					
	BSET <i>n,D[X]</i>	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 <i>n,X</i>	DIR (b0)	10					0F
	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 8-1. 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	$\$0F \leftarrow (M)$	\updownarrow	—	IMD DIR IX	3E 4E 4E	ii 0F dd 0F 0E 0E	4 5 5
LSLA	Logical Shift Left		\updownarrow	\updownarrow	INH	48		1
LSRA	Logical Shift Right		\updownarrow	\updownarrow	INH	44		1
MOV opr8a,opr8a MOV #opr8i,opr8a MOV D[X],opr8a MOV opr8a,D[X] MOV #opr8i,D[X]	Move	$(M)_{\text{destination}} \leftarrow (M)_{\text{source}}$	\updownarrow	—	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	$A \leftarrow (A) \mid (M)$ $A \leftarrow (A) \mid (X)$	\updownarrow	—	IMM DIR IX DIR	AA BA BA BA	ii dd 0E 0F	2 3 3 3
ROLA	Rotate Left through Carry		\updownarrow	\updownarrow	INH	49		1
RORA	Rotate Right through Carry		\updownarrow	\updownarrow	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	$A \leftarrow (A) - (M) - (C)$ $A \leftarrow (A) - (X) - (C)$	\updownarrow	\updownarrow	IMM DIR IX DIR	A2 B2 B2 B2	ii dd 0E 0F	2 3 3 3
SEC	Set Carry Bit	$C \leftarrow 1$	—	1	INH	39		1
SHA	Swap Shadow PC High with A	$A \leftrightarrow \text{SPCH}$	—	—	INH	45		1
SLA	Swap Shadow PC Low with A	$A \leftrightarrow \text{SPCL}$	—	—	INH	42		1
STA opr8a STA opr5a STA ,X ⁽¹⁾ STA X	Store Accumulator in Memory	$M \leftarrow (A)$	\updownarrow	—	DIR SRT IX SRT	B7 Ex / Fx EE EF	dd	3 2 2 2
STX opr8a ⁽¹⁾	Store Index Register in Memory	$M \leftarrow (X)$	\updownarrow	—	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 8-2. 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	BRSET0 ⁵ ₃ DIR	BSET0 ⁵ ₂ DIR	INC ⁴ ₁ TNY	BRA ³ ₂ REL		DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	SUB ² ₂ IMM	SUB ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
1	BRCLR0 ⁵ ₃ DIR	BCLR0 ⁵ ₂ DIR	INC ⁴ ₁ TNY	CBEQ ⁵ ₃ DIR	CBEQA ⁴ ₃ IMM	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	CMP ² ₂ IMM	CMP ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
2	BRSET1 ⁵ ₃ DIR	BSET1 ⁵ ₂ DIR	INC ⁴ ₁ TNY		SLA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	SBC ² ₂ IMM	SBC ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
3	BRCLR1 ⁵ ₃ DIR	BCLR1 ⁵ ₂ DIR	INC ⁴ ₁ TNY		COMA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT			LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
4	BRSET2 ⁵ ₃ DIR	BSET2 ⁵ ₂ DIR	INC ⁴ ₁ TNY	BCC ³ ₂ REL	LSRA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	AND ² ₂ IMM	AND ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
5	BRCLR2 ⁵ ₃ DIR	BCLR2 ⁵ ₂ DIR	INC ⁴ ₁ TNY	BCS ³ ₂ REL	SHA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT			LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
6	BRSET3 ⁵ ₃ DIR	BSET3 ⁵ ₂ DIR	INC ⁴ ₁ TNY	BNE ³ ₂ REL	RORA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	LDA ² ₂ IMM	LDA ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
7	BRCLR3 ⁵ ₃ DIR	BCLR3 ⁵ ₂ DIR	INC ⁴ ₁ TNY	BEQ ³ ₂ REL		DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT		STA ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
8	BRSET4 ⁵ ₃ DIR	BSET4 ⁵ ₂ DIR	INC ⁴ ₁ TNY	CLC ¹ ₁ INH	LSLA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	EOR ² ₂ IMM	EOR ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
9	BRCLR4 ⁵ ₃ DIR	BCLR4 ⁵ ₂ DIR	INC ⁴ ₁ TNY	SEC ¹ ₁ INH	ROLA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	ADC ² ₂ IMM	ADC ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
A	BRSET5 ⁵ ₃ DIR	BSET5 ⁵ ₂ DIR	INC ⁴ ₁ TNY	DEC ⁵ ₂ DIR	DECA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	ORA ² ₂ IMM	ORA ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
B	BRCLR5 ⁵ ₃ DIR	BCLR5 ⁵ ₂ DIR	INC ⁴ ₁ TNY	DBNZ ⁶ ₃ DIR	DBNZA ⁴ ₂ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	ADD ² ₂ IMM	ADD ³ ₂ DIR	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
C	BRSET6 ⁵ ₃ DIR	BSET6 ⁵ ₂ DIR	INC ⁴ ₁ TNY	INC ⁵ ₂ DIR	INCA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	NOP ¹ ₁ INH	JMP ⁴ ₃ EXT	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
D	BRCLR6 ⁵ ₃ DIR	BCLR6 ⁵ ₂ DIR	INC ⁴ ₁ TNY			DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	BSR ³ ₂ REL	JSR ⁴ ₃ EXT	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
E	BRSET7 ⁵ ₃ DIR	BSET7 ⁵ ₂ DIR	INC ⁴ ₁ TNY	MOV ⁴ ₃ IMD	MOV ⁵ ₃ DD	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	STOP ²⁺ ₁ INH	RTS ³ ₁ INH	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT
F	BRCLR7 ⁵ ₃ DIR	BCLR7 ⁵ ₂ DIR	INC ⁴ ₁ TNY	CLR ³ ₂ DIR	CLRA ¹ ₁ INH	DEC ⁴ ₁ TNY	ADD ³ ₁ TNY	SUB ³ ₁ TNY	CLR ² ₁ SRT	CLR ² ₁ SRT	WAIT ²⁺ ₁ INH	BGND ⁵⁺ ₁ INH	LDA ³ ₁ SRT	LDA ³ ₁ SRT	STA ² ₁ SRT	STA ² ₁ SRT

INH

IMM

DIR

EXT

DD

Inherent

Immediate

Direct

Extended

Direct-Direct

REL

SRT

TNY

Relative

Short

Tiny

IMD

Immediate-Direct

Gray box is decoded as illegal instruction

High Byte of Opcode in Hexadecimal

B

Low Byte of Opcode in Hexadecimal

0

SUB³₂ DIR

RS08 Cycles
Opcode Mnemonic
Number of Bytes /
Addressing Mode

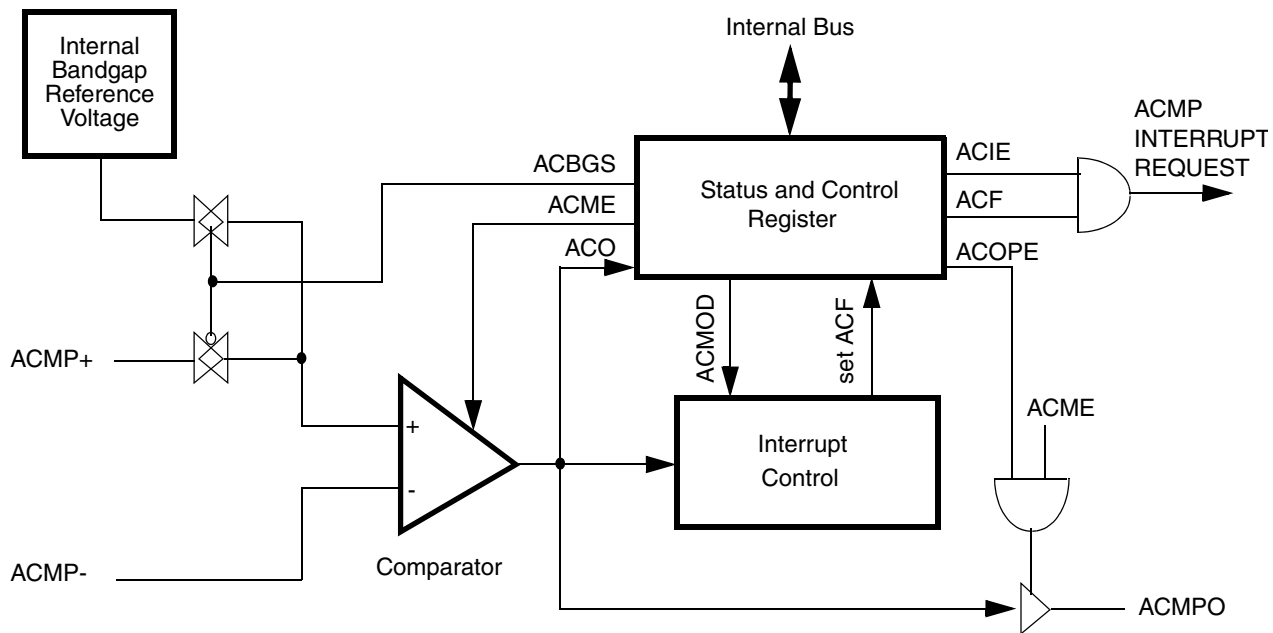


Figure 10-2. Analog Comparator (ACMP) Block Diagram

Table 10-2. 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.

10.4 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.

11.1.3 Block Diagram

The block diagram for the modulo timer module is shown [Figure 11-2](#).

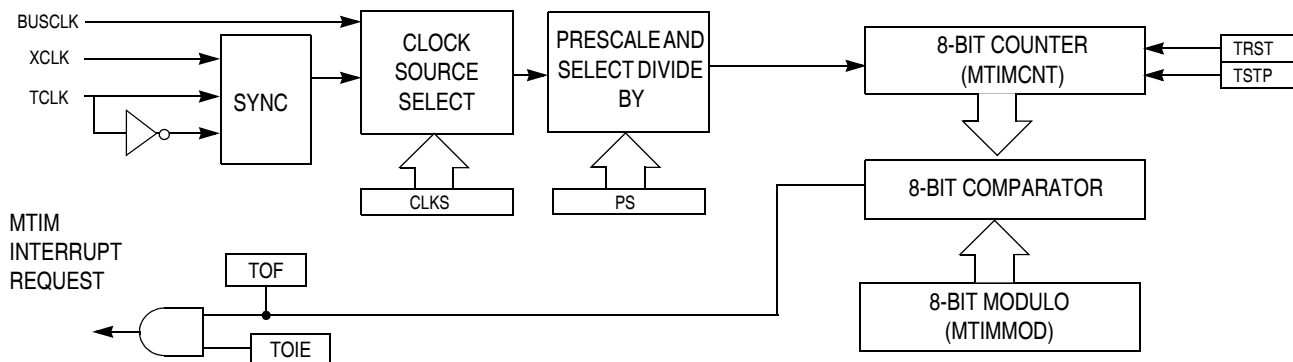


Figure 11-2. Modulo Timer (MTIM) Block Diagram

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.

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.

$$P_D = K \div (T_J + 273^\circ\text{C})$$

Eqn. A-2

Solving Equation A-1 and Equation A-2 for K gives:

$$K = P_D \times (T_A + 273^\circ\text{C}) + \theta_{JA} \times (P_D)^2$$

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.4 Electrostatic Discharge (ESD) Protection Characteristics

Although damage from static discharge is much less common on these devices than on early CMOS circuits, normal handling precautions should 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 CDF-AEC-Q00 Stress Test Qualification for Automotive Grade Integrated Circuits. (<http://www.aecouncil.com/>) A device is considered to have failed if, after exposure to ESD pulses, the device no longer meets the device specification requirements. 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-3. ESD Protection Characteristics

Parameter	Symbol	Value	Unit
ESD Target for Machine Model (MM) MM circuit description	V_{THMM}	200	V
ESD Target for Human Body Model (HBM) HBM circuit description	V_{THHBM}	2000	V

A.5 DC Characteristics

This section includes information about power supply requirements, I/O pin characteristics, and power supply current in various operating modes.

Table A-4. DC Characteristics
(Temperature Range = -40 to 85°C Ambient)

Parameter	Symbol	Min	Typical	Max	Unit
Supply voltage (run, wait and stop modes.) $0 < f_{BUS} < 10\text{MHz}$	V_{DD}	1.8	—	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}	1.80 1.88	1.86 1.94	1.95 2.03	V
Power on RESET (POR) voltage	V_{POR}	0.9	1.4	1.7	V
Input high voltage ($V_{DD} > 2.3\text{V}$) (all digital inputs)	V_{IH}	$0.70 \times V_{DD}$	—	—	V
Input high voltage ($1.8\text{V} \leq V_{DD} \leq 2.3\text{V}$) (all digital inputs)	V_{IH}	$0.85 \times V_{DD}$	—	—	V



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED
DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS
ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

MECHANICAL OUTLINES
DICTIONARY


DOCUMENT NO: 98ARL10602D

PAGE: 1677

DO NOT SCALE THIS DRAWING

REV: C

NOTES:

- 1. ALL DIMENSIONS ARE IN MILLIMETERS.
- 2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
- 3. THE COMPLETE JEDEC DESIGNATOR FOR THIS PACKAGE IS: HV–PDSO–N.
- 4.  COPLANARITY APPLIES TO LEADS AND DIE ATTACH PAD.
- 5. MIN. METAL GAP SHOULD BE 0.2 MM.

TITLE: THERMALLY ENHANCED DUAL
FLAT NON–LEADED PACKAGE (DFN)
6 TERMINAL, 0.95 PITCH (3 X 3 X 0.8)

CASE NUMBER: 1677–02

STANDARD: FREESCALE STD

PACKAGE CODE: 6197

SHEET: 5 OF 6