

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

**Embedded - Microcontrollers - Application Specific: Tailored Solutions for Precision and Performance**

**Embedded - Microcontrollers - Application Specific** represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering enhanced performance, efficiency, and functionality to meet the demands of specialized applications.

**What Are Embedded - Microcontrollers - Application Specific?**

Application specific microcontrollers are engineered to

#### Details

Product Status	Obsolete
Applications	Automotive
Core Processor	S12
Program Memory Type	FLASH (64kB)
Controller Series	HCS12
RAM Size	6K x 8
Interface	LIN, SCI
Number of I/O	9
Voltage - Supply	2.25V ~ 5.25V
Operating Temperature	-40°C ~ 125°C
Mounting Type	Surface Mount
Package / Case	48-LQFP Exposed Pad
Supplier Device Package	48-HLQFP (7x7)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/nxp-semiconductors/mm912h634cm1aer2">https://www.e-xfl.com/product-detail/nxp-semiconductors/mm912h634cm1aer2</a>

Table 60. 0x0020–0x002F Debug Module (DBG)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
0x0022	DBGTCR	R	0	TSOURCE	0	0	TRCMOD	0	TALIGN	
		W								
0x0023	DBGC2	R	0	0	0	0	0	ABCM		
		W								
0x0024	DBGTBH	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0025	DBGTBL	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0026	DBGCNT	R	TBF	0	CNT					
		W								
0x0027	DBGSCRX	R	0	0	0	0	SC3	SC2	SC1	SC0
		W								
0x0027	DBGMFR	R	0	0	0	0	MC2	MC1	MC0	
		W								
0x0028	DBGACTL	R	SZE	SZ	TAG	BRK	RW	RWE	NDB	COMPE
		W								
0x0028	DBGBCTL	R	SZE	SZ	TAG	BRK	RW	RWE	0	COMPE
		W								
0x0028	DBGCCTL	R	0	0	TAG	BRK	RW	RWE	0	COMPE
		W								
0x0029	DBGXAH	R	0	0	0	0	0	Bit 17	Bit 16	
		W								
0x002A	DBGXAM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002B	DBGXAL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002C	DBGADH	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002D	DBGADL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002E	DBGADHM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002F	DBGADLM	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

Table 61. 0x0030–0x0033 Reserved

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0030-0x0033	Reserved	R	0	0	0	0	0	0	0
		W							

**Table 69. 0x0120 Port Integration Module (PIM) Map 3 of 3**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0120	PTIA	R	PTIA7	PTIA6	PTIA5	PTIA4	PTIA3	PTIA2	PTIA1	PTIA0
		W								
0x0121	PTIB	R	0	0	0	0	0	0	PTIB1	PTIB0
		W								
0x0122-0x017F	Reserved	R	0	0	0	0	0	0	0	0
		W								

**Table 70. 0x0180–0x1EF Reserved**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0180-0x01EF	Reserved	R	0	0	0	0	0	0	0	0
		W								

**Table 71. 0x01F0–0x01FF Clock and Power Management (CPMU) Map 2 of 2**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
0x01F0	Reserved	R	0	0	0	0	0	0	0	0	
		W									
0x01F1	CPMU LVCTL	R	0	0	0	0	0	LVDS	LVIE	LVIF	
		W									
0x01F6	Reserved	R	0	0	0	0	0	0	0	0	
		W									
0x01F7	Reserved	R	0	0	0	0	0	0	0	0	
		W									
0x01F8	CPMU IRCTRIMH	R	TCTRIM[3:0]				0	0	IRCTRIM[9:8]		
		W									
0x01F9	CPMU IRCTRIML	R	IRCTRIM[7:0]								
		W									
0x01FA	CPMUOSC	R	OSCE	OSCBW	OSCPINS_EN	OSCFILT[4:0]					
		W									
0x01FB	CPMUPROT	R	0	0	0	0	0	0	0	PROT	
		W									
0x01FC	Reserved	R	0	0	0	0	0	0	0	0	
		W									

**Table 72. 0x01FD–0x1FF Reserved**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x01FD-0x01FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

**Table 74. Analog die Registers<sup>(63)</sup> - 0x0200–0x02FF D2D Blocking Access (D2DI) 2 of 3/  
0x0300–0x03FF D2D Non Blocking Access (D2DI) 3 of 3**

Offset	Name		7	6	5	4	3	2	1	0
0x21	PTBC2	R	0	0	0	0	PWMCS	PWMEN	SERMOD	
	Port B Config Register 2	W								
0x22	PTB	R	0	0	0	0	0	PTB2	PTB1	PTB0
	Port B Data Register	W								
0x28	HSCR	R	HSOTIE	HSHVSD E	PWMCS2	PWMCS1	PWMHS2	PWMHS1	HS2	HS1
	High Side Control Register	W								
0x29	HSSR	R	HSOTC	0	0	0	HS2CL	HS1CL	HS2OL	HS1OL
	High Side Status Register	W								
0x30	LSCR	R	LSOTIE	0	PWMCS2	PWMCS1	PWMLS2	PWMLS1	LS2	LS1
	Low Side Control Register	W								
0x31	LSSR	R	LSOTC	0	0	0	LS2CL	LS1CL	LS2OL	LS1OL
	Low Side Status Register	W								
0x32	LSCEN	R	0	0	0	0	LSCEN			
	Low-Side Control Enable Register	W								
0x38	HSR	R	HOTIE	HOTC	0	0	0	0	0	HSUPON
	Hall Supply Register	W								
0x3C	CSR	R	CSE	0	0	0	CCD	CSGS		
	Current Sense Register	W								
0x40	SCIBD (hi)	R	LBKDIE	RXEDGIE	0	SBR12	SBR11	SBR10	SBR9	SBR8
	SCI Baud Rate Register	W								
0x41	SCIBD (lo)	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
	SCI Baud Rate Register	W								
0x42	SCIC1	R	LOOPS	0	RSRC	M	0	ILT	PE	PT
	SCI Control Register 1	W								
0x43	SCIC2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	SCI Control Register 2	W								
0x44	SCIS1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
	SCI Status Register 1	W								
0x45	SCIS2	R	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
	SCI Status Register 2	W								
0x46	SCIC3	R	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
	SCI Control Register 3	W								
0x47	SCID	R	R7	R6	R5	R4	R3	R2	R1	R0
	SCI Data Register	W	T7	T6	T5	T4	T3	T2	T1	T0
0x60	PWMCTL	R	CAE1	CAE0	PCLK1	PCLK0	PPOL1	PPOL0	PWME1	PWME0
	PWM Control Register	W								
0x61	PWMPRCLK	R	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
	PWM Presc. Clk Select Reg	W								
0x62	PWMSCLA	R	Bit 7	6	5	4	3	2	1	Bit 0
	PWM Scale A Register	W								

If the associated error was detected in the received character that caused RDRF to be set, the error flags — noise flag (NF), framing error (FE), and parity error flag (PF) — get set at the same time as RDRF. These flags are not set in overrun cases.

If RDRF was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (OR) flag gets set instead the data along with any associated NF, FE, or PF condition is lost.

At any time, an active edge on the RxD serial data input pin causes the RXEDGIF flag to set. The RXEDGIF flag is cleared by writing a “1” to it. This function does depend on the receiver being enabled (RE = 1).

### 5.16.3.5 Additional SCI Functions

The following sections describe additional SCI functions.

#### 5.16.3.5.1 8- and 9-Bit Data Modes

The SCI system (transmitter and receiver) can be configured to operate in 9-bit data mode by setting the M control bit in SCIC1. In 9-bit mode, there is a ninth data bit to the left of the MSB of the SCI data register. For the transmit data buffer, this bit is stored in T8 in SCIC3. For the receiver, the ninth bit is held in R8 in SCIC3.

For coherent writes to the transmit data buffer, write to the T8 bit before writing to SCID.

If the bit value to be transmitted as the ninth bit of a new character is the same as for the previous character, it is not necessary to write to T8 again. When data is transferred from the transmit data buffer to the transmit shifter, the value in T8 is copied at the same time data is transferred from SCID to the shifter.

9-bit data mode typically is used in conjunction with parity to allow eight bits of data plus the parity in the ninth bit. Or it is used with address-mark wake-up so the ninth data bit can serve as the wake-up bit. In custom protocols, the ninth bit can also serve as a software-controlled marker.

#### 5.16.3.5.2 Stop Mode Operation

During all stop modes, clocks to the SCI module are halted.

In stop1 and stop2 modes, all SCI register data is lost and must be re-initialized upon recovery from these two stop modes. No SCI module registers are affected in stop3 mode.

The receive input active edge detect circuit is still active in stop3 mode, but not in stop2. An active edge on the receive input brings the CPU out of stop3 mode if the interrupt is not masked (RXEDGIE = 1).

Note that because the clocks are halted, the SCI module will resume operation upon exit from stop (only in stop3 mode). Software should ensure stop mode is not entered while there is a character being transmitted out of or received into the SCI module.

#### 5.16.3.5.3 Loop Mode

When LOOPS = 1, the RSRC bit in the same register chooses between loop mode (RSRC = 0) or single-wire mode (RSRC = 1). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the RxD pin is not used by the SCI, so it reverts to a general purpose port I/O pin.

#### 5.16.3.5.4 Single-wire Operation

When LOOPS = 1, the RSRC bit in the same register chooses between loop mode (RSRC = 0) or single-wire mode (RSRC = 1). Single-wire mode is used to implement a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TxD pin. The RxD pin is not used and reverts to a general purpose port I/O pin.

In single-wire mode, the TXDIR bit in SCIC3 controls the direction of serial data on the TxD pin. When TXDIR = 0, the TxD pin is an input to the SCI receiver and the transmitter is temporarily disconnected from the TxD pin so an external device can send serial data to the receiver. When TXDIR = 1, the TxD pin is an output driven by the transmitter. In single-wire mode, the internal loop back connection from the transmitter to the receiver causes the receiver to receive characters that are sent out by the transmitter.

## 5.18.4 Register definition

### 5.18.4.1 Port B Configuration Register 1 (PTBC1)

**Table 155. Port B Configuration Register 1 (PTBC1)**

Offset <sup>(111)</sup>	0x20							Access: User read/write
	7	6	5	4	3	2	1	0
R	0	PUEB2	PUEB1	PUEB0	0	DDRB2	DDRB1	DDRB0
W								
Reset	0	0	0	0	0	0	0	0

Note:

111. Offset related to 0x0200 for blocking access and 0x300 for non blocking access within the global address space.

**Table 156. PTBC1 - Register Field Descriptions**

Field	Description
6-4 PUEB[2-0]	Pull-up Enable Port B[2...0] 0 - Pull-up disabled on PTBx pin. 1 - Pull-up enabled on PTBx pin.
2-0 DDRB[2-0]	Data Direction Port B[2...0] 0 - PTBx configured as input. 1 - PTBx configured as output.

#### NOTE

The pull-up resistor is not active once the port is configured as an output.

### 5.18.4.2 Port B Configuration Register 2 (PTBC2)

**Table 157. Port B Configuration Register 2 (PTBC2)**

Offset <sup>(112)</sup>	0x21							Access: User read/write
	7	6	5	4	3	2	1	0
R	0	0	0	0	PWMCS	PWMEN	SERMOD	
W								
Reset	0	0	0	0	0	0	0	0

Note:

112. Offset related to 0x0200 for blocking access and 0x300 for non blocking access within the global address space.

Table 169. OC3D - Register Field Descriptions

Field	Description
3-0 OC3D[3-0]	Output Compare 3 Data for Channel "n"

**NOTE**

A channel 3 output compare will cause bits in the output compare 3 data register to transfer to the timer port data register if the corresponding output compare 3 mask register bits are set.

**5.19.3.3.5 Timer Count Register (TCNT)**

Table 170. Timer Count Register (TCNT)

Offset <sup>(121)</sup> 0xC4, 0xC5		Access: User read(anytime)/write (special mode)							
		15	14	13	12	11	10	9	8
R		tcnt15	tcnt14	tcnt13	tcnt12	tcnt11	tcnt10	tcnt9	tcnt8
W									
Reset		0	0	0	0	0	0	0	0
		7	6	5	4	3	2	1	0
R		tcnt7	tcnt6	tcnt5	tcnt4	tcnt3	tcnt2	tcnt1	tcnt0
W									
Reset		0	0	0	0	0	0	0	0

Note:

121. Offset related to 0x0200 for blocking access and 0x300 for non blocking access within the global address space.

Table 171. TCNT - Register Field Descriptions

Field	Description
15-0 tcnt[15-0]	16 Bit Timer Count Register

**NOTE**

The 16-bit main timer is an up counter. A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word. The period of the first count after a write to the TCNT registers may be a different length because the write is not synchronized with the prescaler clock.

**5.19.3.3.6 Timer System Control Register 1 (TSCR1)**

Table 172. Timer System Control Register 1 (TSCR1)

Offset <sup>(122)</sup> 0xC6		Access: User read/write							
		7	6	5	4	3	2	1	0
R		TEN	0	0	TFFCA	0	0	0	0
W									
Reset		0	0	0	0	0	0	0	0

Note:

122. Offset related to 0x0200 for blocking access and 0x300 for non blocking access within the global address space.

**Table 198. Analog Digital Converter Module - Memory Map**

Register / Offset <sup>(138)</sup>		Bit 7	6	5	4	3	2	1	Bit 0
0x8F	R	ADR4[1:0]							
ADR4 (lo)	W								
0x90	R	ADR5[9:2]							
ADR5 (hi)	W								
0x91	R	ADR5[1:0]							
ADR5 (lo)	W								
0x92	R	ADR6[9:2]							
ADR6 (hi)	W								
0x93	R	ADR6[1:0]							
ADR6 (lo)	W								
0x94	R	ADR7[9:2]							
ADR7 (hi)	W								
0x95	R	ADR7[1:0]							
ADR7 (lo)	W								
0x96	R	ADR8[9:2]							
ADR8 (hi)	W								
0x97	R	ADR8[1:0]							
ADR8 (lo)	W								
0x98	R	ADR9[9:2]							
ADR9 (hi)	W								
0x99	R	ADR9[1:0]							
ADR9 (lo)	W								
0x9A	R	ADR10[9:2]							
ADR10 (hi)	W								
0x9B	R	ADR10[1:0]							
ADR10 (lo)	W								
0x9C	R	ADR11[9:2]							
ADR11 (hi)	W								
0x9D	R	ADR11[1:0]							
ADR11 (lo)	W								
0x9E	R	ADR12[9:2]							
ADR12 (hi)	W								
0x9F	R	ADR12[1:0]							
ADR12 (lo)	W								
0xA0	R								
Reserved	W								
0xA1	R								
Reserved	W								
0xA2	R	ADR14[9:2]							
ADR14 (hi)	W								
0xA3	R	ADR14[1:0]							
ADR14 (lo)	W								



Table 217. CTR1 - Register Field Descriptions

Field	Description
7 BGTRE	Bandgap trim enable 0 - no trim can be done 1 - trim can be done by setting BGTRIMUP and BGTRIMDN bits
6 CTR1_6	Spare Trim Bit
5 BGTRIMUP	Bandgap trim up bit 0 - default slope 1 - increase bandgap slope
4 BGTRIMDN	Bandgap trim down bit 0 - default slope 1 - decrease bandgap slope
3 IREFTRE	Iref trim enable bit 0 - no trim can be done 1 - trim can be done by setting IREFTR[2:0] bits
2-0 IREFTR2...0	Iref trim - This trim is used to adjust the internal zero TC current reference 000: 0% 001: +7.6% 010: +16.43% 011: +26.83% 100: -8.54% 101: -15.75% 110: -21.79% 111: 0%

## 5.26.1.2.3 Trimming Register 2 (CTR2)

Table 218. Trimming Register 2 (CTR2)

Offset<sup>(152)</sup> 0xF2

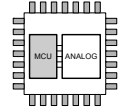
Access: User read/write

	7	6	5	4	3	2	1	0
R	0	0	0	SLPBGTRE	SLPBG_LOCK	SLPBGTR2	SLPBGTR1	SLPBGTR0
W								
Reset	0	0	0	0	0	0	0	0

Note:

152. Offset related to 0x0200 for blocking access and 0x300 for non blocking access within the global address space.

## 5.27 MM912\_634 - MCU Die Overview



### 5.27.1 Introduction

The MC9S12I64 micro controller implemented in the MM912\_634 is designed as counter part to an analog die, and is not being offered as a standalone MCU.

The MC9S12I64 device contains a S12 Central Processing Unit (CPU), offers 64kB of Flash memory and 6.0 kB of system SRAM, up to eight general purpose I/Os, an on-chip oscillator and clock multiplier, one Serial Peripheral Interface (SPI), an interrupt module and debug capabilities via the on-chip debug module (DBG) in combination with the Background Debug Mode (BDM) interface. Additionally there is a die-to-die initiator (D2DI) which represents the communication interface to the companion (analog) die.

### 5.27.2 Features

This section describes the key features of the MC9S12I64 micro controller unit.

#### 5.27.2.1 Chip-Level Features

On-chip modules available within the family include the following features:

- S12 CPU core (CPU12\_V1)
- Kbyte on-chip flash with ECC
- 4.0 kbyte on-chip data flash with ECC
- 6.0 kbyte on-chip SRAM
- Phase locked loop (IPLL) frequency multiplier with internal filter
- 4–16 MHz amplitude controlled Pierce oscillator
- 1.0 MHz internal RC oscillator
- One serial peripheral interface (SPI) module
- On-chip voltage regulator (VREG) for regulation of input supply and all internal voltages
- Die to Die Initiator (D2DI)

### 5.27.3 Module Features

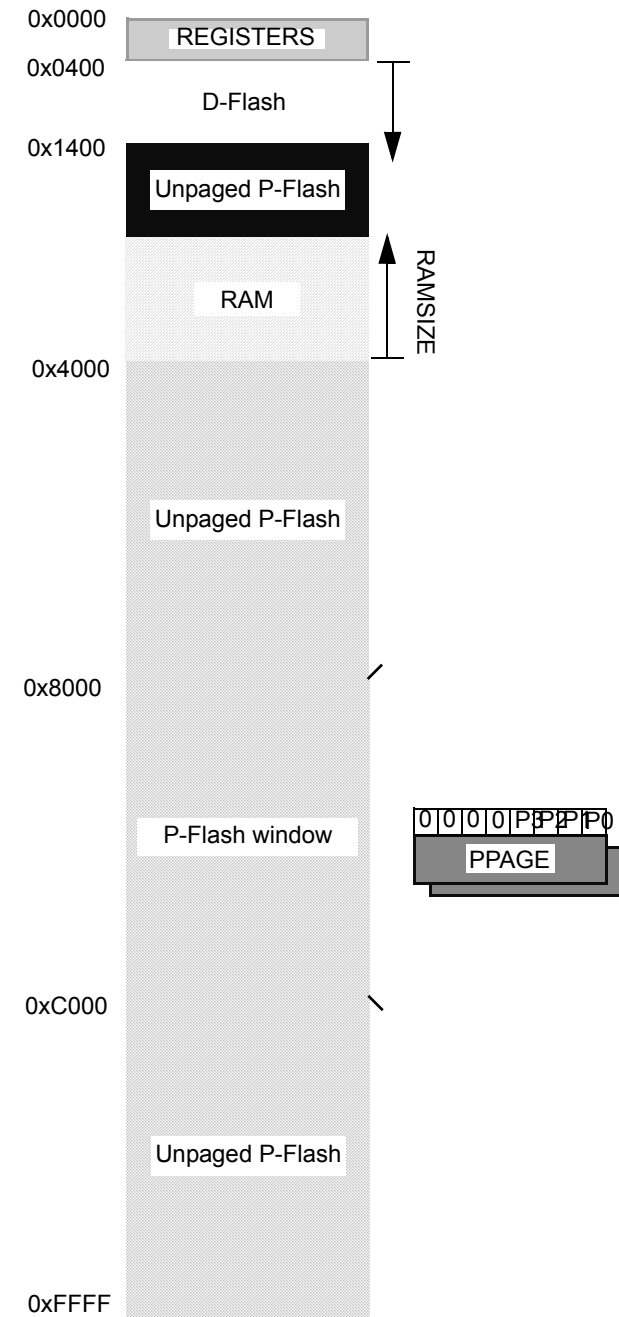
The following sections provide more details of the modules implemented on the MC9S12I64.

#### 5.27.3.1 S12 16-Bit Central Processor Unit (CPU)

S12 CPU is a high-speed 16-bit processing unit:

- Full 16-bit data paths supports efficient arithmetic operation and high-speed math execution
- Includes many single-byte instructions. This allows much more efficient use of ROM space.
- Extensive set of indexed addressing capabilities, including:
  - Using the stack pointer as an indexing register in all indexed operations
  - Using the program counter as an indexing register in all but auto increment/decrement mode
  - Accumulator offsets using A, B, or D accumulators
  - Automatic index pre-decrement, pre-increment, post-decrement, and post-increment (by –8 to +8)

CPU and BDM  
Local Memory Map



Global Memory Map

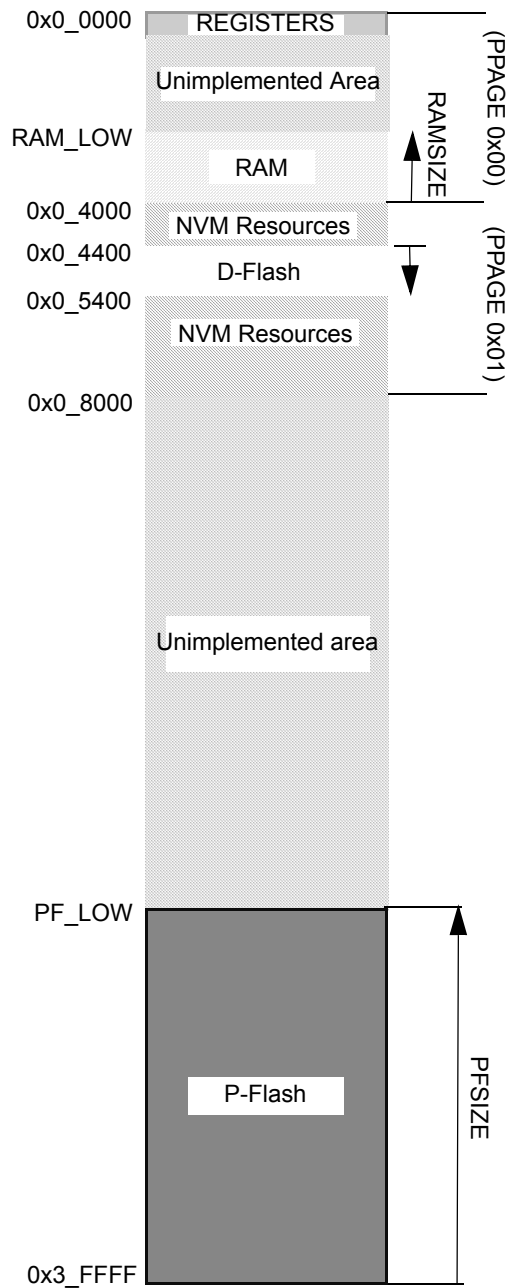


Figure 50. Implemented Global Address Mapping

5.29.5.1 Chip Bus Control

The S12PMMC controls the address buses and the data buses that interface the bus masters (CPU12, S12SBDM) with the rest of the system (master buses). In addition the MMC handles all CPU read data bus swapping operations. All internal resources are connected to specific target buses (see Figure 51).

### 5.31.4.7 Serial Interface Hardware Handshake Protocol

BDM commands that require CPU execution are ultimately treated at the MCU bus rate. Since the BDM clock source can be modified, it is very helpful to provide a handshake protocol in which the host could determine when an issued command is executed by the CPU. The alternative is to always wait the amount of time equal to the appropriate number of cycles at the slowest possible rate the clock could be running. This sub-section will describe the hardware handshake protocol.

The hardware handshake protocol signals to the host controller when an issued command was successfully executed by the target. This protocol is implemented by a 16 serial clock cycle low pulse followed by a brief speedup pulse in the BKGD pin. This pulse is generated by the target MCU when a command, issued by the host, has been successfully executed (see Figure 58). This pulse is referred to as the ACK pulse. After the ACK pulse has finished: the host can start the bit retrieval if the last issued command was a read command, or start a new command if the last command was a write command or a control command (BACKGROUND, GO, GO\_UNTIL(182) or TRACE1). The ACK pulse is not issued earlier than 32 serial clock cycles after the BDM command was issued. The end of the BDM command is assumed to be the 16th tick of the last bit. This minimum delay assures enough time for the host to perceive the ACK pulse. Note also that, there is no upper limit for the delay between the command and the related ACK pulse, since the command execution depends upon the CPU bus, which in some cases could be very slow due to long accesses taking place. This protocol allows a great flexibility for the POD designers, since it does not rely on any accurate time measurement or short response time to any event in the serial communication.

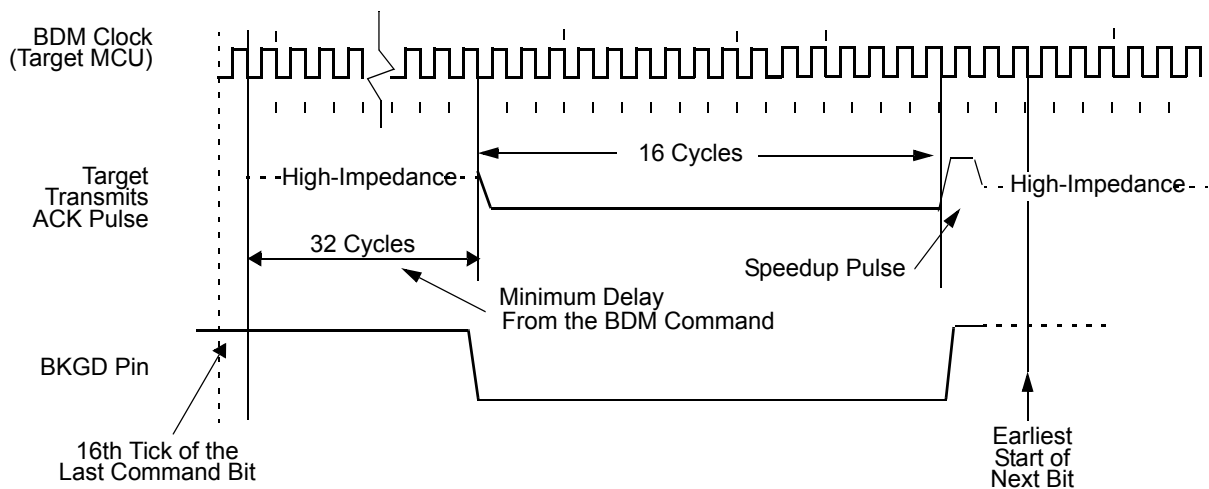


Figure 58. Target Acknowledge Pulse (ACK)

#### NOTE

If the ACK pulse was issued by the target, the host assumes the previous command was executed. If the CPU enters stop prior to executing a hardware command, the ACK pulse will not be issued meaning that the BDM command was not executed. After entering stop mode, the BDM command is no longer pending.

Figure 59 shows the ACK handshake protocol in a command level timing diagram. The READ\_BYTE instruction is used as an example. First, the 8-bit instruction opcode is sent by the host, followed by the address of the memory location to be read. The target BDM decodes the instruction. A bus cycle is grabbed (free or stolen) by the BDM and it executes the READ\_BYTE operation. Having retrieved the data, the BDM issues an ACK pulse to the host controller, indicating that the addressed byte is ready to be retrieved. After detecting the ACK pulse, the host initiates the byte retrieval process. Note that data is sent in the form of a word and the host needs to determine which is the appropriate byte based on whether the address was odd or even.

## 5.32.2 External Signal Description

There are no external signals associated with this module.

## 5.32.3 Memory Map and Registers

### 5.32.3.1 Module Memory Map

A summary of the registers associated with the DBG sub-block is shown in Figure 270. Detailed descriptions of the registers and bits are given in the subsections that follow.

**Table 270. Quick Reference to DBG Registers**

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0020	DBGC1	R	ARM	0	0	BDM	DBGBRK	0	COMRV	
		W		TRIG						
0x0021	DBGSR	R	TBF <sup>(184)</sup>	0	0	0	0	SSF2	SSF1	SSF0
		W								
0x0022	DBGTCR	R	0	TSOURCE	0	0	TRCMOD		0	TALIGN
		W								
0x0023	DBGC2	R	0	0	0	0	0	0	ABCM	
		W								
0x0024	DBGTBH	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0025	DBGTBL	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0026	DBGCNT	R	TBF	0	CNT					
		W								
0x0027	DBGSCRX	R	0	0	0	0	SC3	SC2	SC1	SC0
		W								
0x0027	DBGMFR	R	0	0	0	0	0	MC2	MC1	MC0
		W								
0x0028	DBGACTL	R	SZE	SZ	TAG	BRK	RW	RWE	NDB	COMPE
		W								
0x0028	DBGBCTL	R	SZE	SZ	TAG	BRK	RW	RWE	0	COMPE
		W								
0x0028	DBGCCTL	R	0	0	TAG	BRK	RW	RWE	0	COMPE
		W								
0x0029	DBGXAH	R	0	0	0	0	0	0	Bit 17	Bit 16
		W								
0x002A	DBGXAM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002B	DBGXAL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002C	DBGADH	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002D	DBGADL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

**Table 356. RTI Frequency Divide Rates for RTDEC=1**

RTR[3:0]	RTR[6:4] =							
	000 (1x10 <sup>3</sup> )	001 (2x10 <sup>3</sup> )	010 (5x10 <sup>3</sup> )	011 (10x10 <sup>3</sup> )	100 (20x10 <sup>3</sup> )	101 (50x10 <sup>3</sup> )	110 (100x10 <sup>3</sup> )	111 (200x10 <sup>3</sup> )
0010 (÷3)	3x10 <sup>3</sup>	6x10 <sup>3</sup>	15x10 <sup>3</sup>	30x10 <sup>3</sup>	60x10 <sup>3</sup>	150x10 <sup>3</sup>	300x10 <sup>3</sup>	600x10 <sup>3</sup>
0011 (÷4)	4x10 <sup>3</sup>	8x10 <sup>3</sup>	20x10 <sup>3</sup>	40x10 <sup>3</sup>	80x10 <sup>3</sup>	200x10 <sup>3</sup>	400x10 <sup>3</sup>	800x10 <sup>3</sup>
0100 (÷5)	5x10 <sup>3</sup>	10x10 <sup>3</sup>	25x10 <sup>3</sup>	50x10 <sup>3</sup>	100x10 <sup>3</sup>	250x10 <sup>3</sup>	500x10 <sup>3</sup>	1x10 <sup>6</sup>
0101 (÷6)	6x10 <sup>3</sup>	12x10 <sup>3</sup>	30x10 <sup>3</sup>	60x10 <sup>3</sup>	120x10 <sup>3</sup>	300x10 <sup>3</sup>	600x10 <sup>3</sup>	1.2x10 <sup>6</sup>
0110 (÷7)	7x10 <sup>3</sup>	14x10 <sup>3</sup>	35x10 <sup>3</sup>	70x10 <sup>3</sup>	140x10 <sup>3</sup>	350x10 <sup>3</sup>	700x10 <sup>3</sup>	1.4x10 <sup>6</sup>
0111 (÷8)	8x10 <sup>3</sup>	16x10 <sup>3</sup>	40x10 <sup>3</sup>	80x10 <sup>3</sup>	160x10 <sup>3</sup>	400x10 <sup>3</sup>	800x10 <sup>3</sup>	1.6x10 <sup>6</sup>
1000 (÷9)	9x10 <sup>3</sup>	18x10 <sup>3</sup>	45x10 <sup>3</sup>	90x10 <sup>3</sup>	180x10 <sup>3</sup>	450x10 <sup>3</sup>	900x10 <sup>3</sup>	1.8x10 <sup>6</sup>
1001 (÷10)	10 x10 <sup>3</sup>	20x10 <sup>3</sup>	50x10 <sup>3</sup>	100x10 <sup>3</sup>	200x10 <sup>3</sup>	500x10 <sup>3</sup>	1x10 <sup>6</sup>	2x10 <sup>6</sup>
1010 (÷11)	11 x10 <sup>3</sup>	22x10 <sup>3</sup>	55x10 <sup>3</sup>	110x10 <sup>3</sup>	220x10 <sup>3</sup>	550x10 <sup>3</sup>	1.1x10 <sup>6</sup>	2.2x10 <sup>6</sup>
1011 (÷12)	12x10 <sup>3</sup>	24x10 <sup>3</sup>	60x10 <sup>3</sup>	120x10 <sup>3</sup>	240x10 <sup>3</sup>	600x10 <sup>3</sup>	1.2x10 <sup>6</sup>	2.4x10 <sup>6</sup>
1100 (÷13)	13x10 <sup>3</sup>	26x10 <sup>3</sup>	65x10 <sup>3</sup>	130x10 <sup>3</sup>	260x10 <sup>3</sup>	650x10 <sup>3</sup>	1.3x10 <sup>6</sup>	2.6x10 <sup>6</sup>
1101 (÷14)	14x10 <sup>3</sup>	28x10 <sup>3</sup>	70x10 <sup>3</sup>	140x10 <sup>3</sup>	280x10 <sup>3</sup>	700x10 <sup>3</sup>	1.4x10 <sup>6</sup>	2.8x10 <sup>6</sup>
1110 (÷15)	15x10 <sup>3</sup>	30x10 <sup>3</sup>	75x10 <sup>3</sup>	150x10 <sup>3</sup>	300x10 <sup>3</sup>	750x10 <sup>3</sup>	1.5x10 <sup>6</sup>	3x10 <sup>6</sup>
1111 (÷16)	16x10 <sup>3</sup>	32x10 <sup>3</sup>	80x10 <sup>3</sup>	160x10 <sup>3</sup>	320x10 <sup>3</sup>	800x10 <sup>3</sup>	1.6x10 <sup>6</sup>	3.2x10 <sup>6</sup>

**5.38.3.2.9 S12CPMU COP Control Register (CPMUCOP)**

This register controls the COP (Computer Operating Properly) watchdog.

The clock source for the COP is either IRCCLK or OSCCLK depending on the setting of the COPOSCSEL bit. In Stop Mode with PSTP=1(Pseudo Stop Mode), COPOSCSEL=1 and PCE=1 the COP continues to run, else the COP counter halts in Stop Mode.

**Table 357. S12CPMU COP Control Register (CPMUCOP)**

0x003C

	7	6	5	4	3	2	1	0
R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
W			WRTMASK					
Reset	F	0	0	0	0	F	F	F
Reset	0	0				0	0	0

= Unimplemented or Reserved

Note:

197. After de-assert of System Reset the values are automatically loaded from the Flash memory. See Device specification for details.

Read: Anytime

Write:

1. RSBCK: anytime in Special Mode; write to “1” but not to “0” in Normal Mode
2. WCOP, CR2, CR1, CR0:
  - Anytime in Special Mode, when WRTMASK is 0, otherwise it has no effect
  - Write once in Normal Mode, when WRTMASK is 0, otherwise it has no effect.
    - Writing CR[2:0] to “000” has no effect, but counts for the “write once” condition.
    - Writing WCOP to “0” has no effect, but counts for the “write once” condition.


### 5.38.3.2.14 Low Voltage Control Register (CPMULVCTL)

The CPMULVCTL register allows the configuration of the low-voltage detect features.

**Table 367. Low Voltage Control Register (CPMULVCTL)**

0x02F1

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	LVDS	LVIE	LVIF
W								
Reset	0	0	0	0	0	U	0	U

 = Unimplemented or Reserved

Note:

198. The Reset state of LVDS and LVIF depends on the external supplied VDDA level

Read: Anytime

Write: LVIE and LVIF are write anytime, LVDS is read only

**Table 368. CPMULVCTL Field Descriptions**

Field	Description
2 LVDS	<b>Low-voltage Detect Status Bit</b> — This read-only status bit reflects the voltage level on VDDRX. Writes have no effect. 0 Input voltage VDDRX is above level $V_{LVID}$ or RPM. 1 Input voltage VDDRX is below level $V_{LVIA}$ and FPM.
1 LVIE	<b>Low-voltage Interrupt Enable Bit</b> 0 Interrupt request is disabled. 1 Interrupt will be requested whenever LVIF is set.
0 LVIF	<b>Low-voltage Interrupt Flag</b> — LVIF is set to 1 when LVDS status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (LVIE = 1), LVIF causes an interrupt request. 0 No change in LVDS bit. 1 LVDS bit has changed.


### 5.38.3.2.15 Autonomous Periodical Interrupt Control Register (CPMUAPICTL)

The CPMUAPICTL register allows the configuration of the autonomous periodical interrupt features.

**Table 369. Autonomous Periodical Interrupt Control Register (CPMUAPICTL)**

0x02F2

	7	6	5	4	3	2	1	0
R		0	0	APIES	APIEA	APIFE	APIE	APIF
W	APICLK							
Reset	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Read: Anytime

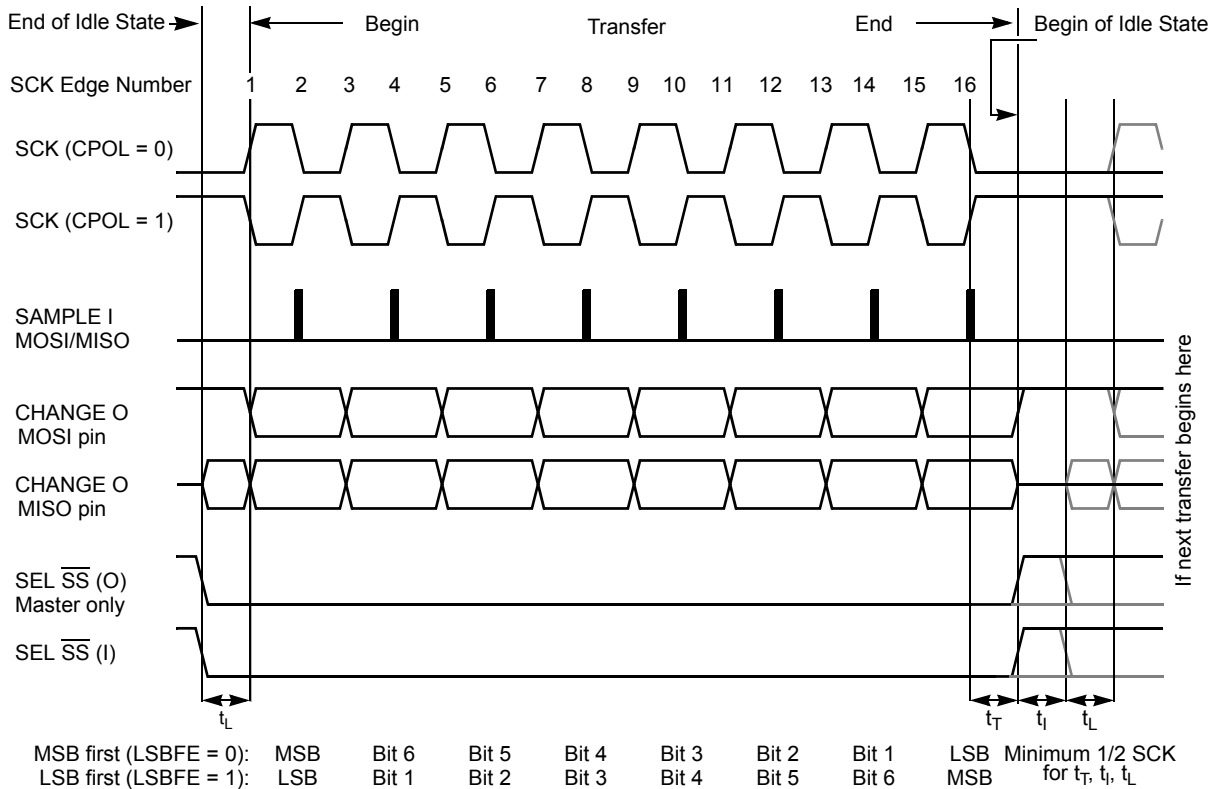
Write: Anytime

Data reception is double buffered, data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After  $2n$  SCK edges:

- Data that was previously in the SPI data register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

Figure 102 shows two clocking variations for  $CPHA = 1$ . The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The  $\overline{SS}$  line is the slave select input to the slave. The  $\overline{SS}$  pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.



$t_L$  = Minimum leading time before the first SCK edge, not required for back-to-back transfers

$t_T$  = Minimum trailing time after the last SCK edge

$t_i$  = Minimum idling time between transfers (minimum  $\overline{SS}$  high time), not required for back-to-back transfers

**Figure 102. SPI Clock Format 1 (CPHA = 1), with 8-Bit Transfer Width Selected (XFRW = 0)**



### 5.39.4.6 Error Conditions

The SPI has one error condition:

Mode fault error

#### 5.39.4.6.1 Mode Fault Error

If the  $\overline{SS}$  input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SCK lines simultaneously. This condition is not permitted in normal operation, the MODF bit in the SPI status register is set automatically, provided the MODFEN bit is set.

In the special case where the SPI is in master mode and MODFEN bit is cleared, the  $\overline{SS}$  pin is not used by the SPI. In this special case, the mode fault error function is inhibited and MODF remains cleared. In case the SPI system is configured as a slave, the  $\overline{SS}$  pin is a dedicated input pin. Mode fault error doesn't occur in slave mode.

If a mode fault error occurs, the SPI is switched to slave mode, with the exception that the slave output buffer is disabled. So SCK, MISO, and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver. A transmission in progress is aborted and the SPI is forced into idle state.

If the mode fault error occurs in the bidirectional mode for a SPI system configured in master mode, output enable of the MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for SPI system configured in slave mode.

The mode fault flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to SPI control register 1. If the mode fault flag is cleared, the SPI becomes a normal master or slave again.

#### NOTE

If a mode fault error occurs and a received data byte is pending in the receive shift register, this data byte will be lost.

### 5.39.4.7 Low Power Mode Options

#### 5.39.4.7.1 SPI in Run Mode

In run mode with the SPI system enable (SPE) bit in the SPI control register clear, the SPI system is in a low-power, disabled state. SPI registers remain accessible, but clocks to the core of this module are disabled.

#### 5.39.4.7.2 SPI in Stop Mode

Stop mode is dependent on the system. The SPI enters stop mode when the module clock is disabled (held high or low). If the SPI is in master mode and exchanging data when the CPU enters stop mode, the transmission is frozen until the CPU exits stop mode. After stop, data to and from the external SPI is exchanged correctly. In slave mode, the SPI will stay synchronized with the master.

The stop mode is not dependent on the SPISWAI bit.

#### 5.39.4.7.3 Reset

The reset values of registers and signals are described in Section 5.39.3, "Memory Map and Register Definition", which details the registers and their bit fields.

- If a data transmission occurs in slave mode after reset without a write to SPIDR, it will transmit garbage, or the data last received from the master before the reset.
- Reading from the SPIDR after reset will always read zeros.

#### 5.39.4.7.4 Interrupts

The SPI only originates interrupt requests when SPI is enabled (SPE bit in SPICR1 set). The following is a description of how the SPI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt priority are chip dependent.

The interrupt flags MODF, SPIF, and SPTEF are logically ORed to generate an interrupt request.

**Table 420. Flash KEYEN States**

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED <sup>(217)</sup>
10	ENABLED
11	DISABLED

Note:

217. Preferred KEYEN state to disable backdoor key access.

**Table 421. Flash Security States**

SEC[1:0]	Status of Security
00	SECURED
01	SECURED <sup>(218)</sup>
10	UNSECURED
11	SECURED

Note:

218. Preferred SEC state to set MCU to secured state


The security function in the Flash module is described in Section 5.40.5, "Security".

#### 5.40.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for Flash memory operations.

**Table 422. FCCOB Index Register (FCCOBIX)**

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCOBIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

**Table 423. FCCOBIX Field Descriptions**


Field	Description
2-0 CCOBIX[1:0]	<b>Common Command Register Index</b> — The CCOBIX bits are used to select which word of the FCCOB register array is being read or written to. See Section 5.40.3.2.12, "Flash Common Command Object Register (FCCOB)", for more details.

#### 5.40.3.2.4 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

**Table 424. Flash Reserved0 Register (FRSV0)**

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

 = Unimplemented or Reserved


All bits in the FRSV0 register read 0 and are not writable.

### 5.40.3.2.18 Flash Reserved5 Register (FRSV5)

This Flash register is reserved for factory testing.

**Table 451. Flash Reserved5 Register (FRSV5)**

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

 = Unimplemented or Reserved


All bits in the FRSV5 register read 0 and are not writable.

### 5.40.3.2.19 Flash Reserved6 Register (FRSV6)

This Flash register is reserved for factory testing.

**Table 452. Flash Reserved6 Register (FRSV6)**

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

 = Unimplemented or Reserved


All bits in the FRSV6 register read 0 and are not writable.

### 5.40.3.2.20 Flash Reserved7 Register (FRSV7)

This Flash register is reserved for factory testing.

**Table 453. Flash Reserved7 Register (FRSV7)**

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

 = Unimplemented or Reserved

All bits in the FRSV7 register read 0 and are not writable.

## 5.40.4 Functional Description

### 5.40.4.1 Modes of Operation

The FTMRC64K1 module provides the modes of operation shown in Table 454. The operating mode is determined by module-level inputs and affects the FCLKDIV, FCNFG, and DFPROT registers, Scratch RAM writes, and the command set availability (see Table 456).

**Table 454. Modes and Mode Control Inputs**

Operating Mode	FTMRC Input
	mmc_mode_ss_t2
Normal:	0
Special:	1

**Table 457. P-Flash Commands**

FCMD	Command	Function on P-Flash Memory
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all P-Flash (and D-Flash) blocks and verifying that all P-Flash (and D-Flash) blocks are erased.
0x0C	Verify Backdoor Access Key	Supports a method of releasing MCU security by verifying a set of security keys.
0x0D	Set User Margin Level	Specifies a user margin read level for all P-Flash blocks.
0x0E	Set Field Margin Level	Specifies a field margin read level for all P-Flash blocks (special modes only).

**5.40.4.3.5 D-Flash Commands**

Table 458 summarizes the valid D-Flash commands along with the effects of the commands on the D-Flash block.

**Table 458. D-Flash Commands**

FCMD	Command	Function on D-Flash Memory
0x01	Erase Verify All Blocks	Verify that all D-Flash (and P-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that the D-Flash block is erased.
0x08	Erase All Blocks	Erase all D-Flash (and P-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the DPOPEN bit in the DFPROT register are set prior to launching the command.
0x09	Erase Flash Block	Erase a D-Flash (or P-Flash) block. An erase of the full D-Flash block is only possible when DPOPEN bit in the DFPROT register is set prior to launching the command.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all D-Flash (and P-Flash) blocks and verifying that all D-Flash (and P-Flash) blocks are erased.
0x0D	Set User Margin Level	Specifies a user margin read level for the D-Flash block.
0x0E	Set Field Margin Level	Specifies a field margin read level for the D-Flash block (special modes only).
0x10	Erase Verify D-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x11	Program D-Flash	Program up to four words in the D-Flash block.
0x12	Erase D-Flash Sector	Erase all bytes in a sector of the D-Flash block.

**5.40.4.4 Allowed Simultaneous P-Flash and D-Flash Operations**

Only the operations marked 'OK' in Table 459 are permitted to be run simultaneously on the Program Flash and Data Flash blocks. Some operations cannot be executed simultaneously because certain hardware resources are shared by the two memories. The priority has been placed on permitting Program Flash reads while program and erase operations execute on the Data Flash, providing read (P-Flash) while write (D-Flash) functionality.

**Table 459. Allowed P-Flash and D-Flash Simultaneous Operations**

Program Flash	Data Flash				
	Read	Margin Read <sup>(227)</sup>	Program	Sector Erase	Mass Erase <sup>(229)</sup>
Read		OK	OK	OK	
Margin Read <sup>(227)</sup>		OK <sup>(228)</sup>			
Program					
Sector Erase				OK	

## 5.41 Die-to-Die Initiator (D2DIV1)

### 5.41.0.1 Preface

This document contains the user specification of the D2D Initiator.

#### 5.41.0.1.1 Acronyms and Abbreviations

Table 495 contains sample acronyms and abbreviations used in this document.

**Table 495. Acronyms and Abbreviated Terms**

Term	Meaning
D2D	Die-to-Die

#### 5.41.0.1.2 Glossary

Table 496 shows a glossary of the major terms used in this document.

**Table 496. Glossary**

Term	Definition
Active low	The signal is asserted when it changes to logic-level zero.
Active high	The signal is asserted when it changes to logic-level one.
Asserted	Discrete signal is in active logic state.
Customer	The end user of an SoC design or device.
EOT	End of Transaction
Negated	A discrete signal is in inactive logic state.
Pin	External physical connection.
Revision	Revised or new version of a document. Revisions produce versions; there can be no 'Rev 0.0.'
Signal	Electronic construct whose state or change in state conveys information.
Transfer	A read or write on the CPU bus following the IP-Bus protocol.
Transaction	Command, address and if required data sent on the D2D interface. A transaction is finished by the EOT acknowledge cycle.
Version	Particular form or variation of an earlier or original document.

### 5.41.1 Introduction

This section describes the functionality of the die-to-die (D2DIV1) initiator block especially designed for low cost connections between a microcontroller die (Interface Initiator) and an analog die (Interface Target) located in the same package.

The D2DI block

- realizes the initiator part of the D2D interface, including supervision and error interrupt generation
- generates the clock for this interface
- disables/enables the interrupt from the D2D interface

#### 5.41.1.1 Overview

The D2DI is the initiator for a data transfer to and from a target typically located on another die in the same package. It provides a set of configuration registers and two memory mapped 256 Byte address windows. When writing to a window a transaction is initiated sending a write command, followed by an 8-bit address and the data byte or word to the target. When reading from a window a transaction is initiated sending a read command, followed by an 8-bit address to the target. The target then responds with the data. The basic idea is that a peripheral located on another die, can be addressed like an on-chip peripheral, except for a small transaction delay.