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	Not For New Designs
Core Processor	HCS12X
Core Size	16-Bit
Speed	80MHz
Connectivity	CANbus, I²C, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	91
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	2.35V ~ 5.5V
Data Converters	A/D 8x10b, 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	112-LQFP
Supplier Device Package	112-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/s912xdp512f0mal

MC9S12XDP512RMV2 Data Sheet

MC9S12XDP512RMV2
Rev. 2.21
October 2009

freescale™
semiconductor

6.4.4 Semaphores . . . 204
6.4.5 Software Error Detection . . . 205
6.5 Interrupts . . . 206
6.5.1 Incoming Interrupt Requests . . . 206
6.5.2 Outgoing Interrupt Requests . . . 206
6.6 Debug Mode . . . 206
6.6.1 Debug Features . . . 206
6.6.2 Entering Debug Mode . . . 207
6.6.3 Leaving Debug Mode . . . 208
6.7 Security . . . 208
6.8 Instruction Set . . . 208
6.8.1 Addressing Modes . . . 208
6.8.2 Instruction Summary and Usage . . . 212
6.8.3 Cycle Notation . . . 215
6.8.4 Thread Execution . . . 215
6.8.5 Instruction Glossary . . . 215
6.8.6 Instruction Coding . . . 288
6.9 Initialization and Application Information . . . 291
6.9.1 Initialization . . . 291
6.9.2 Code Example (Transmit "Hello World!" on SCI) . . . 291

Chapter 7
Enhanced Capture Timer (S12ECT16B8CV2)

7.1 Introduction . . . 309
7.1.1 Features . . . 309
7.1.2 Modes of Operation . . . 309
7.1.3 Block Diagram . . . 310
7.2 External Signal Description . . . 311
7.2.1 IOC7 — Input Capture and Output Compare Channel 7 . . . 311
7.2.2 IOC6 — Input Capture and Output Compare Channel 6 . . . 311
7.2.3 IOC5 — Input Capture and Output Compare Channel 5 . . . 311
7.2.4 IOC4 — Input Capture and Output Compare Channel 4 . . . 311
7.2.5 IOC3 — Input Capture and Output Compare Channel 3 . . . 311
7.2.6 IOC2 — Input Capture and Output Compare Channel 2 . . . 311
7.2.7 IOC1 — Input Capture and Output Compare Channel 1 . . . 311
7.2.8 IOC0 — Input Capture and Output Compare Channel 0 . . . 311
7.3 Memory Map and Register Definition . . . 312
7.3.1 Module Memory Map . . . 312
7.3.2 Register Descriptions . . . 314
7.4 Functional Description . . . 350
7.4.1 Enhanced Capture Timer Modes of Operation . . . 357
7.4.2 Reset . . . 360

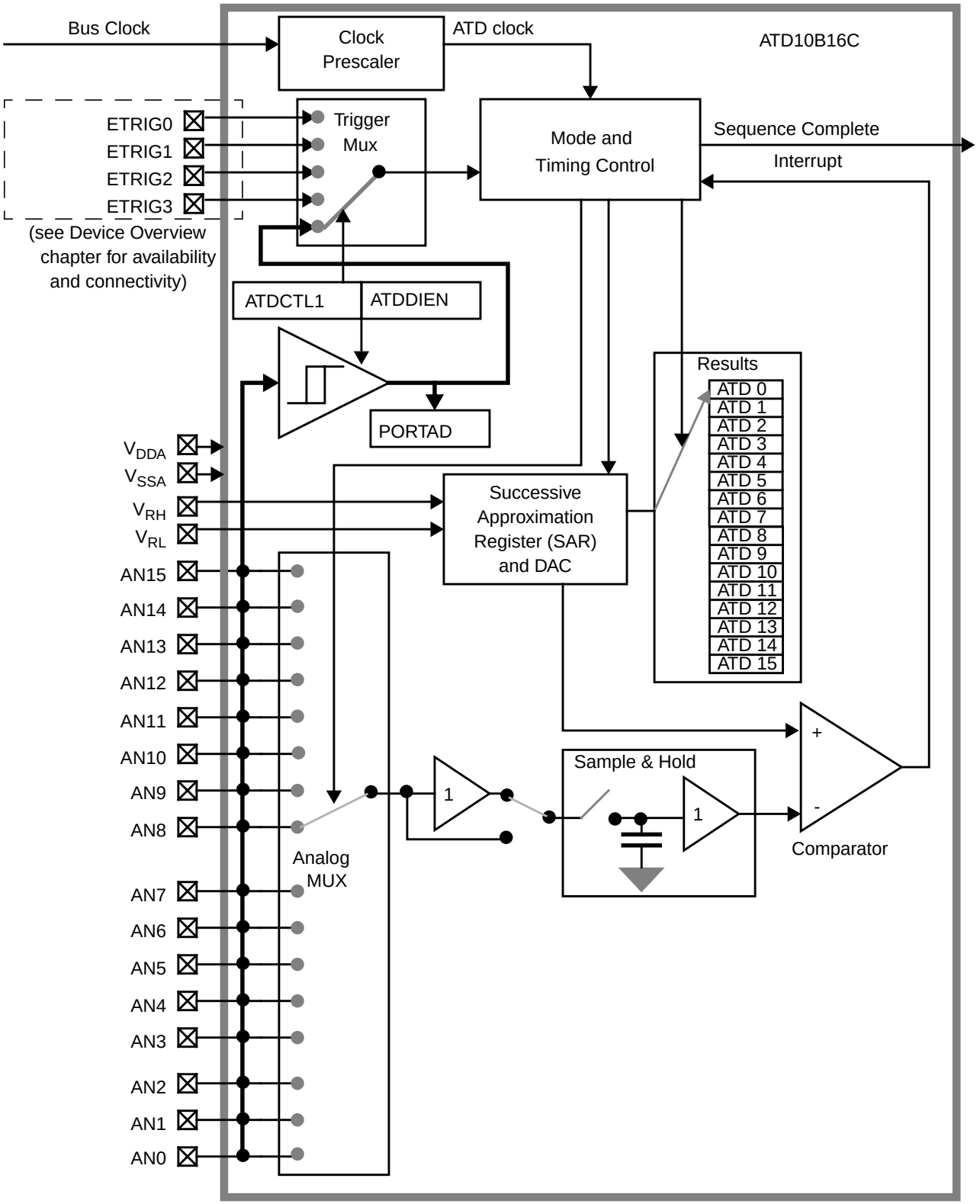


Figure 4-1. ATD10B16C Block Diagram

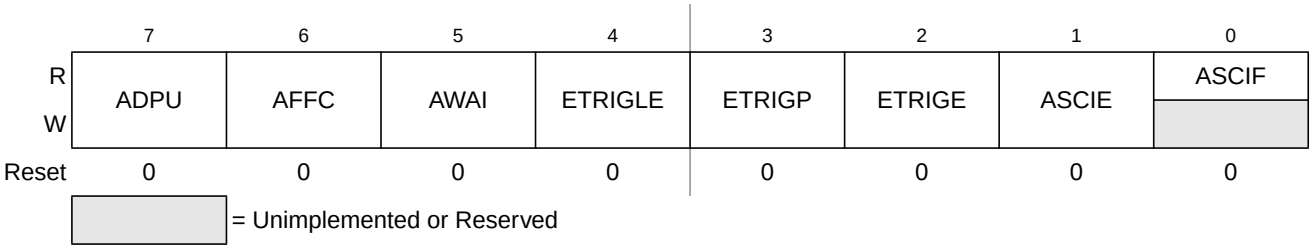


Figure 4-5. ATD Control Register 2 (ATDCTL2)

Read: Anytime

Write: Anytime

Table 4-6. ATDCTL2 Field Descriptions

Field	Description
7 ADPU	**ATD Power Down** — This bit provides on/off control over the ATD10B16C block allowing reduced MCU power consumption. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after ADPU bit is enabled. 0 Power down ATD 1 Normal ATD functionality
6 AFFC	**ATD Fast Flag Clear All** 0 ATD flag clearing operates normally (read the status register ATDSTAT1 before reading the result register to clear the associate CCF flag). 1 Changes all ATD conversion complete flags to a fast clear sequence. Any access to a result register will cause the associate CCF flag to clear automatically.
5 AWAI	**ATD Power Down in Wait Mode** — When entering Wait Mode this bit provides on/off control over the ATD10B16C block allowing reduced MCU power. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after exit from Wait mode. 0 ATD continues to run in Wait mode 1 Halt conversion and power down ATD during Wait mode After exiting Wait mode with an interrupt conversion will resume. But due to the recovery time the result of this conversion should be ignored.
4 ETRIGLE	**External Trigger Level/Edge Control** — This bit controls the sensitivity of the external trigger signal. See Table 4-7 for details.
3 ETRIGP	**External Trigger Polarity** — This bit controls the polarity of the external trigger signal. See Table 4-7 for details.
2 ETRIGE	**External Trigger Mode Enable** — This bit enables the external trigger on one of the AD channels or one of the ETRIG[3:0] inputs as described in Table 4-5. If external trigger source is one of the AD channels, the digital input buffer of this channel is enabled. The external trigger allows to synchronize the start of conversion with external events. 0 Disable external trigger 1 Enable external trigger
1 ASCIE	**ATD Sequence Complete Interrupt Enable** 0 ATD Sequence Complete interrupt requests are disabled. 1 ATD Interrupt will be requested whenever ASCIF = 1 is set.
0 ASCIF	**ATD Sequence Complete Interrupt Flag** — If ASCIE = 1 the ASCIF flag equals the SCF flag (see Section 4.3.2.7, “ATD Status Register 0 (ATDSTAT0)”), else ASCIF reads zero. Writes have no effect. 0 No ATD interrupt occurred 1 ATD sequence complete interrupt pending

5.3.2.7 ATD Status Register 0 (ATDSTAT0)

This read-only register contains the sequence complete flag, overrun flags for external trigger and FIFO mode, and the conversion counter.

	7	6	5	4	3	2	1	0
R	SCF	0	ETORF	FIFOR	0	CC2	CC1	CC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-9. ATD Status Register 0 (ATDSTAT0)

Read: Anytime

Write: Anytime (No effect on (CC2, CC1, CC0))

Table 5-17. ATDSTAT0 Field Descriptions

Field	Description
7 SCF	**Sequence Complete Flag** — This flag is set upon completion of a conversion sequence. If conversion sequences are continuously performed (SCAN = 1), the flag is set after each one is completed. This flag is cleared when one of the following occurs: A) Write “1” to SCF B) Write to ATDCTL5 (a new conversion sequence is started) C) If AFFC=1 and read of a result register 0 Conversion sequence not completed 1 Conversion sequence has completed
5 ETORF	**External Trigger Overrun Flag** — While in edge trigger mode (ETRIGLE = 0), if additional active edges are detected while a conversion sequence is in process the overrun flag is set. This flag is cleared when one of the following occurs: A) Write “1” to ETORF B) Write to ATDCTL2, ATDCTL3 or ATDCTL4 (a conversion sequence is aborted) C) Write to ATDCTL5 (a new conversion sequence is started) 0 No External trigger over run error has occurred 1 External trigger over run error has occurred
4 FIFOR	**FIFO Over Run Flag** — This bit indicates that a result register has been written to before its associated conversion complete flag (CCF) has been cleared. This flag is most useful when using the FIFO mode because the flag potentially indicates that result registers are out of sync with the input channels. However, it is also practical for non-FIFO modes, and indicates that a result register has been over written before it has been read (i.e., the old data has been lost). This flag is cleared when one of the following occurs: A) Write “1” to FIFOR B) Start a new conversion sequence (write to ATDCTL5 or external trigger) 0 No over run has occurred 1 An over run condition exists
2–0 CC[2:0]	**Conversion Counter** — These 3 read-only bits are the binary value of the conversion counter. The conversion counter points to the result register that will receive the result of the current conversion. E.g. CC2 = 1, CC1 = 1, CC0 = 0 indicates that the result of the current conversion will be in ATD result register 6. If in non-FIFO mode (FIFO = 0) the conversion counter is initialized to zero at the begin and end of the conversion sequence. If in FIFO mode (FIFO = 1) the register counter is not initialized. The conversion counters wraps around when its maximum value is reached. Aborting a conversion or starting a new conversion by write to an ATDCTL register (ATDCTL5-0) clears the conversion counter even if FIFO=1.

6.8.1.1 Naming Conventions

RD	Destination register, allowed range is R0–R7
RD.L	Low byte of the destination register, bits [7:0]
RD.H	High byte of the destination register, bits [15:8]
RS, RS1, RS2	Source register, allowed range is R0–R7
RS.L, RS1.L, RS2.L	Low byte of the source register, bits [7:0]
RS.H, RS1.H, RS2.H	High byte of the source register, bits[15:8]
RB	Base register for indexed addressing modes, allowed range is R0–R7
RI	Offset register for indexed addressing modes with register offset, allowed range is R0–R7
RI+	Offset register for indexed addressing modes with register offset and post-increment, Allowed range is R0–R7 (R0+ is equivalent to R0)
–RI	Offset register for indexed addressing modes with register offset and pre-decrement, Allowed range is R0–R7 (–R0 is equivalent to R0)

NOTE

Even though register R1 is intended to be used as a pointer to the variable segment, it may be used as a general purpose data register as well.

Selecting R0 as destination register will discard the result of the instruction. Only the condition code register will be updated

6.8.1.2 Inherent Addressing Mode (INH)

Instructions that use this addressing mode either have no operands or all operands are in internal XGATE registers:.

Examples

```
BRK
RTS
```

6.8.1.3 Immediate 3-Bit Wide (IMM3)

Operands for immediate mode instructions are included in the instruction stream and are fetched into the instruction queue along with the rest of the 16 bit instruction. The ’#’ symbol is used to indicate an immediate addressing mode operand. This address mode is used for semaphore instructions.

Examples:

```
CSEM    #1      ; Unlock semaphore 1
SSEM    #3      ; Lock Semaphore 3
```

NOTE

Writing to these registers when in special modes can alter the PWM functionality.

8.3.2.12 PWM Channel Counter Registers (PWMCNTx)

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source. The counter can be read at any time without affecting the count or the operation of the PWM channel. In left aligned output mode, the counter counts from 0 to the value in the period register - 1. In center aligned output mode, the counter counts from 0 up to the value in the period register and then back down to 0.

Any value written to the counter causes the counter to reset to $00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. The counter is also cleared at the end of the effective period (see Section 8.4.2.5, “Left Aligned Outputs” and Section 8.4.2.6, “Center Aligned Outputs” for more details). When the channel is disabled (PWMEx = 0), the PWMCNTx register does not count. When a channel becomes enabled (PWMEx = 1), the associated PWM counter starts at the count in the PWMCNTx register. For more detailed information on the operation of the counters, see Section 8.4.2.4, “PWM Timer Counters”.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

NOTE

Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

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

Figure 8-14. PWM Channel Counter Registers (PWMCNTx)

Read: Anytime

Write: Anytime (any value written causes PWM counter to be reset to $00).

8.3.2.13 PWM Channel Period Registers (PWMPERx)

There is a dedicated period register for each channel. The value in this register determines the period of the associated PWM channel.

The period registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to $00)
- The channel is disabled

Table 9-4. Multiplier Factor

IBC7-6	MUL
00	01
01	02
10	04
11	RESERVED

The number of clocks from the falling edge of SCL to the first tap (Tap[1]) is defined by the values shown in the scl2tap column of Table 9-3, all subsequent tap points are separated by 2^{IBC5-3} as shown in the tap2tap column in Table 9-3. The SCL Tap is used to generated the SCL period and the SDA Tap is used to determine the delay from the falling edge of SCL to SDA changing, the SDA hold time.

IBC7–6 defines the multiplier factor MUL. The values of MUL are shown in the Table 9-4.

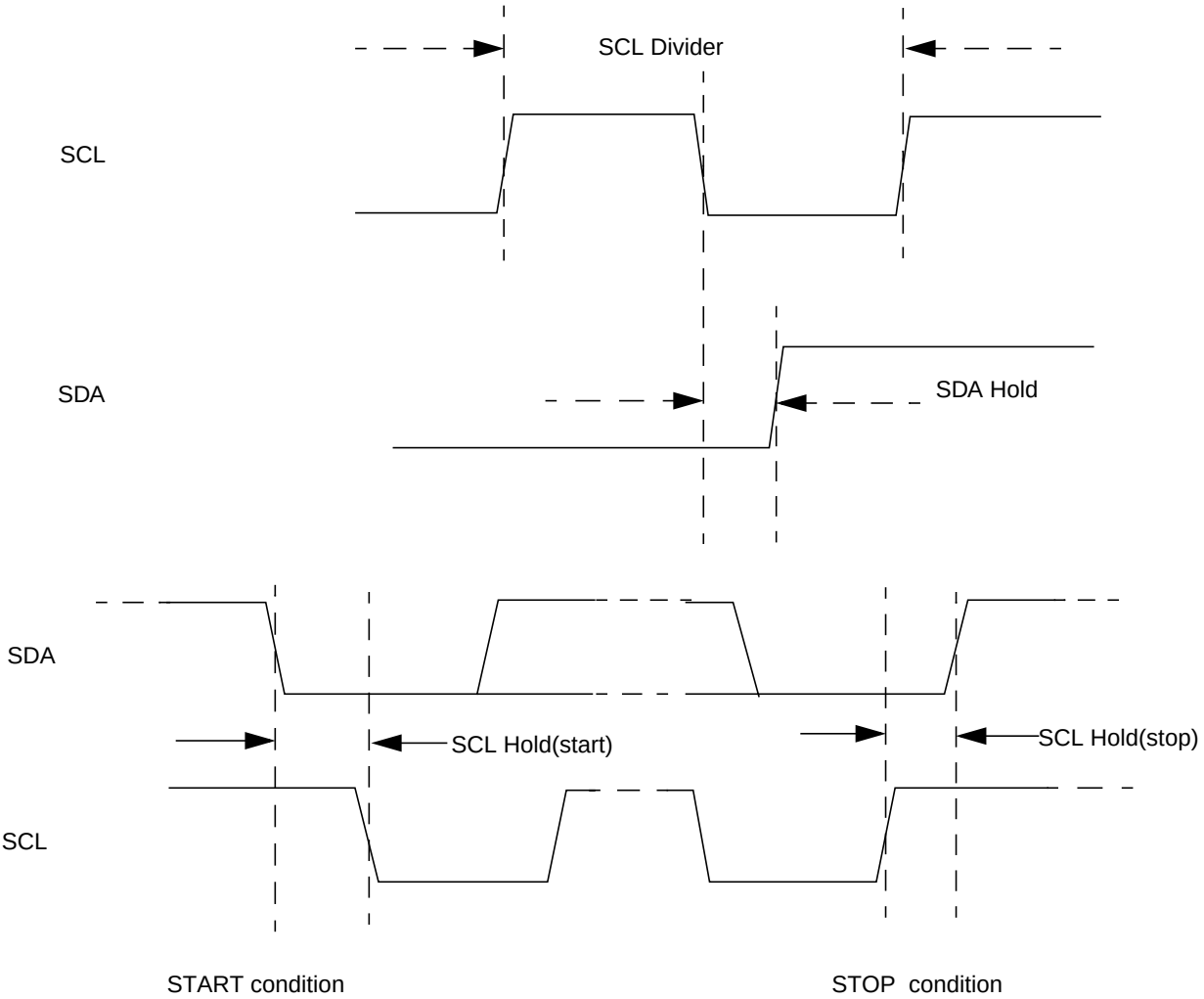


Figure 9-5. SCL Divider and SDA Hold

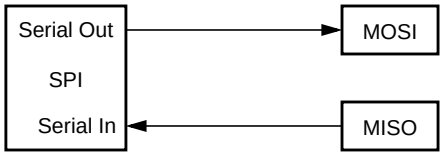
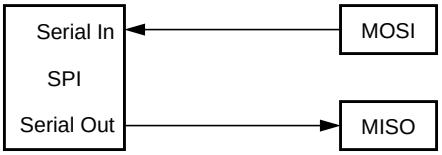
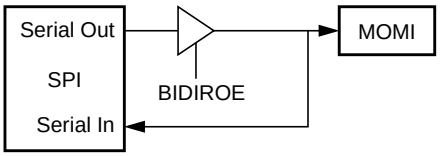
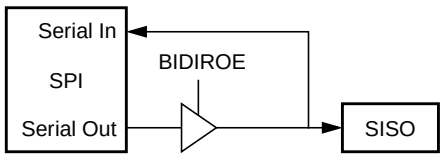
The equation used to generate the divider values from the IBFD bits is:

SCL Divider = MUL x {2 x (scl2tap + [(SCL_Tap -1) x tap2tap] + 2)}

12.4.5.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI control register 2 (see Table 12-8). In this mode, the SPI uses only one serial data pin for the interface with external device(s). The MSTR bit decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in master mode and MOSI pin in slave mode are not used by the SPI.

Table 12-8. Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode SPC0 = 0		
Bidirectional Mode SPC0 = 1		

The direction of each serial I/O pin depends on the BIDIROE bit. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

- The SCK is output for the master mode and input for the slave mode.
- The $\overline{SS}$ is the input or output for the master mode, and it is always the input for the slave mode.
- The bidirectional mode does not affect SCK and $\overline{SS}$ functions.

NOTE

In bidirectional master mode, with mode fault enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode. In this case MISO becomes occupied by the SPI and MOSI is not used. This must be considered, if the MISO pin is used for another purpose.

12.4.7.2 SPI in Wait Mode

SPI operation in wait mode depends upon the state of the SPISWAI bit in SPI control register 2.

- If SPISWAI is clear, the SPI operates normally when the CPU is in wait mode
- If SPISWAI is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
  - If SPISWAI is set and the SPI is configured for master, any transmission and reception in progress stops at wait mode entry. The transmission and reception resumes when the SPI exits wait mode.
  - If SPISWAI is set and the SPI is configured as a slave, any transmission and reception in progress continues if the SCK continues to be driven from the master. This keeps the slave synchronized to the master and the SCK.

    If the master transmits several bytes while the slave is in wait mode, the slave will continue to send out bytes consistent with the operation mode at the start of wait mode (i.e., if the slave is currently sending its SPIDR to the master, it will continue to send the same byte. Else if the slave is currently sending the last received byte from the master, it will continue to send each previous master byte).

NOTE

Care must be taken when expecting data from a master while the slave is in wait or stop mode. Even though the shift register will continue to operate, the rest of the SPI is shut down (i.e., a SPIF interrupt will **not** be generated until exiting stop or wait mode). Also, the byte from the shift register will not be copied into the SPIDR register until after the slave SPI has exited wait or stop mode. In slave mode, a received byte pending in the receive shift register will be lost when entering wait or stop mode. An SPIF flag and SPIDR copy is generated only if wait mode is entered or exited during a tranmission. If the slave enters wait mode in idle mode and exits wait mode in idle mode, neither a SPIF nor a SPIDR copy will occur.

12.4.7.3 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.

If the interrupt source is unknown (for example, in the case where an interrupt request becomes inactive after the interrupt has been recognized, but prior to the vector request), the vector address supplied to the CPU will default to that of the spurious interrupt vector.

NOTE

Care must be taken to ensure that all exception requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not get processed at all or the result may be a spurious interrupt request (vector at address (vector base + 0x0010)).

16.4.5 Reset Exception Requests

The XINT supports three system reset exception request types (please refer to CRG for details):

1. Pin reset, power-on reset, low-voltage reset, or illegal address reset
2. Clock monitor reset request
3. COP watchdog reset request

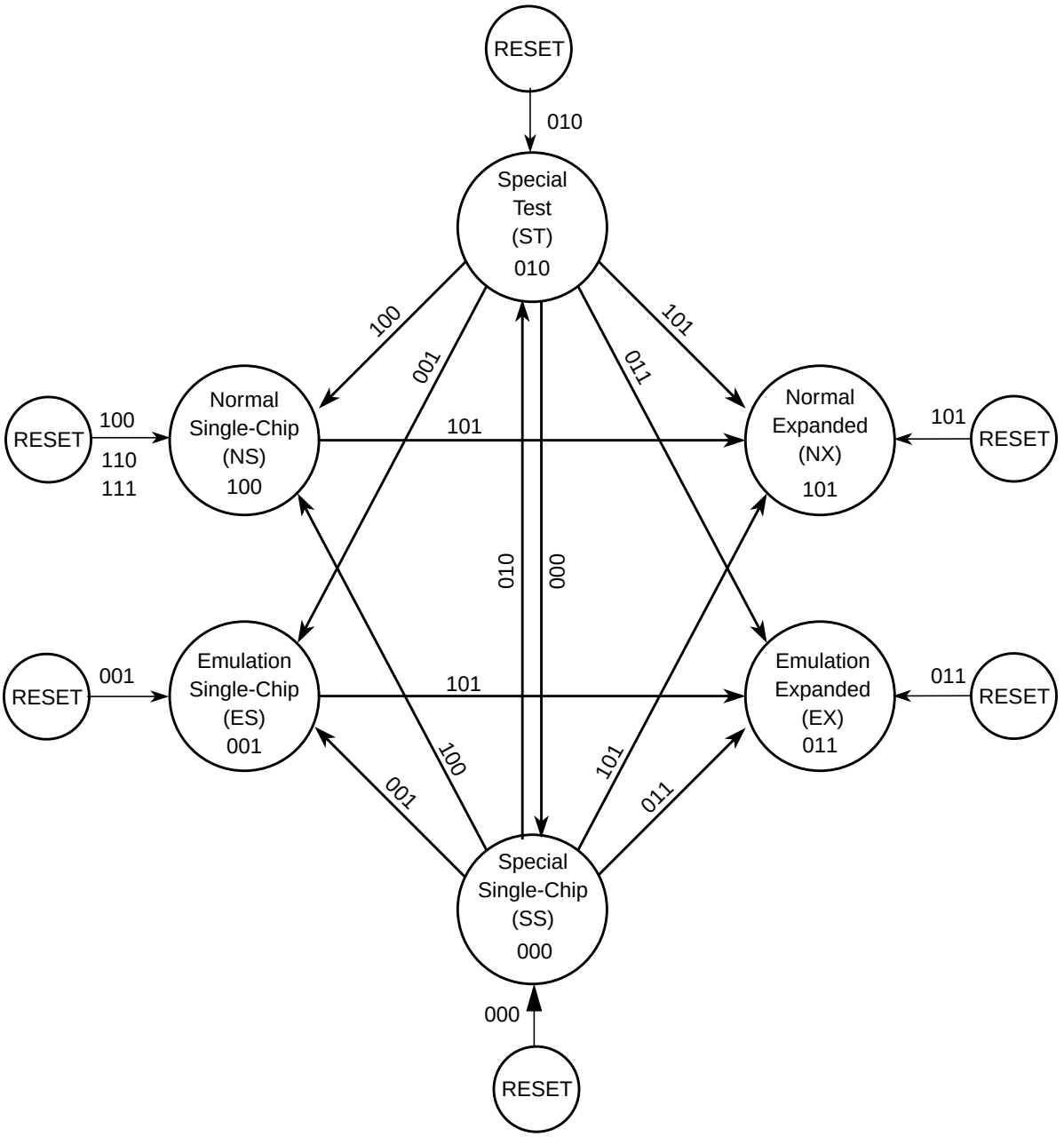
16.4.6 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the XINT upon request by the CPU is shown in Table 16-8.

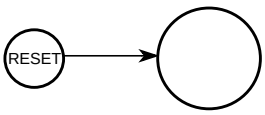
Table 16-8. Exception Vector Map and Priority

Vector Address[1]	Source
0xFFFE	Pin reset, power-on reset, low-voltage reset, illegal address reset
0xFFFC	Clock monitor reset
0xFFFA	COP watchdog reset
(Vector base + 0x00F8)	Unimplemented opcode trap
(Vector base + 0x00F6)	Software interrupt instruction (SWI) or BDM vector request
(Vector base + 0x00F4)	$\overline{\text{XIRQ}}$ interrupt request
(Vector base + 0x00F2)	$\overline{\text{IRQ}}$ interrupt request
(Vector base + 0x00F0–0x0012)	Device specific I bit maskable interrupt sources (priority determined by the associated configuration registers, in descending order)
(Vector base + 0x0010)	Spurious interrupt

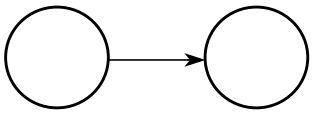
1 16 bits vector address based



Transition done by external pins (MODC, MODB, MODA)



Transition done by write access to the MODE register



110
111

Illegal (MODC, MODB, MODA) pin values.

Do not use. (Reserved for future use).

Figure 17-5. Mode Transition Diagram when MCU is Unsecured

17.3.2.5 MMC Control Register (MMCCTL1)

Address: 0x0013 PRR

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	EROMON	ROMHM	ROMON
W								
Reset	0	0	0	0	0	EROMCTL	0	ROMCTL

= Unimplemented or Reserved

Figure 17-10. MMC Control Register (MMCCTL1)

Read: Anytime. In emulation modes read operations will return the data from the external bus. In all other modes the data are read from this register.

Write: Refer to each bit description. In emulation modes write operations will also be directed to the external bus.

CAUTION

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

Table 17-9. MMCCTL1 Field Descriptions

Field	Description
2 EROMON	**Enables emulated Flash or ROM memory in the memory map** Write: Never 0 Disables the emulated Flash or ROM in the memory map. 1 Enables the emulated Flash or ROM in the memory map.
1 ROMHM	**FLASH or ROM only in higher Half of Memory Map** Write: Once in normal and emulation modes and anytime in special modes 0 The fixed page of Flash or ROM can be accessed in the lower half of the memory map. Accesses to $4000–$7FFF will be mapped to $7F_4000-$7F_7FFF in the global memory space. 1 Disables access to the Flash or ROM in the lower half of the memory map.These physical locations of the Flash or ROM can still be accessed through the program page window. Accesses to $4000–$7FFF will be mapped to 14_4000-14_7FFF in the global memory space (external access).
0 ROMON	**Enable FLASH or ROM in the memory map** Write: Once in normal and emulation modes and anytime in special modes 0 Disables the Flash or ROM from the memory map. 1 Enables the Flash or ROM in the memory map.

EROMON and ROMON control the visibility of the Flash in the memory map for CPU or BDM (not for XGATE). Both local and global memory maps are affected.

18.3.2.10 RAM XGATE Upper Boundary Register (RAMXGU)

Address: 0x011D

	7	6	5	4	3	2	1	0
R	1	XGU6	XGU5	XGU4	XGU3	XGU2	XGU1	XGU0
W								
Reset	1	1	1	1	1	1	1	1

= Unimplemented or Reserved

Figure 18-18. RAM XGATE Upper Boundary Register (RAMXGU)

Read: Anytime

Write: Anytime when RWPE = 0

Table 18-16. RAMXGU Field Descriptions

Field	Description
6–0 XGU[6:0]	**XGATE Region Upper Boundary Bits 6-0** — These bits define the upper boundary of the RAM region allocated to the XGATE module in multiples of 256 bytes. The 256 byte block selected by this register is included in the region. See Figure 18-25 for details.

18.3.2.11 RAM Shared Region Lower Boundary Register (RAMSHL)

Address: 0x011E

	7	6	5	4	3	2	1	0
R	1	SHL6	SHL5	SHL4	SHL3	SHL2	SHL1	SHL0
W								
Reset	1	1	1	1	1	1	1	1

= Unimplemented or Reserved

Figure 18-19. RAM Shared Region Lower Boundary Register (RAMSHL)

Read: Anytime

Write: Anytime when RWPE = 0

Table 18-17. RAMSHL Field Descriptions

Field	Description
6–0 SHL[6:0]	**RAM Shared Region Lower Boundary Bits 6–0** — These bits define the lower boundary of the shared memory region in multiples of 256 bytes. The block selected by this register is included in the region. See Figure 18-25 for details.

26.5 Operating Modes

26.5.1 Wait Mode

If a command is active (CCIF = 0) when the MCU enters the wait mode, the active command and any buffered command will be completed.

The EEPROM module can recover the MCU from wait mode if the CBEIF and CCIF interrupts are enabled (see Section 26.8, “Interrupts”).

26.5.2 Stop Mode

If a command is active (CCIF = 0) when the MCU enters the stop mode, the operation will be aborted and, if the operation is program, sector erase, mass erase, or sector modify, the EEPROM array data being programmed or erased may be corrupted and the CCIF and ACCERR flags will be set. If active, the high voltage circuitry to the EEPROM memory will immediately be switched off when entering stop mode. Upon exit from stop mode, the CBEIF flag is set and any buffered command will not be launched. The ACCERR flag must be cleared before starting a command write sequence (see Section 26.4.1.2, “Command Write Sequence”).

NOTE

As active commands are immediately aborted when the MCU enters stop mode, it is strongly recommended that the user does not use the STOP instruction during program, sector erase, mass erase, or sector modify operations.

26.5.3 Background Debug Mode

In background debug mode (BDM), the EPROT register is writable. If the MCU is unsecured, then all EEPROM commands listed in Table 26-10 can be executed. If the MCU is secured and is in special single chip mode, the only command available to execute is mass erase.

26.6 EEPROM Module Security

The EEPROM module does not provide any security information to the MCU. After each reset, the security state of the MCU is a function of information provided by the Flash module (see the specific FTX Block Guide).

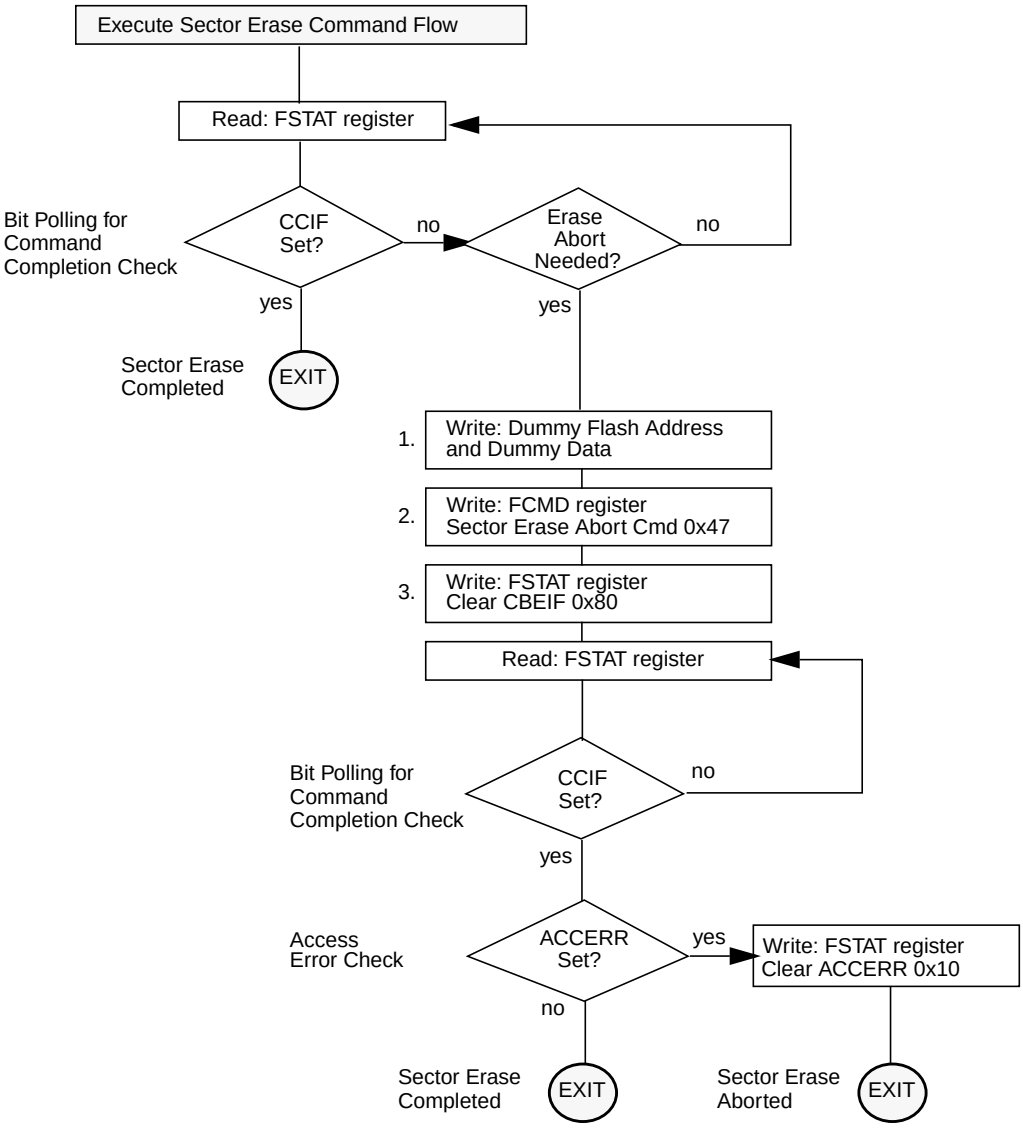


Figure 27-31. Example Sector Erase Abort Command Flow

CBEIF, PVIOL, and ACCERR are readable and writable, CCIF and BLANK are readable and not writable, remaining bits read 0 and are not writable in normal mode. FAIL is readable and writable in special mode. FAIL must be clear in special mode when starting a command write sequence.

Table 29-14. FSTAT Field Descriptions

Field	Description
7 CBEIF	**Command Buffer Empty Interrupt Flag** — The CBEIF flag indicates that the address, data and command buffers are empty so that a new command write sequence can be started. Writing a 0 to the CBEIF flag has no effect on CBEIF. Writing a 0 to CBEIF after writing an aligned word to the Flash address space, but before CBEIF is cleared, will abort a command write sequence and cause the ACCERR flag to be set. Writing a 0 to CBEIF outside of a command write sequence will not set the ACCERR flag. The CBEIF flag is cleared by writing a 1 to CBEIF. The CBEIF flag is used together with the CBEIE bit in the FCNFG register to generate an interrupt request (see Figure 29-30). 0 Command buffers are full. 1 Command buffers are ready to accept a new command.
6 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that there are no more commands pending. The CCIF flag is cleared when CBEIF is cleared and sets automatically upon completion of all active and pending commands. The CCIF flag does not set when an active commands completes and a pending command is fetched from the command buffer. Writing to the CCIF flag has no effect on CCIF. The CCIF flag is used together with the CCIE bit in the FCNFG register to generate an interrupt request (see Figure 29-30). 0 Command in progress. 1 All commands are completed.
5 PVIOL	**Protection Violation Flag** —The PVIOL flag indicates an attempt was made to program or erase an address in a protected area of the Flash memory during a command write sequence. Writing a 0 to the PVIOL flag has no effect on PVIOL. The PVIOL flag is cleared by writing a 1 to PVIOL. While PVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation detected. 1 Protection violation has occurred.
4 ACCERR	**Access Error Flag** — The ACCERR flag indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see Section 29.4.1.2, “Command Write Sequence”), issuing an illegal Flash command (see Table 29-16), launching the sector erase abort command terminating a sector erase operation early (see Section 29.4.2.6, “Sector Erase Abort Command”) or the execution of a CPU STOP instruction while a command is executing (CCIF = 0). Writing a 0 to the ACCERR flag has no effect on ACCERR. The ACCERR flag is cleared by writing a 1 to ACCERR.While ACCERR is set, it is not possible to launch a command or start a command write sequence. If ACCERR is set by an erase verify operation or a data compress operation, any buffered command will not launch. 0 No access error detected. 1 Access error has occurred.
2 BLANK	**Flag Indicating the Erase Verify Operation Status** — When the CCIF flag is set after completion of an erase verify command, the BLANK flag indicates the result of the erase verify operation. The BLANK flag is cleared by the Flash module when CBEIF is cleared as part of a new valid command write sequence. Writing to the BLANK flag has no effect on BLANK. 0 Flash block verified as not erased. 1 Flash block verified as erased.
1 FAIL	**Flag Indicating a Failed Flash Operation** — The FAIL flag will set if the erase verify operation fails (Flash block verified as not erased). Writing a 0 to the FAIL flag has no effect on FAIL. The FAIL flag is cleared by writing a 1 to FAIL. 0 Flash operation completed without error. 1 Flash operation failed.

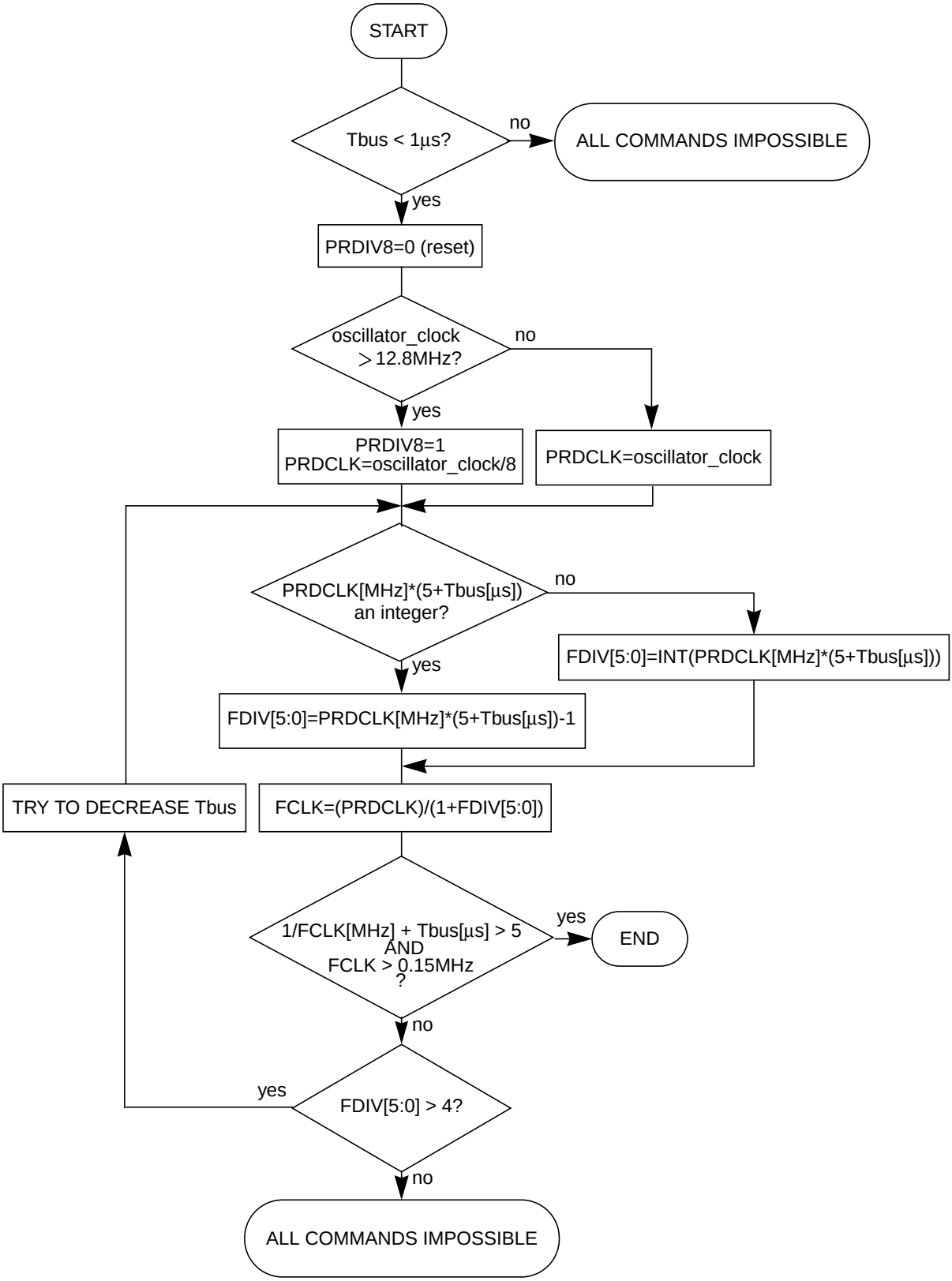


Figure 29-22. Determination Procedure for PRDIV8 and FDIV Bits