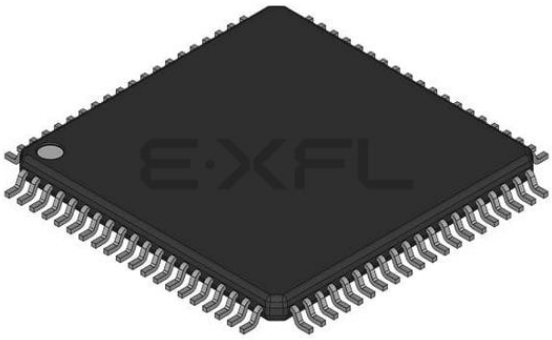


Welcome to E-XFL.COM

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	EBI/EMI, I²C, 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/pro/item?MUrl=&PartUrl=mc9s12xa256caa

Table 1-1. Device Register Memory Map (continued)

Address	Module	Size (Bytes)
0x0110–0x011B	EEPROM control register	12
0x011C–0x011F	MMC (memory map control)	4
0x0120–0x012F	INT (interrupt module)	16
0x0130–0x013F	Reserved	16
0x0130–0x0137	SCI4 (serial communications interface)	8
0x0138–0x013F	SCI5 (serial communications interface)	8
0x0140–0x017F	CAN0 (scalable CAN)	64
0x0180–0x01BF	CAN1 (scalable CAN)	64
0x0180–0x023F	Reserved	192
0x01C0–0x01FF	CAN2 (scalable CAN)	64
0x0200–0x023F	Reserved	64
0x0200–0x023F	CAN3 (scalable CAN)	64
0x0240–0x027F	PIM (port integration module)	64
0x0280–0x02BF	CAN4 (scalable CAN)	64
0x02C0–0x02DF	Reserved	32
0x02C0–0x02DF	ATD0 (analog-to-digital converter 10 bit 8-channel)	32
0x02E0–0x02EF	Unimplemented	16
0x02F0–0x02F7	Voltage regulator	8
0x02F8–0x02FF	Unimplemented	8
0x0300–0x0327	PWM (pulse-width modulator 8 channels)	40
0x0328–0x033F	Unimplemented	24
0x0340–0x0367	Periodic interrupt timer	40
0x0368–0x037F	Unimplemented	24
0x0380–0x03BF	XGATE	64
0x03C0–0x03FF	Unimplemented	64
0x0400–0x07FF	Unimplemented	1024

1.2.3.40 PK7 / EWAIT / ROMCTL — Port K I/O Pin 7

PK7 is a general-purpose input or output pin. During MCU emulation modes and normal expanded modes of operation, this pin is used to enable the Flash EEPROM memory in the memory map (ROMCTL). At the rising edge of RESET, the state of this pin is latched to the ROMON bit. The EWAIT input signal maintains the external bus access until the external device is ready to capture data (write) or provide data (read).

The input voltage threshold for PK7 can be configured to reduced levels, to allow data from an external 3.3-V peripheral to be read by the MCU operating at 5.0 V. The input voltage threshold for PK7 is configured to reduced levels out of reset in expanded and emulation modes.

1.2.3.41 PK[6:4] / ADDR[22:20] / ACC[2:0] — Port K I/O Pin [6:4]

PK[6:4] are general-purpose input or output pins. During MCU expanded modes of operation, the ACC[2:0] signals are used to indicate the access source of the bus cycle. This pins also provide the expanded addresses ADDR[22:20] for the external bus. In Emulation modes ACC[2:0] is available and is time multiplexed with the high addresses

1.2.3.42 PK[3:0] / ADDR[19:16] / IQSTAT[3:0] — Port K I/O Pins [3:0]

PK3-PK0 are general-purpose input or output pins. In MCU expanded modes of operation, these pins provide the expanded address ADDR[19:16] for the external bus and carry instruction pipe information.

1.2.3.43 PK7,PK[5:0] — Port K I/O Pins 7 & [5:0]

PK7 and PK[5:0] are general-purpose input or output pins.

1.2.3.44 PM7 / TXCAN3 / TXCAN4 / TXD3 — Port M I/O Pin 7

PM7 is a general-purpose input or output pin. It can be configured as the transmit pin TXCAN of the scalable controller area network controller 3 or 4 (CAN3 or CAN4). PM7 can be configured as the transmit pin TXD3 of the serial communication interface 3 (SCI3).

1.2.3.45 PM6 / RXCAN3 / RXCAN4 / RXD3 — Port M I/O Pin 6

PM6 is a general-purpose input or output pin. It can be configured as the receive pin RXCAN of the scalable controller area network controller 3 or 4 (CAN3 or CAN4). PM6 can be configured as the receive pin RXD3 of the serial communication interface 3 (SCI3).

1.2.3.46 PM5 / TXCAN0 / TXCAN2 / TXCAN4 / SCK0 — Port M I/O Pin 5

PM5 is a general-purpose input or output pin. It can be configured as the transmit pin TXCAN of the scalable controller area network controllers 0, 2 or 4 (CAN0, CAN2, or CAN4). It can be configured as the serial clock pin SCK of the serial peripheral interface 0 (SPI0).

Table 1-9. Chip Modes and Data Sources

Chip Modes	BKGD = MODC	PE6 = MODB	PE5 = MODA	PK7 = ROMCTL	PE3 = EROMCTL	Data Source[1]
Normal single chip	1	0	0	X	X	Internal
Special single chip	0	0	0			
Emulation single chip	0	0	1	X	0	Emulation memory
				X	1	Internal Flash
Normal expanded	1	0	1	0	X	External application
				1	X	Internal Flash
Emulation expanded	0	1	1	0	X	External application
				1	0	Emulation memory
				1	1	Internal Flash
Special test	0	1	0	0	X	External application
				1	X	Internal Flash

1 Internal means resources inside the MCU are read/written.
Internal Flash means Flash resources inside the MCU are read/written.
Emulation memory means resources inside the emulator are read/written (PRU registers, Flash replacement, RAM, EEPROM, and register space are always considered internal).
External application means resources residing outside the MCU are read/written.

The configuration of the oscillator can be selected using the $\overline{\text{XCLKS}}$ signal (see Table 1-10). For a detailed description please refer to the S12CRG section.

Table 1-10. Clock Selection Based on PE7

PE7 = $\overline{\text{XCLKS}}$	Description
0	Full swing Pierce oscillator or external clock source selected
1	Loop controlled Pierce oscillator selected

The logic level on the voltage regulator enable pin V_{REGEN} determines whether the on-chip voltage regulator is enabled or disabled (see Table 1-11).

Table 1-11. Voltage Regulator VREGEN

V_{REGEN}	Description
1	Internal voltage regulator enabled
0	Internal voltage regulator disabled, $V_{DD1,2}$ and V_{DDPLL} must be supplied externally

7.4.1.2 OC Channel Initialization

Internal register whose output drives OCx when TIOS is set, can be force loaded with a desired data by writing to CFORC register before OCx is configured for output compare action. This allows a glitch free switch over of port from general purpose I/O to timer output once the output compare is enabled.

7.4.1.3 Pulse Accumulators

There are four 8-bit pulse accumulators with four 8-bit holding registers associated with the four IC buffered channels 3–0. A pulse accumulator counts the number of active edges at the input of its channel.

The minimum pulse width for the PAI input is greater than two bus clocks.The maximum input frequency on the pulse accumulator channel is one half the bus frequency or Eclk.

The user can prevent the 8-bit pulse accumulators from counting further than 0x00FF by utilizing the PACMX control bit in the ICSYS register. In this case, a value of 0x00FF means that 255 counts or more have occurred.

Each pair of pulse accumulators can be used as a 16-bit pulse accumulator (see Figure 7-70).

Pulse accumulator B operates only as an event counter, it does not feature gated time accumulation mode.The edge control for pulse accumulator B as a 16-bit pulse accumulator is defined by TCTL4[1:0].

To operate the 16-bit pulse accumulators A and B (PACA and PACB) independently of input capture or output compare 7 and 0 respectively, the user must set the corresponding bits: IOSx = 1, OMx = 0, and OLx = 0. OC7M7 or OC7M0 in the OC7M register must also be cleared.

There are two modes of operation for the pulse accumulators:

- Pulse accumulator latch mode

  The value of the pulse accumulator is transferred to its holding register when the modulus down-counter reaches zero, a write 0x0000 to the modulus counter or when the force latch control bit ICLAT is written.

  At the same time the pulse accumulator is cleared.
- Pulse accumulator queue mode

  When queue mode is enabled, reads of an input capture holding register will transfer the contents of the associated pulse accumulator to its holding register.

  At the same time the pulse accumulator is cleared.

7.4.1.4 Modulus Down-Counter

The modulus down-counter can be used as a time base to generate a periodic interrupt. It can also be used to latch the values of the IC registers and the pulse accumulators to their holding registers.

The action of latching can be programmed to be periodic or only once.

As an example of a center aligned output, consider the following case:

Clock Source = E, where E = 10 MHz (100 ns period)

PPOLx = 0

PWMPERx = 4

PWMDTYx = 1

PWMx Frequency = 10 MHz/8 = 1.25 MHz

PWMx Period = 800 ns

PWMx Duty Cycle = 3/4 *100% = 75%

Shown in Figure 8-23 is the output waveform generated.

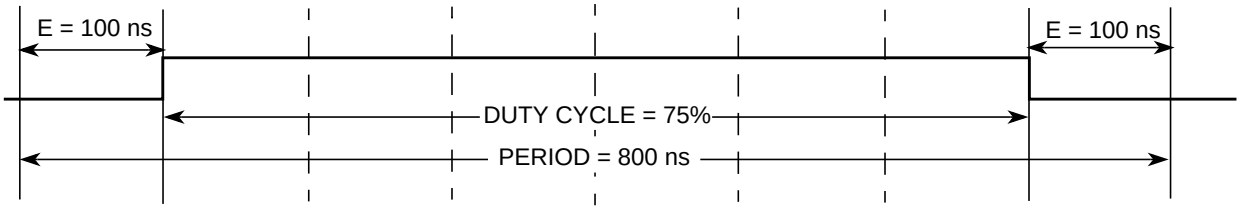


Figure 8-23. PWM Center Aligned Output Example Waveform

8.4.2.7 PWM 16-Bit Functions

The PWM timer also has the option of generating 8-channels of 8-bits or 4-channels of 16-bits for greater PWM resolution. This 16-bit channel option is achieved through the concatenation of two 8-bit channels.

The PWMCTL register contains four control bits, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. Channels 6 and 7 are concatenated with the CON67 bit, channels 4 and 5 are concatenated with the CON45 bit, channels 2 and 3 are concatenated with the CON23 bit, and channels 0 and 1 are concatenated with the CON01 bit.

NOTE

Change these bits only when both corresponding channels are disabled.

When channels 6 and 7 are concatenated, channel 6 registers become the high order bytes of the double byte channel, as shown in Figure 8-24. Similarly, when channels 4 and 5 are concatenated, channel 4 registers become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double byte channel.

When using the 16-bit concatenated mode, the clock source is determined by the low order 8-bit channel clock select control bits. That is channel 7 when channels 6 and 7 are concatenated, channel 5 when channels 4 and 5 are concatenated, channel 3 when channels 2 and 3 are concatenated, and channel 1 when channels 0 and 1 are concatenated. The resulting PWM is output to the pins of the corresponding low order 8-bit channel as also shown in Figure 8-24. The polarity of the resulting PWM output is controlled by the PPOLx bit of the corresponding low order 8-bit channel as well.

9.4 Functional Description

This section provides a complete functional description of the IICV2.

9.4.1 I-Bus Protocol

The IIC bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. All devices connected to it must have open drain or open collector outputs. Logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors is system dependent.

Normally, a standard communication is composed of four parts: START signal, slave address transmission, data transfer and STOP signal. They are described briefly in the following sections and illustrated in Figure 9-9.

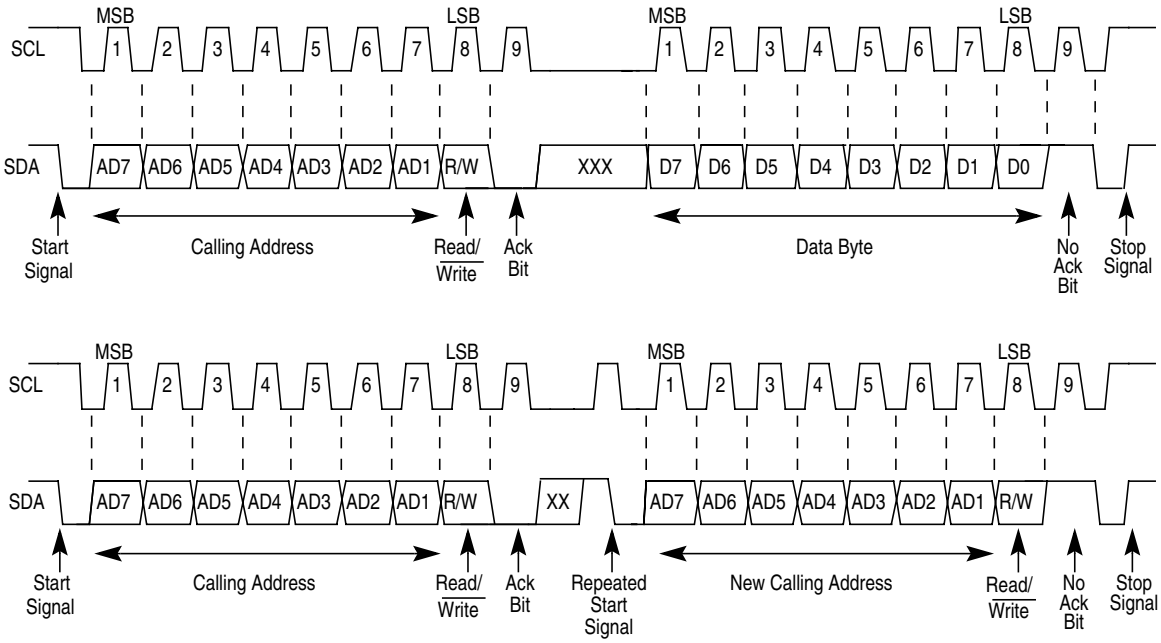


Figure 9-9. IIC-Bus Transmission Signals

9.4.1.1 START Signal

When the bus is free, i.e. no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a START signal.As shown in Figure 9-9, a START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and brings all slaves out of their idle states.

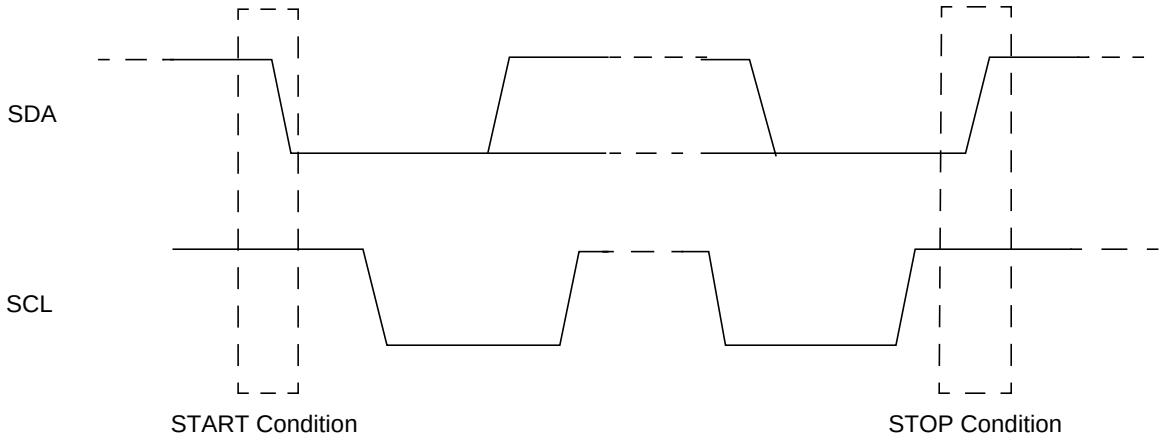


Figure 9-10. Start and Stop Conditions

9.4.1.2 Slave Address Transmission

The first byte of data transfer immediately after the START signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a R/W bit. The R/W bit tells the slave the desired direction of data transfer.

1 = Read transfer, the slave transmits data to the master.
0 = Write transfer, the master transmits data to the slave.

Only the slave with a calling address that matches the one transmitted by the master will respond by sending back an acknowledge bit. This is done by pulling the SDA low at the 9th clock (see Figure 9-9).

No two slaves in the system may have the same address. If the IIC bus is master, it must not transmit an address that is equal to its own slave address. The IIC bus cannot be master and slave at the same time.However, if arbitration is lost during an address cycle the IIC bus will revert to slave mode and operate correctly even if it is being addressed by another master.

9.4.1.3 Data Transfer

As soon as successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/W bit sent by the calling master

All transfers that come after an address cycle are referred to as data transfers, even if they carry sub-address information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low and must be held stable while SCL is high as shown in Figure 9-9. There is one clock pulse on SCL for each data bit, the MSB being transferred first. Each data byte has to be followed by an acknowledge bit, which is signalled from the receiving device by pulling the SDA low at the ninth clock. So one complete data byte transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master, the SDA line must be left high by the slave. The master can then generate a stop signal to abort the data transfer or a start signal (repeated start) to commence a new calling.

10.4.4.3 Emulation Modes

In all emulation modes, the MSCAN module behaves just like normal system operation modes as described within this specification.

10.4.4.4 Listen-Only Mode

In an optional CAN bus monitoring mode (listen-only), the CAN node is able to receive valid data frames and valid remote frames, but it sends only "recessive" bits on the CAN bus. In addition, it cannot start a transmision. If the MAC sub-layer is required to send a "dominant" bit (ACK bit, overload flag, or active error flag), the bit is rerouted internally so that the MAC sub-layer monitors this "dominant" bit, although the CAN bus may remain in recessive state externally.

10.4.4.5 Security Modes

The MSCAN module has no security features.

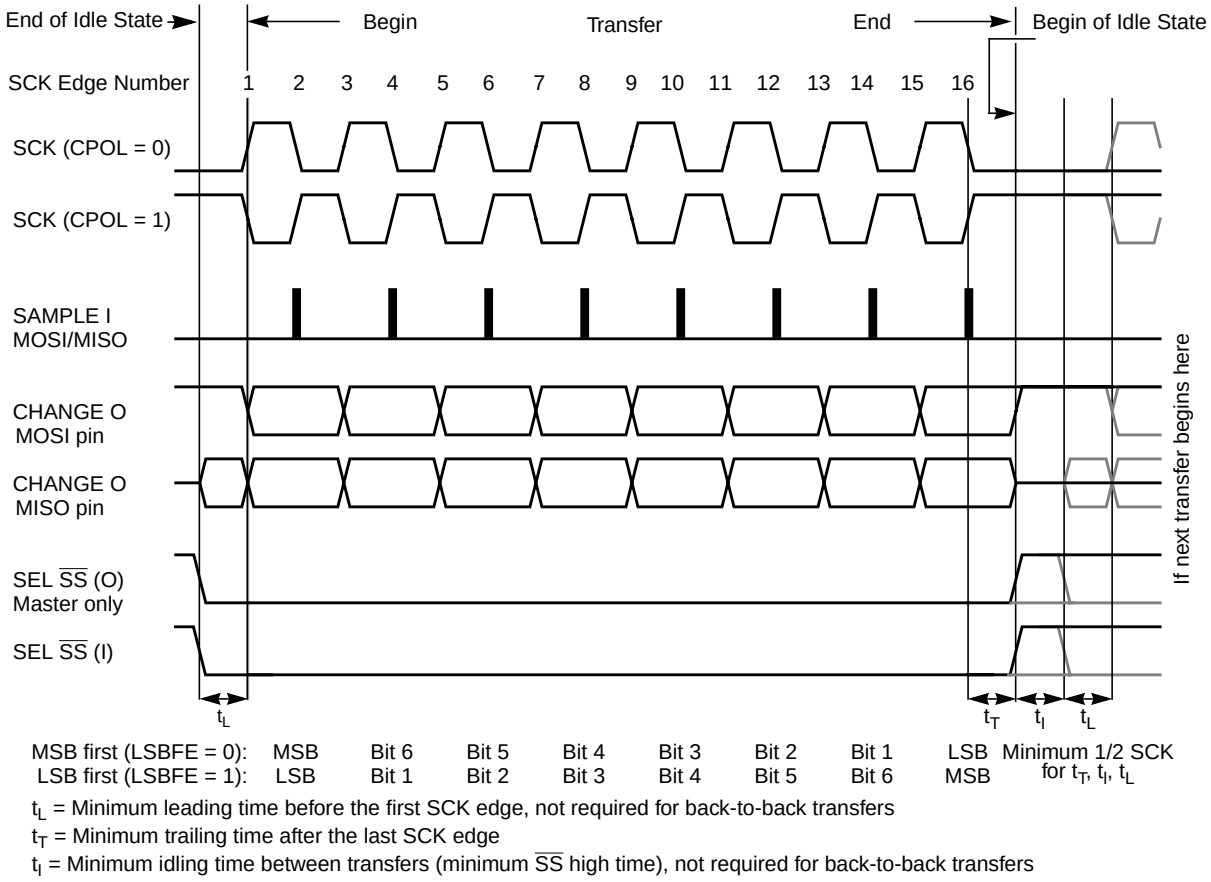
10.4.5 Low-Power Options

If the MSCAN is disabled (CANE = 0), the MSCAN clocks are stopped for power saving.

If the MSCAN is enabled (CANE = 1), the MSCAN has two additional modes with reduced power consumption, compared to normal mode: sleep and power down mode. In sleep mode, power consumption is reduced by stopping all clocks except those to access the registers from the CPU side. In power down mode, all clocks are stopped and no power is consumed.

Table 10-36 summarizes the combinations of MSCAN and CPU modes. A particular combination of modes is entered by the given settings on the CSWAI and SLPRQ/SLPAK bits.

For all modes, an MSCAN wake-up interrupt can occur only if the MSCAN is in sleep mode (SLPRQ = 1 and SLPAK = 1), wake-up functionality is enabled (WUPE = 1), and the wake-up interrupt is enabled (WUPIE = 1).



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 12-12. SPI Clock Format 1 (CPHA = 1)

The $\overline{SS}$ line can remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave that drive the MISO data line.

- Back-to-back transfers in master mode

  In master mode, if a transmission has completed and a new data byte is available in the SPI data register, this byte is sent out immediately without a trailing and minimum idle time.

The SPI interrupt request flag (SPIF) is common to both the master and slave modes. SPIF gets set one half SCK cycle after the last SCK edge.

14.1.3 Block Diagram

Figure 14-1 shows the function principle of VREG_3V3 by means of a block diagram. The regulator core REG consists of two parallel subblocks, REG1 and REG2, providing two independent output voltages.

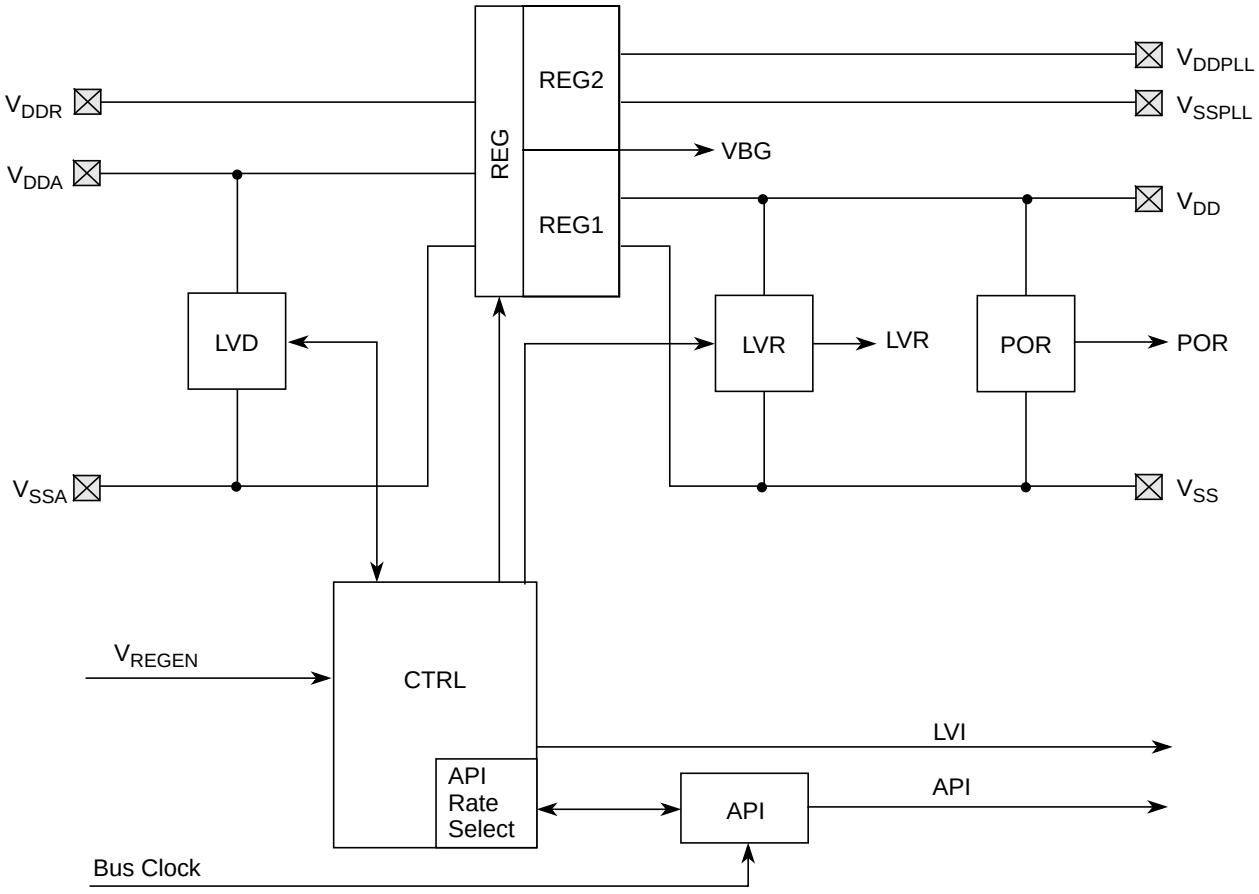


Figure 14-1. VREG_3V3 Block Diagram

Chapter 17
Memory Mapping Control (S12XMMCV2)

17.1 Introduction

This section describes the functionality of the module mapping control (MMC) sub-block of the S12X platform. The block diagram of the MMC is shown in Figure 1-1.

The MMC module controls the multi-master priority accesses, the selection of internal resources and external space. Internal buses including internal memories and peripherals are controlled in this module. The local address space for each master is translated to a global memory space.

17.1.1 Features

The main features of this block are:

- Paging capability to support a global 8 Mbytes memory address space
- Bus arbitration between the masters CPU, BDM, and XGATE
- Simultaneous accesses to different resources[1] (internal, external, and peripherals) (see Figure 1-1)
- Resolution of target bus access collision
- Access restriction control from masters to some targets (e.g., RAM write access protection for user specified areas)
- MCU operation mode control
- MCU security control
- Separate memory map schemes for each master CPU, BDM, and XGATE
- ROM control bits to enable the on-chip FLASH or ROM selection
- Port replacement registers access control
- Generation of system reset when CPU accesses an unimplemented address (i.e., an address which does not belong to any of the on-chip modules) in single-chip modes

17.1.2 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the MMC.

17.1.2.1 Power Saving Modes

- Run mode

  MMC is functional during normal run mode.

1. Resources are also called targets.

17.3.2.4 Direct Page Register (DIRECT)

Address: 0x0011

	7	6	5	4	3	2	1	0
R W	DP15	DP14	DP13	DP12	DP11	DP10	DP9	DP8
Reset	0	0	0	0	0	0	0	0

Figure 17-8. Direct Register (DIRECT)

Read: Anytime

Write: anytime in special modes, one time only in other modes.

This register determines the position of the direct page within the memory map.

Table 17-8. DIRECT Field Descriptions

Field	Description
7–0 DP[15:8]	**Direct Page Index Bits 15–8** — These bits are used by the CPU when performing accesses using the direct addressing mode. The bits from this register form bits [15:8] of the address (see Figure 1-9).

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

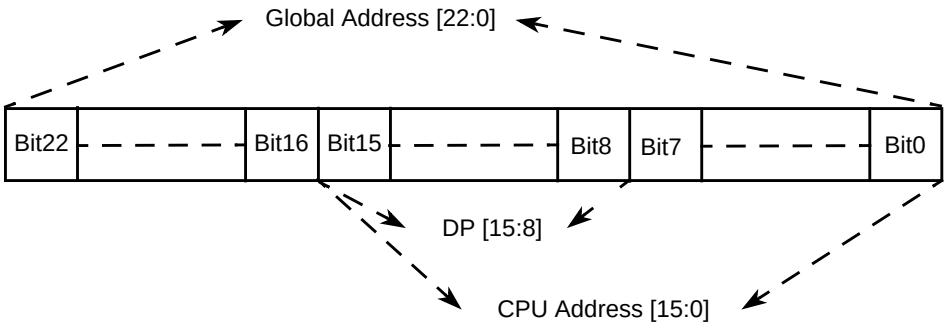


Figure 17-9. DIRECT Address Mapping

Bits [22:16] of the global address will be formed by the GPAGE[6:0] bits in case the CPU executes a global instruction in direct addressing mode or by the appropriate local address to the global address expansion (refer to Expansion of the CPU Local Address Map).

Example 17-2. This example demonstrates usage of the Direct Addressing Mode by a global instruction

```
LDAADR   EQU $0000          ;Initialize LDADDR with the value of $0000
MOVB     #$80,DIRECT        ;Initialize DIRECT register with the value of $80
MOVB     #$14,GPAGE         ;Initialize GPAGE register with the value of $14
GLDAA    <LDAADR            ;Load Accu A from the global address $14_8000
```

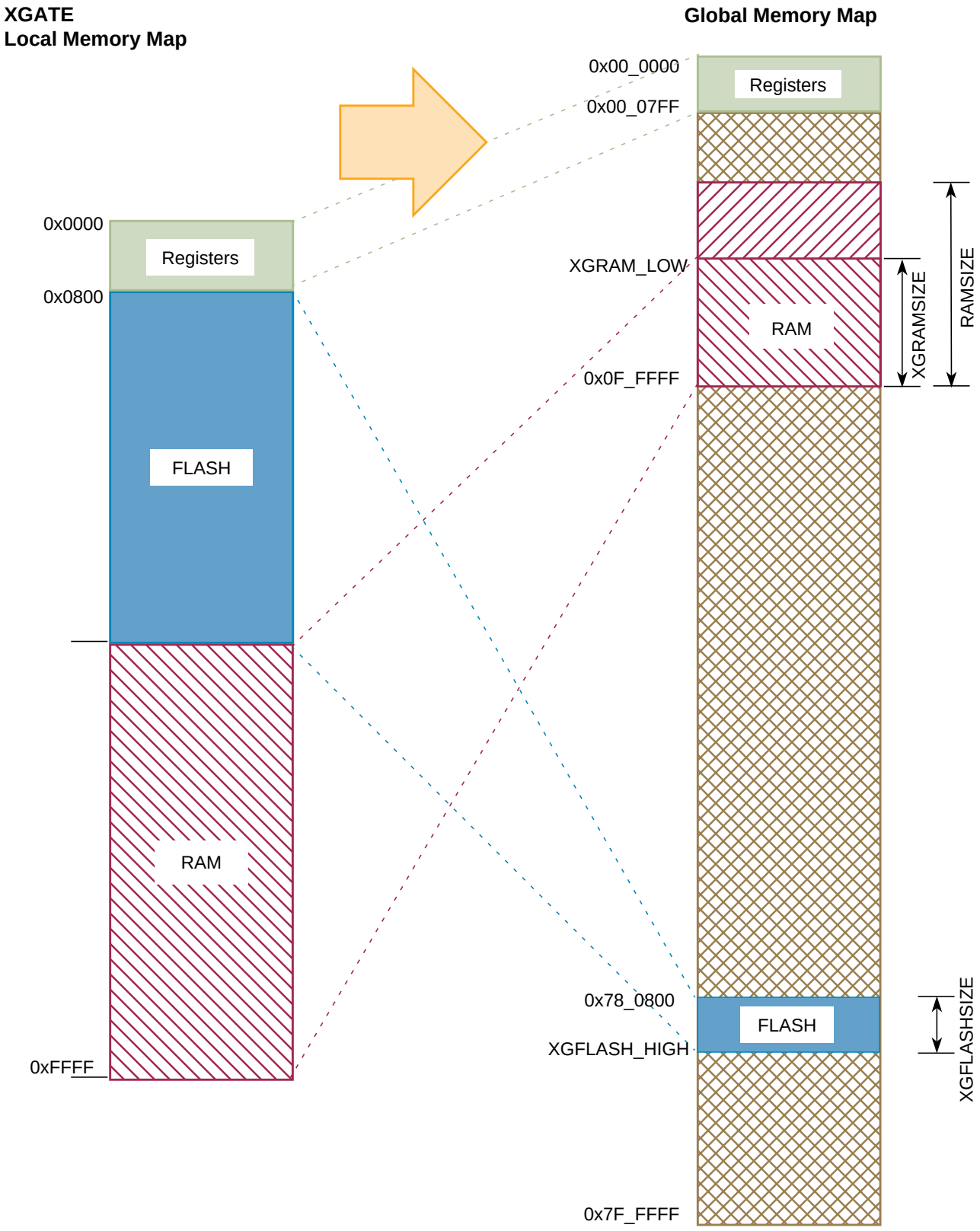


Figure 18-24. XGATE Global Address Mapping

19.3.1.3 Debug Trace Control Register (DBGTCR)

0x0022

	7	6	5	4	3	2	1	0
R W	TSOURCE		TRANGE		TRCMOD		TALIGN	
Reset	0	0	0	0	0	0	0	0

Figure 19-5. Debug Trace Control Register (DBGTCR)

Read: Anytime

Write: Bits 7:6 only when DBG is neither secure nor armed.
Bits 5:0 anytime the module is disarmed.

Table 19-8. DBGTCR Field Descriptions

Field	Description
7–6 TSOURCE	**Trace Source Control Bits** — The TSOURCE bits select the data source for the tracing session. If the MCU system is secured, these bits cannot be set and tracing is inhibited. See Table 19-9.
5–4 TRANGE[5:4]	**Trace Range Bits** —The TRANGE bits allow filtering of trace information from a selected address range when tracing from the CPU in detail mode. The XGATE tracing range cannot be narrowed using these bits. To use a comparator for range filtering, the corresponding COMPE and SRC bits must remain cleared. If the COMPE bit is not clear then the comparator will also be used to generate state sequence triggers or tags. If the SRC bit is set the comparator is mapped to the XGATE busses, corrupting the trace. See Table 19-10.
3–2 TRCMOD[3:2]	**Trace Mode Bits** — See Section 19.4.5.2, “Trace Modes“ for detailed trace mode descriptions. In normal mode, change of flow information is stored. In loop1 mode, change of flow information is stored but redundant entries into trace memory are inhibited. In detail mode, address and data for all memory and register accesses is stored. See Table 19-11
1–0 TALIGN[1:0]	**Trigger Align Bits** — These bits control whether the trigger is aligned to the beginning, end or the middle of a tracing session. See Table 19-12.

Table 19-9. TSOURCE Trace Source Bit Encoding

TSOURCE	Tracing Source
00	No tracing requested
01	CPU
10[1]	XGATE
11[1, 2]	Both CPU and XGATE

1 No range limitations are allowed. Thus tracing operates as if TRANGE = 00.

2 No detail mode tracing supported. If TRCMOD =10, no information is stored.

Table 19-10. TRANGE Trace Range Encoding

TRANGE	Tracing Source
00	Trace from all addresses (No filter)
01	Trace only in address range from 0x0000 to comparator D
10	Trace only in address range from comparator C to 0x7FFFFF

20.3.2.8.8 Debug Comparator Data Low Mask Register (DBGXDLM)

Address: 0x002F

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

Figure 20-21. Debug Comparator Data Low Mask Register (DBGXDLM)

Read: Anytime

Write: Anytime when S12XDBG not armed.

Table 20-35. DBGXDLM Field Descriptions

Field	Description
7–0 Bits[7:0]	**Comparator Data Low Mask Bits** — The Comparator data low mask bits control whether the selected comparator compares the data bus bits [7:0] to the corresponding comparator data compare bits. This register is available only for comparators A and C. 0 Do not compare corresponding data bit 1 Compare corresponding data bit

20.4 Functional Description

This section provides a complete functional description of the S12XDBG module. If the part is in secure mode, the S12XDBG module can generate breakpoints but tracing is not possible.

20.4.1 S12XDBG Operation

Arming the S12XDBG module by setting ARM in DBGC1 allows triggering, and storing of data in the trace buffer and can be used to cause breakpoints to the S12XCPU or the XGATE module. The DBG module is made up of four main blocks, the comparators, control logic, the state sequencer, and the trace buffer.

The comparators monitor the bus activity of the S12XCPU and XGATE modules. Comparators can be configured to monitor address and databus. Comparators can also be configured to mask out individual data bus bits during a compare and to use R/W and word/byte access qualification in the comparison. When a match with a comparator register value occurs the associated control logic can trigger the state sequencer to another state (see Figure 20-23). Either forced or tagged triggers are possible. Using a forced trigger, the trigger is generated immediately on a comparator match. Using a tagged trigger, at a comparator match, the instruction opcode is tagged and only if the instruction reaches the execution stage of the instruction queue is a trigger generated. In the case of a transition to Final State, bus tracing is triggered and/or a breakpoint can be generated. Tracing of both S12XCPU and/or XGATE bus activity is possible.

Independent of the state sequencer, a breakpoint can be triggered by the external $\overline{\text{TAGHI}}$ / $\overline{\text{TAGLO}}$ signals, by an XGATE S/W breakpoint request or by writing to the TRIG bit in the DBGC1 control register.

21.5.1 Normal Expanded Mode

This mode allows interfacing to external memories or peripherals which are available in the commercial market. In these applications the normal bus operation requires a minimum of 1 cycle stretch for each external access.

21.5.1.1 Example 1a: External Wait Feature Disabled

The first example of bus timing of an external read and write access with the external wait feature disabled is shown in

- Figure ‘Example 1a: Normal Expanded Mode — Read Followed by Write’

The associated supply voltage dependent timing are numbers given in

- Table ‘Example 1a: Normal Expanded Mode Timing V_{DD5} = 5.0 V (EWAITE = 0)’
- Table ‘Example 1a: Normal Expanded Mode Timing V_{DD5} = 3.0 V (EWAITE = 0)’

Systems designed this way rely on the internal programmable access stretching. These systems have predictable external memory access times. The additional stretch time can be programmed up to 8 cycles to provide longer access times.

21.5.1.2 Example 1b: External Wait Feature Enabled

The external wait operation is shown in this example. It can be used to exceed the amount of stretch cycles over the programmed number in EXSTR[2:0]. The feature must be enabled by writing EWAITE = 1.

If the $\overline{\text{EWAIT}}$ signal is not asserted, the number of stretch cycles is forced to a minimum of 2 cycles. If $\overline{\text{EWAIT}}$ is asserted within the predefined time window during the access it will be strobed active and another stretch cycle is added. If strobed inactive, the next cycle will be the last cycle before the access is finished. $\overline{\text{EWAIT}}$ can be held asserted as long as desired to stretch the access.

An access with 1 cycle stretch by $\overline{\text{EWAIT}}$ assertion is shown in

- Figure ‘Example 1b: Normal Expanded Mode — Stretched Read Access’
- Figure ‘Example 1b: Normal Expanded Mode — Stretched Write Access’

The associated timing numbers for both operations are given in

- Table ‘Example 1b: Normal Expanded Mode Timing V_{DD5} = 5.0 V (EWAITE = 1)’
- Table ‘Example 1b: Normal Expanded Mode Timing V_{DD5} = 3.0 V (EWAITE = 1)’

It is recommended to use the free-running clock (ECLK) at the fastest rate (bus clock rate) to synchronize the $\overline{\text{EWAIT}}$ input signal.

Table 24-38. PPSP Field Descriptions

Field	Description
7–0 PPSP[7:0]	**Polarity Select Port P** 0 Falling edge on the associated port P pin sets the associated flag bit in the PIFP register.A pull-up device is connected to the associated port P pin, if enabled by the associated bit in register PERP and if the port is used as input. 1 Rising edge on the associated port P pin sets the associated flag bit in the PIFP register.A pull-down device is connected to the associated port P pin, if enabled by the associated bit in register PERP and if the port is used as input.

24.0.5.40 Port P Interrupt Enable Register (PIEP)

	7	6	5	4	3	2	1	0
R W	PIEP7	PIEP6	PIEP5	PIEP4	PIEP3	PIEP2	PIEP1	PIEP0
Reset	0	0	0	0	0	0	0	0

Figure 24-42. Port P Interrupt Enable Register (PIEP)

Read: Anytime.

Write: Anytime.

This register disables or enables on a per-pin basis the edge sensitive external interrupt associated with Port P.

Table 24-39. PIEP Field Descriptions

Field	Description
7–0 PIEP[7:0]	**Interrupt Enable Port P** 0 Interrupt is disabled (interrupt flag masked). 1 Interrupt is enabled.

24.0.5.41 Port P Interrupt Flag Register (PIFP)

	7	6	5	4	3	2	1	0
R W	PIFP7	PIFP6	PIFP5	PIFP4	PIFP3	PIFP2	PIFP1	PIFP0
Reset	0	0	0	0	0	0	0	0

Figure 24-43. Port P Interrupt Flag Register (PIFP)

Read: Anytime.

Write: Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSP register. To clear this flag, write logic level “1” to the corresponding bit in the PIFP register. Writing a “0” has no effect.

27.4.1.1 Writing the FCLKDIV Register

Prior to issuing any Flash command after a reset, the user is required to write the FCLKDIV register to divide the oscillator clock down to within the 150 kHz to 200 kHz range. Since the program and erase timings are also a function of the bus clock, the FCLKDIV determination must take this information into account.

If we define:

- FCLK as the clock of the Flash timing control block
- Tbus as the period of the bus clock
- INT(x) as taking the integer part of x (e.g. INT(4.323) = 4)

then FCLKDIV register bits PRDIV8 and FDIV[5:0] are to be set as described in Figure 27-24.

For example, if the oscillator clock frequency is 950kHz and the bus clock frequency is 10MHz, FCLKDIV bits FDIV[5:0] should be set to 0x04 (000100) and bit PRDIV8 set to 0. The resulting FCLK frequency is then 190kHz. As a result, the Flash program and erase algorithm timings are increased over the optimum target by:

$$(200 - 190)/200 \times 100 = 5\%$$

If the oscillator clock frequency is 16MHz and the bus clock frequency is 40MHz, FCLKDIV bits FDIV[5:0] should be set to 0x0A (001010) and bit PRDIV8 set to 1. The resulting FCLK frequency is then 182kHz. In this case, the Flash program and erase algorithm timings are increased over the optimum target by:

$$(200 - 182)/200 \times 100 = 9\%$$

CAUTION

Program and erase command execution time will increase proportionally with the period of FCLK. Because of the impact of clock synchronization on the accuracy of the functional timings, programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Programming or erasing the Flash memory with FCLK < 150 kHz should be avoided. Setting FCLKDIV to a value such that FCLK < 150 kHz can destroy the Flash memory due to overstress. Setting FCLKDIV to a value such that (1/FCLK+Tbus) < 5µs can result in incomplete programming or erasure of the Flash memory cells.

If the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written to, the Flash command loaded during a command write sequence will not execute and the ACCERR flag in the FSTAT register will set.

28.4.2.1 Erase Verify Command

The erase verify operation will verify that a Flash block is erased.

An example flow to execute the erase verify operation is shown in Figure 28-25. The erase verify command write sequence is as follows:

1. Write to a Flash block address to start the command write sequence for the erase verify command. The address and data written will be ignored. Multiple Flash blocks can be simultaneously erase verified by writing to the same relative address in each Flash block.
2. Write the erase verify command, 0x05, to the FCMD register.
3. Clear the CBEIF flag in the FSTAT register by writing a 1 to CBEIF to launch the erase verify command.

After launching the erase verify command, the CCIF flag in the FSTAT 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 addresses in a Flash block 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 FSTAT register will be set if all addresses in the selected Flash blocks are verified to be erased. If any address in a selected Flash block is not erased, the erase verify operation will terminate and the BLANK flag in the FSTAT register will remain clear. The MRDS bits in the FTSTMOD register will determine the sense-amp margin setting during the erase verify operation.

[2] This represents the contents if the Comparator B or D control register is blended into this address