



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

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

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

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

Details

Product Status	Active
Core Processor	HCS12X
Core Size	16-Bit
Speed	80MHz
Connectivity	CANbus, EBI/EMI, I ² C, IrDA, LINbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	59
Program Memory Size	256KB (256K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	2.35V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-QFP
Supplier Device Package	80-QFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s12xdt256caa

Chapter 2

Clocks and Reset Generator (S12CRGV6)

2.1 Introduction

This specification describes the function of the clocks and reset generator (CRG).

2.1.1 Features

The main features of this block are:

- Phase locked loop (PLL) frequency multiplier
 - Reference divider
 - Automatic bandwidth control mode for low-jitter operation
 - Automatic frequency lock detector
 - Interrupt request on entry or exit from locked condition
 - Self clock mode in absence of reference clock
- System clock generator
 - Clock quality check
 - User selectable fast wake-up from Stop in self-clock mode for power saving and immediate program execution
 - Clock switch for either oscillator or PLL based system clocks
- Computer operating properly (COP) watchdog timer with time-out clear window
- System reset generation from the following possible sources:
 - Power on reset
 - Low voltage reset
 - Illegal address reset
 - COP reset
 - Loss of clock reset
 - External pin reset
- Real-time interrupt (RTI)

Table 4-17. Analog Input Channel Select Coding

CD	CC	CB	CA	Analog Input Channel
0	0	0	0	AN0
0	0	0	1	AN1
0	0	1	0	AN2
0	0	1	1	AN3
0	1	0	0	AN4
0	1	0	1	AN5
0	1	1	0	AN6
0	1	1	1	AN7
1	0	0	0	AN8
1	0	0	1	AN9
1	0	1	0	AN10
1	0	1	1	AN11
1	1	0	0	AN12
1	1	0	1	AN13
1	1	1	0	AN14
1	1	1	1	AN15

4.3.2.16 ATD Conversion Result Registers (ATDDR_x)

The A/D conversion results are stored in 16 read-only result registers. The result data is formatted in the result registers based on two criteria. First there is left and right justification; this selection is made using the DJM control bit in ATDCTL5. Second there is signed and unsigned data; this selection is made using the DSGN control bit in ATDCTL5. Signed data is stored in 2's complement format and only exists in left justified format. Signed data selected for right justified format is ignored.

Read: Anytime

Write: Anytime in special mode, unimplemented in normal modes

4.3.2.16.1 Left Justified Result Data

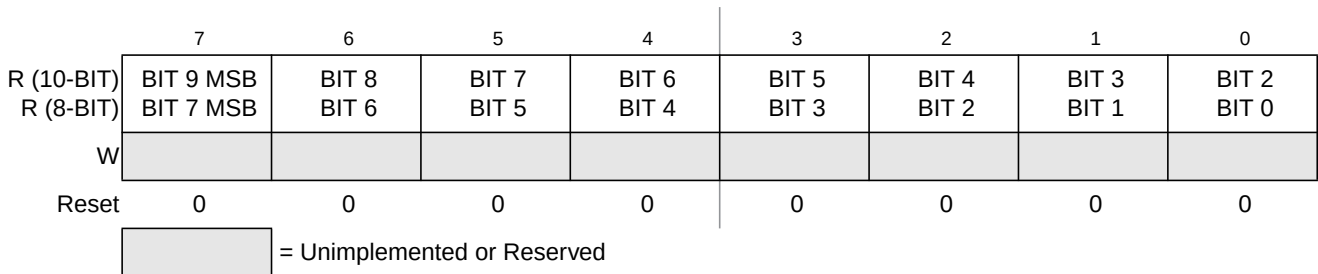


Figure 4-18. Left Justified, ATD Conversion Result Register x, High Byte (ATDDR_xH)

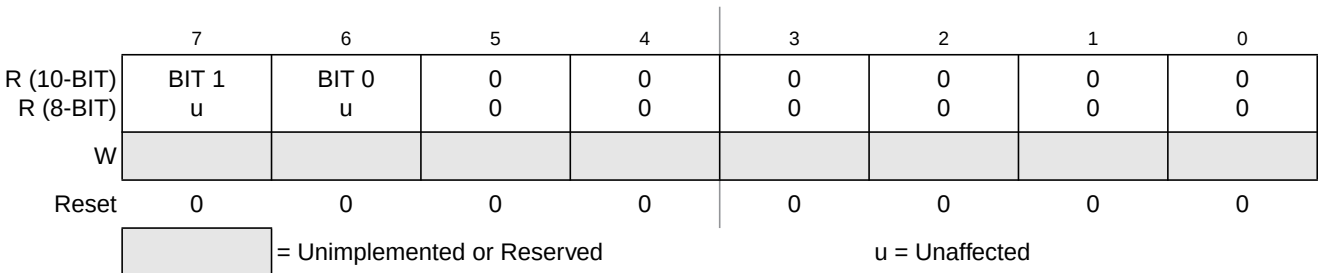


Figure 4-19. Left Justified, ATD Conversion Result Register x, Low Byte (ATDDR_xL)

In either level or edge triggered modes, the first conversion begins when the trigger is received. In both cases, the maximum latency time is one bus clock cycle plus any skew or delay introduced by the trigger circuitry.

After ETRIGE is enabled, conversions cannot be started by a write to ATDCTL5, but rather must be triggered externally.

If the level mode is active and the external trigger both de-asserts and re-asserts itself during a conversion sequence, this does not constitute an overrun. Therefore, the flag is not set. If the trigger remains asserted in level mode while a sequence is completing, another sequence will be triggered immediately.

4.4.2.2 General-Purpose Digital Input Port Operation

The input channel pins can be multiplexed between analog and digital data. As analog inputs, they are multiplexed and sampled to supply signals to the A/D converter. As digital inputs, they supply external input data that can be accessed through the digital port registers (PORTAD0 & PORTAD1) (input-only).

The analog/digital multiplex operation is performed in the input pads. The input pad is always connected to the analog inputs of the ATD10B16C. The input pad signal is buffered to the digital port registers. This buffer can be turned on or off with the ATDDIEN0 & ATDDIEN1 register. This is important so that the buffer does not draw excess current when analog potentials are presented at its input.

4.4.3 Operation in Low Power Modes

The ATD10B16C can be configured for lower MCU power consumption in three different ways:

- **Stop Mode**
Stop Mode: This halts A/D conversion. Exit from Stop mode will resume A/D conversion, But due to the recovery time the result of this conversion should be ignored.
Entering stop mode causes all clocks to halt and thus the system is placed in a minimum power standby mode. This halts any conversion sequence in progress. During recovery from stop mode, there must be a minimum delay for the stop recovery time t_{SR} before initiating a new ATD conversion sequence.
- **Wait Mode**
Wait Mode with AWAI = 1: This halts A/D conversion. Exit from Wait mode will resume A/D conversion, but due to the recovery time the result of this conversion should be ignored.
Entering wait mode, the ATD conversion either continues or halts for low power depending on the logical value of the AWAIT bit.
- **Freeze Mode**
Writing ADPU = 0 (Note that all ATD registers remain accessible.): This aborts any A/D conversion in progress.
In freeze mode, the ATD10B16C will behave according to the logical values of the FRZ1 and FRZ0 bits. This is useful for debugging and emulation.

NOTE

The reset value for the ADPU bit is zero. Therefore, when this module is reset, it is reset into the power down state.



COM

One's Complement

COM

Operation

$\sim RS \Rightarrow RD$ (translates to XNOR RD, R0, RS)
 $\sim RD \Rightarrow RD$ (translates to XNOR RD, R0, RD)
Performs a one's complement on a general purpose register.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

- N: Set if bit 15 of the result is set; cleared otherwise.
- Z: Set if the result is \$0000; cleared otherwise.
- V: 0; cleared.
- C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles		
COM RD, RS	TRI	0	0	0	1	0	RD	0	0	0	RS	1	1	P
COM RD	TRI	0	0	0	1	0	RD	0	0	0	RD	1	1	P

7.3.2.2 Timer Compare Force Register (CFORC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
Reset	0	0	0	0	0	0	0	0

Figure 7-4. Timer Compare Force Register (CFORC)

Read or write: Anytime but reads will always return 0x0000 (1 state is transient).

All bits reset to zero.

Table 7-3. CFORC Field Descriptions

Field	Description
7:0 FOC[7:0]	Force Output Compare Action for Channel 7:0 — A write to this register with the corresponding data bit(s) set causes the action which is programmed for output compare “x” to occur immediately. The action taken is the same as if a successful comparison had just taken place with the TCx register except the interrupt flag does not get set. Note: A successful channel 7 output compare overrides any channel 6:0 compares. If a forced output compare on any channel occurs at the same time as the successful output compare, then the forced output compare action will take precedence and the interrupt flag will not get set.

7.3.2.3 Output Compare 7 Mask Register (OC7M)

	7	6	5	4	3	2	1	0
R	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
W								
Reset	0	0	0	0	0	0	0	0

Figure 7-5. Output Compare 7 Mask Register (OC7M)

Read or write: Anytime

All bits reset to zero.

Table 7-4. OC7M Field Descriptions

Field	Description
7:0 OC7M[7:0]	Output Compare Mask Action for Channel 7:0 0 The corresponding OC7Dx bit in the output compare 7 data register will not be transferred to the timer port on a successful channel 7 output compare, even if the corresponding pin is setup for output compare. 1 The corresponding OC7Dx bit in the output compare 7 data register will be transferred to the timer port on a successful channel 7 output compare. Note: The corresponding channel must also be setup for output compare (IOSx = 1) for data to be transferred from the output compare 7 data register to the timer port.

9.7.1.4 Generation of STOP

A data transfer ends with a STOP signal generated by the 'master' device. A master transmitter can simply generate a STOP signal after all the data has been transmitted. The following is an example showing how a stop condition is generated by a master transmitter.

```

MASTX      TST      TXCNT      ;GET VALUE FROM THE TRANSMITING COUNTER
           BEQ      END        ;END IF NO MORE DATA
           BRSET    IBSR,#$01,END ;END IF NO ACK
           MOVB     DATABUF,IBDR ;TRANSMIT NEXT BYTE OF DATA
           DEC      TXCNT      ;DECREASE THE TXCNT
           BRA      EMAXTX      ;EXIT
END         BCLR     IBCR,#$20   ;GENERATE A STOP CONDITION
EMASTX      RTI              ;RETURN FROM INTERRUPT

```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data which can be done by setting the transmit acknowledge bit (TXAK) before reading the 2nd last byte of data. Before reading the last byte of data, a STOP signal must be generated first. The following is an example showing how a STOP signal is generated by a master receiver.

```

MASR      DEC      RXCNT      ;DECREASE THE RXCNT
           BEQ      ENMASR     ;LAST BYTE TO BE READ
           MOVB     RXCNT,D1    ;CHECK SECOND LAST BYTE
           DEC      D1         ;TO BE READ
           BNE      NXMAR      ;NOT LAST OR SECOND LAST
LAMAR      BSET     IBCR,#$08   ;SECOND LAST, DISABLE ACK
                               ;TRANSMITTING
           BRA      NXMAR
ENMASR     BCLR     IBCR,#$20   ;LAST ONE, GENERATE 'STOP' SIGNAL
NXMAR      MOVB     IBDR,RXBUF  ;READ DATA AND STORE
           RTI

```

9.7.1.5 Generation of Repeated START

At the end of data transfer, if the master continues to want to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. A program example is as shown.

```

RESTART    BSET     IBCR,#$04   ;ANOTHER START (RESTART)
           MOVB     CALLING,IBDR ;TRANSMIT THE CALLING ADDRESS;D0=R/W

```

9.7.1.6 Slave Mode

In the slave interrupt service routine, the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the transmit/receive mode select bit (Tx/Rx bit of IBCR) according to the R/W command bit (SRW). Writing to the IBCR clears the IAAS automatically. Note that the only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred, interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer may now be initiated by writing information to IBDR, for slave transmits, or dummy reading from IBDR, in slave receive mode. The slave will drive SCL low in-between byte transfers, SCL is released when the IBDR is accessed in the required mode.

Table 10-10. CANRIER Register Field Descriptions

Field	Description
7 WUPIE ¹	Wake-Up Interrupt Enable 0 No interrupt request is generated from this event. 1 A wake-up event causes a Wake-Up interrupt request.
6 CSCIE	CAN Status Change Interrupt Enable 0 No interrupt request is generated from this event. 1 A CAN Status Change event causes an error interrupt request.
5:4 RSTATE[1:0]	Receiver Status Change Enable — These RSTAT enable bits control the sensitivity level in which receiver state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the RSTAT flags continue to indicate the actual receiver state and are only updated if no CSCIF interrupt is pending. 00 Do not generate any CSCIF interrupt caused by receiver state changes. 01 Generate CSCIF interrupt only if the receiver enters or leaves “bus-off” state. Discard other receiver state changes for generating CSCIF interrupt. 10 Generate CSCIF interrupt only if the receiver enters or leaves “RxErr” or “bus-off” ² state. Discard other receiver state changes for generating CSCIF interrupt. 11 Generate CSCIF interrupt on all state changes.
3:2 TSTATE[1:0]	Transmitter Status Change Enable — These TSTAT enable bits control the sensitivity level in which transmitter state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level, the TSTAT flags continue to indicate the actual transmitter state and are only updated if no CSCIF interrupt is pending. 00 Do not generate any CSCIF interrupt caused by transmitter state changes. 01 Generate CSCIF interrupt only if the transmitter enters or leaves “bus-off” state. Discard other transmitter state changes for generating CSCIF interrupt. 10 Generate CSCIF interrupt only if the transmitter enters or leaves “TxErr” or “bus-off” state. Discard other transmitter state changes for generating CSCIF interrupt. 11 Generate CSCIF interrupt on all state changes.
1 OVRIE	Overrun Interrupt Enable 0 No interrupt request is generated from this event. 1 An overrun event causes an error interrupt request.
0 RXFIE	Receiver Full Interrupt Enable 0 No interrupt request is generated from this event. 1 A receive buffer full (successful message reception) event causes a receiver interrupt request.

¹ WUPIE and WUPE (see [Section 10.3.2.1, “MSCAN Control Register 0 \(CANCTL0\)”](#)) must both be enabled if the recovery mechanism from stop or wait is required.

² Bus-off state is defined by the CAN standard (see Bosch CAN 2.0A/B protocol specification: for only transmitters. Because the only possible state change for the transmitter from bus-off to TxOK also forces the receiver to skip its current state to RxOK, the coding of the RXSTAT[1:0] flags define an additional bus-off state for the receiver (see [Section 10.3.2.5, “MSCAN Receiver Flag Register \(CANRFLG\)”](#)).

10.3.2.7 MSCAN Transmitter Flag Register (CANTFLG)

The transmit buffer empty flags each have an associated interrupt enable bit in the CANTIER register.

10.3.2.10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAACK)

The CANTAACK register indicates the successful abort of a queued message, if requested by the appropriate bits in the CANTARQ register.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
W								
Reset:	0	0	0	0	0	0	0	0

11.4.1 Infrared Interface Submodule

This module provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the SCI. The IrDA physical layer specification defines a half-duplex infrared communication link for exchange data. The full standard includes data rates up to 16 Mbits/s. This design covers only data rates between 2.4 Kbits/s and 115.2 Kbits/s.

The infrared submodule consists of two major blocks: the transmit encoder and the receive decoder. The SCI transmits serial bits of data which are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses should be detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder (external from the MCU). The narrow pulses are then stretched by the infrared submodule to get back to a serial bit stream to be received by the SCI. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that uses active low pulses.

The infrared submodule receives its clock sources from the SCI. One of these two clocks are selected in the infrared submodule in order to generate either 3/16, 1/16, 1/32 or 1/4 narrow pulses during transmission. The infrared block receives two clock sources from the SCI, R16XCLK and R32XCLK, which are configured to generate the narrow pulse width during transmission. The R16XCLK and R32XCLK are internal clocks with frequencies 16 and 32 times the baud rate respectively. Both R16XCLK and R32XCLK clocks are used for transmitting data. The receive decoder uses only the R16XCLK clock.

11.4.1.1 Infrared Transmit Encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD pin. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent in the middle of the bit with a duration of 1/32, 1/16, 3/16 or 1/4 of a bit time. A narrow high pulse is transmitted for a zero bit when TXPOL is cleared, while a narrow low pulse is transmitted for a zero bit when TXPOL is set.

11.4.1.2 Infrared Receive Decoder

The infrared receive block converts data from the RXD pin to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow high pulse is expected for a zero bit when RXPOL is cleared, while a narrow low pulse is expected for a zero bit when RXPOL is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

11.4.2 LIN Support

This module provides some basic support for the LIN protocol. At first this is a break detect circuitry making it easier for the LIN software to distinguish a break character from an incoming data stream. As a further addition it supports a collision detection at the bit level as well as cancelling pending transmissions.

- Wait mode
MMC is functional during wait mode.
- Stop mode
MMC is inactive during stop mode.

17.1.2.2 Functional Modes

- Single chip modes
In normal and special single chip mode the internal memory is used. External bus is not active.
- Expanded modes
Address, data, and control signals are activated in normal expanded and special test modes when accessing the external bus.
- Emulation modes
External bus is active to emulate via an external tool the normal expanded or the normal single chip mode.

17.1.3 Block Diagram

Figure 1-1 shows a block diagram of the MMC.

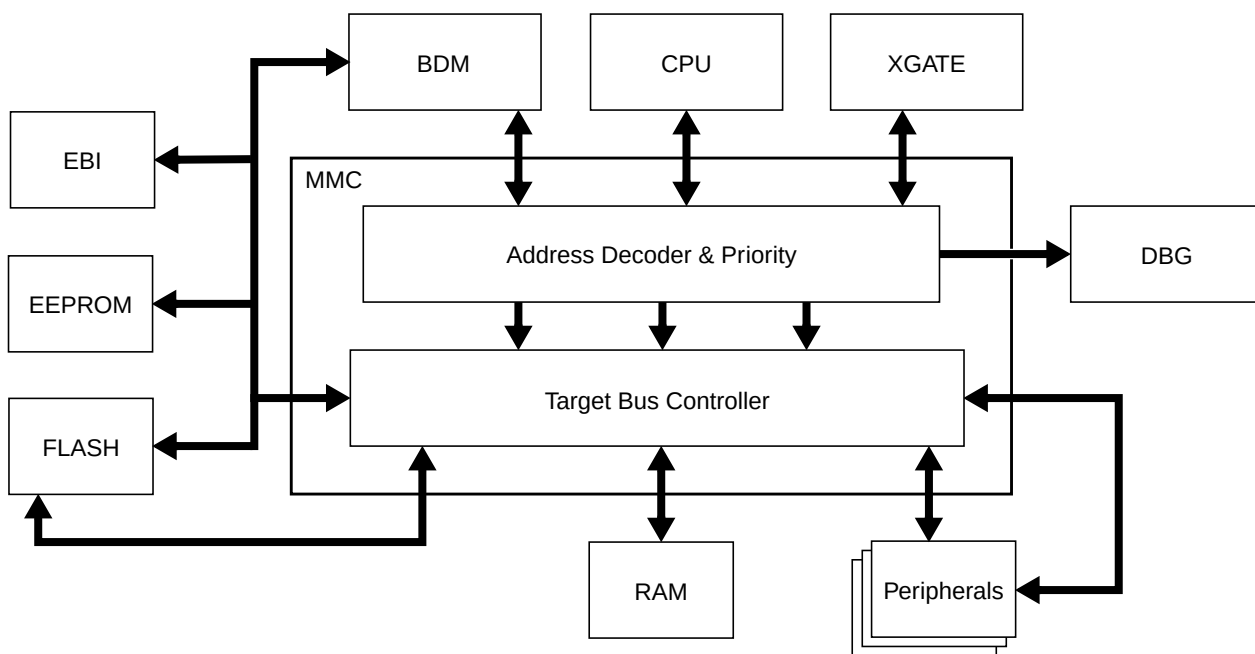


Figure 17-1. MMC Block Diagram

17.2 External Signal Description

The user is advised to refer to the SoC Guide for port configuration and location of external bus signals. Some pins may not be bonded out in all implementations.



20.3.2.4 Debug Control Register2 (DBGCR2)

Address: 0x0023

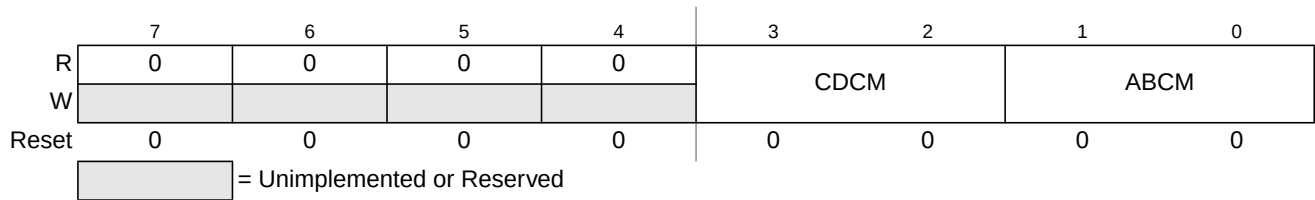


Figure 20-6. Debug Control Register2 (DBGCR2)

Read: Anytime

Write: Anytime the module is disarmed.

This register configures the comparators for range matching.

Table 20-13. DBGCR2 Field Descriptions

Field	Description
3–2 CDCM[1:0]	C and D Comparator Match Control — These bits determine the C and D comparator match mapping as described in Table 20-14 .
1–0 ABCM[1:0]	A and B Comparator Match Control — These bits determine the A and B comparator match mapping as described in Table 20-15 .

Table 20-14. CDCM Encoding

CDCM	Description
00	Match2 mapped to comparator C match..... Match3 mapped to comparator D match.
01	Match2 mapped to comparator C/D inside range..... Match3 disabled.
10	Match2 mapped to comparator C/D outside range..... Match3 disabled.
11	Reserved

Table 20-15. ABCM Encoding

ABCM	Description
00	Match0 mapped to comparator A match..... Match1 mapped to comparator B match.
01	Match 0 mapped to comparator A/B inside range..... Match1 disabled.
10	Match 0 mapped to comparator A/B outside range..... Match1 disabled.
11	Reserved

26.4.2.1 Erase Verify Command

The erase verify operation will verify that the EEPROM memory is erased.

An example flow to execute the erase verify operation is shown in [Figure 26-18](#). The erase verify command write sequence is as follows:

1. Write to an EEPROM address to start the command write sequence for the erase verify command. The address and data written will be ignored.
2. Write the erase verify command, 0x05, to the ECMD register.
3. Clear the CBEIF flag in the ESTAT register by writing a 1 to CBEIF to launch the erase verify command.

After launching the erase verify command, the CCIF flag in the ESTAT register will set after the operation has completed unless a new command write sequence has been buffered. The number of bus cycles required to execute the erase verify operation is equal to the number of words in the EEPROM memory plus 14 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. Upon completion of the erase verify operation, the BLANK flag in the ESTAT register will be set if all addresses in the EEPROM memory are verified to be erased. If any address in the EEPROM memory is not erased, the erase verify operation will terminate and the BLANK flag in the ESTAT register will remain clear.

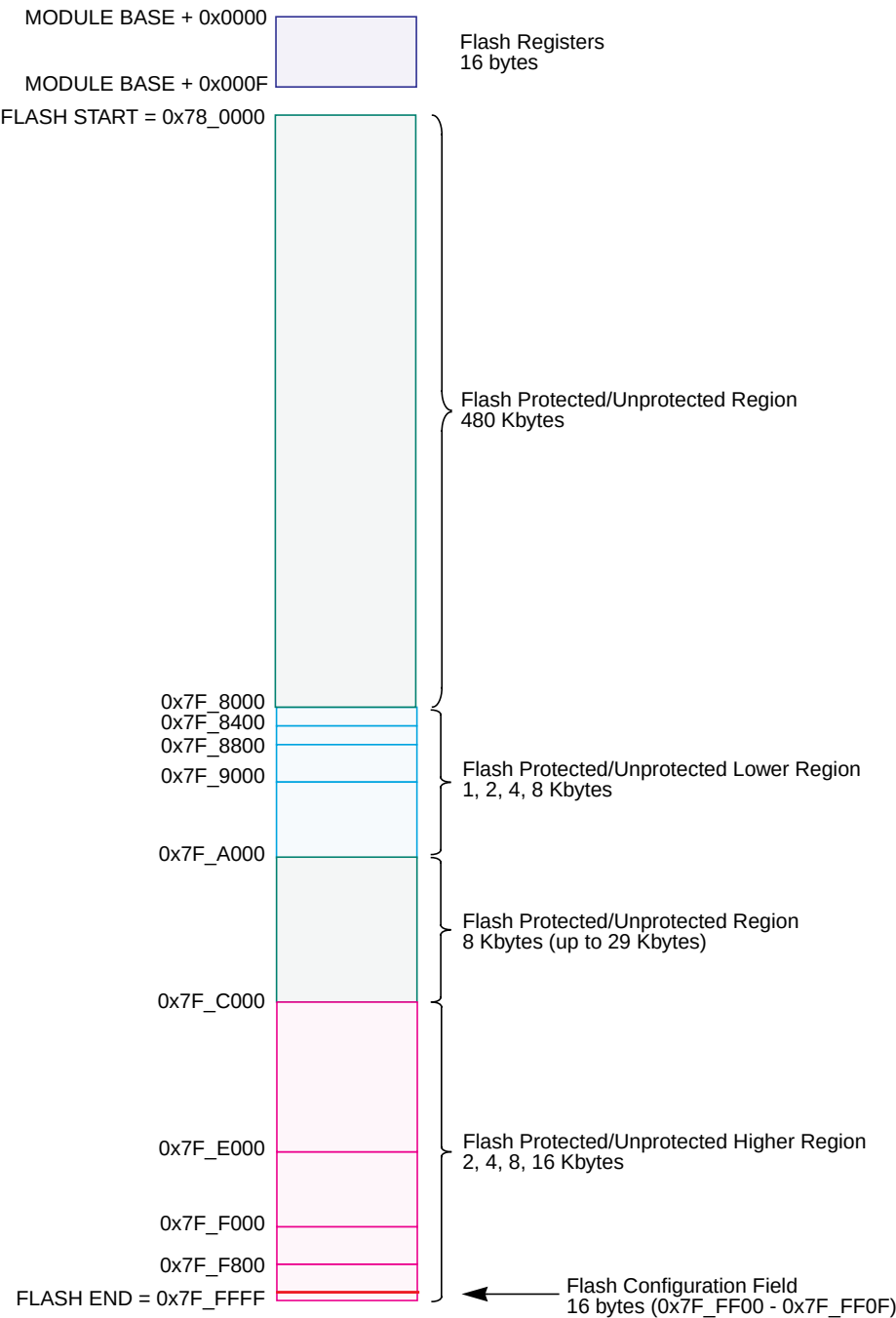


Figure 27-2. Flash Memory Map

The Flash module also contains a set of 16 control and status registers located between module base + 0x0000 and 0x000F. A summary of the Flash module registers is given in [Table 27-2](#) while their accessibility is detailed in [Section 27.3.2, “Register Descriptions”](#).

Table 27-2. Flash Register Map

Module Base +	Register Name	Normal Mode Access
0x0000	Flash Clock Divider Register (FCLKDIV)	R/W
0x0001	Flash Security Register (FSEC)	R
0x0002	Flash Test Mode Register (FTSTMOD)	R/W
0x0003	Flash Configuration Register (FCNFG)	R/W
0x0004	Flash Protection Register (FPROT)	R/W
0x0005	Flash Status Register (FSTAT)	R/W
0x0006	Flash Command Register (FCMD)	R/W
0x0007	Flash Control Register (FCTL)	R
0x0008	Flash High Address Register (FADDRHI) ¹	R
0x0009	Flash Low Address Register (FADDRLO) ¹	R
0x000A	Flash High Data Register (FDATAHI)	R
0x000B	Flash Low Data Register (FDATALO)	R
0x000C	RESERVED1 ¹	R
0x000D	RESERVED2 ¹	R
0x000E	RESERVED3 ¹	R
0x000F	RESERVED4 ¹	R

¹ Intended for factory test purposes only.

29.4.2.2 Data Compress Command

The data compress operation will check Flash code integrity by compressing data from a selected portion of the Flash memory into a signature analyzer.

An example flow to execute the data compress operation is shown in [Figure 29-24](#). The data compress command write sequence is as follows:

1. Write to a Flash block address to start the command write sequence for the data compress command. The address written determines the starting address for the data compress operation and the data written determines the number of consecutive words to compress. If the data value written is 0x0000, 64K addresses or 128 Kbytes will be compressed.
2. Write the data compress command, 0x06, to the FCMD register.
3. Clear the CBEIF flag in the FSTAT register by writing a 1 to CBEIF to launch the data compress command.

After launching the data compress command, the CCIF flag in the FSTAT register will set after the data compress operation has completed. The number of bus cycles required to execute the data compress operation is equal to two times the number of consecutive words to compress plus 18 bus cycles as measured from the time the CBEIF flag is cleared until the CCIF flag is set. Once the CCIF flag is set, the signature generated by the data compress operation is available in the FDATA registers. The signature in the FDATA registers can be compared to the expected signature to determine the integrity of the selected data stored in the selected Flash memory. If the last address of a Flash block is reached during the data compress operation, data compression will continue with the starting address of the Flash block. The MRDS bits in the FTSTMOD register will determine the sense-amp margin setting during the data compress operation.

NOTE

Since the FDATA registers (or data buffer) are written to as part of the data compress operation, a command write sequence is not allowed to be buffered behind a data compress command write sequence. The CBEIF flag will not set after launching the data compress command to indicate that a command should not be buffered behind it. If an attempt is made to start a new command write sequence with a data compress operation active, the ACCERR flag in the FSTAT register will be set. A new command write sequence should only be started after reading the signature stored in the FDATA registers.

In order to take corrective action, it is recommended that the data compress command be executed on a Flash sector or subset of a Flash sector. If the data compress operation on a Flash sector returns an invalid signature, the Flash sector should be erased using the sector erase command and then reprogrammed using the program command.

The data compress command can be used to verify that a sector or sequential set of sectors are erased.



A.5.3 Phase Locked Loop

The oscillator provides the reference clock for the PLL. The PLL's voltage controlled oscillator (VCO) is also the system clock source in self clock mode.

A.5.3.1 XFC Component Selection

This section describes the selection of the XFC components to achieve a good filter characteristics.

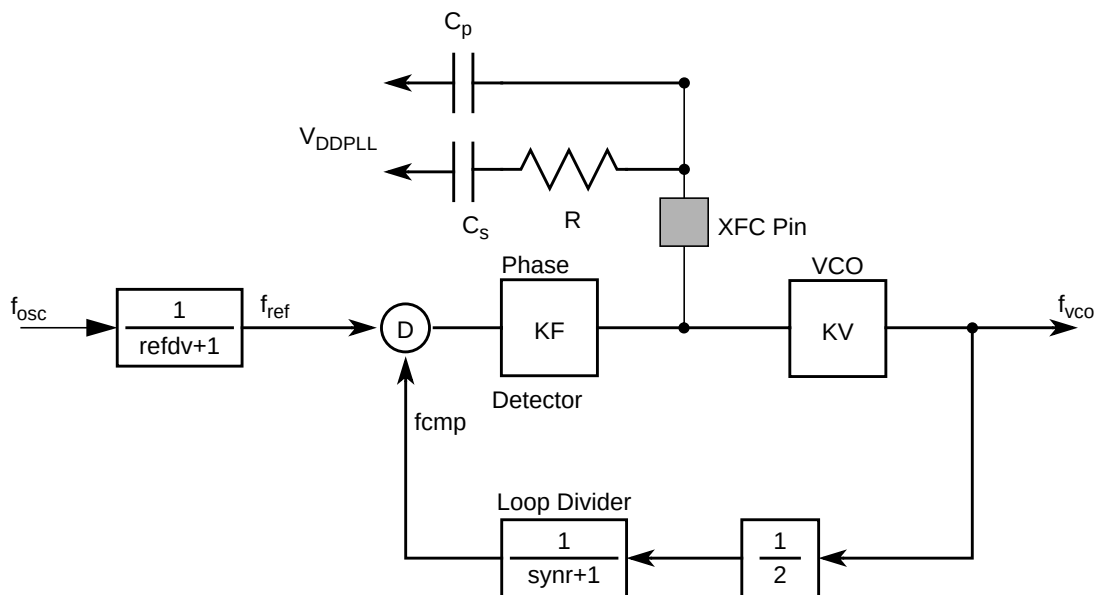


Figure A-3. Basic PLL Functional Diagram

The following procedure can be used to calculate the resistance and capacitance values using typical values for K_1 , f_1 and i_{ch} from Table A-23.

The grey boxes show the calculation for $f_{VCO} = 80$ MHz and $f_{ref} = 4$ MHz. For example, these frequencies are used for $f_{OSC} = 4$ -MHz and a 40-MHz bus clock.

The VCO gain at the desired VCO frequency is approximated by:

$$K_V = K_1 \cdot e^{\frac{(f_1 - f_{VCO})}{K_1 \cdot 1V}} = -195 \text{ MHz/V} \cdot e^{\frac{126 - 80}{-195}} = -154.0 \text{ MHz/V}$$

The phase detector relationship is given by:

$$K_\Phi = -|i_{ch}| \cdot K_V = -3.5 \mu\text{A} \cdot (-154 \text{ MHz/V}) = 539.1 \text{ Hz}/\Omega$$

i_{ch} is the current in tracking mode.

The loop bandwidth f_C should be chosen to fulfill the Gardner's stability criteria by at least a factor of 10, typical values are 50. $\zeta = 0.9$ ensures a good transient response.

0x0040–0x007F Enhanced Capture Timer 16-Bit 8-Channels (ECT) Map (Sheet 3 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x006D	TIMTST	R	0	0	0	0	0	0	0	0
		W	Reserved For Factory Test							
0x006E	PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
		W								
0x006F	PTMCPSR	R	PTMPS7	PTMPS6	PTMPS5	PTMPS4	PTMPS3	PTMPS2	PTMPS1	PTMPS0
		W								
0x0070	PBCTL	R	0	PBEN	0	0	0	0	PBOVI	0
		W								
0x0071	PBFLG	R	0	0	0	0	0	0	PBOVF	0
		W								
0x0072	PA3H	R	PA3H7	PA3H6	PA3H5	PA3H4	PA3H3	PA3H2	PA3H1	PA3H0
		W								
0x0073	PA2H	R	PA2H7	PA2H6	PA2H5	PA2H4	PA2H3	PA2H2	PA2H1	PA2H0
		W								
0x0074	PA1H	R	PA1H7	PA1H6	PA1H5	PA1H4	PA1H3	PA1H2	PA1H1	PA1H 0
		W								
0x0075	PA0H	R	PA0H7	PA0H6	PA0H5	PA0H4	PA0H3	PA0H2	PA0H1	PA0H0
		W								
0x0076	MCCNT (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0077	MCCNT (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0078	TC0H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0079	TC0H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x007A	TC1H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x007B	TC1H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x007C	TC2H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x007D	TC2H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x007E	TC3H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x007F	TC3H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								