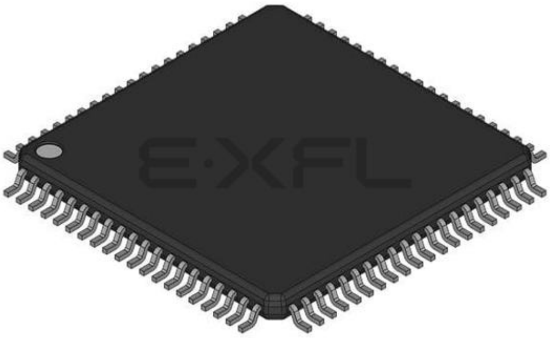

Welcome to E-XFL.COM

## 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 | HCS12 |
| Core Size | 16-Bit |
| Speed | 25MHz |
| Connectivity | CANbus, EBI/EMI, SCI, SPI |
| Peripherals | POR, PWM, WDT |
| Number of I/O | 60 |
| Program Memory Size | 128KB (128K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 4K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.35V ~ 5.5V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 105°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=mc9s12c128vfue |

#### 1.3.4.13 PE2 / R/$\overline{\text{W}}$ — Port E I/O Pin [2] / Read/Write

In all modes this pin can be used as a general-purpose I/O and is an input with an active pull-up out of reset. If the read/write function is required it should be enabled by setting the RDWE bit in the PEAR register. External writes will not be possible until enabled. This pin is not available in the 48- / 52-pin package versions.

#### 1.3.4.14 PE1 / $\overline{\text{IRQ}}$ — Port E Input Pin [1] / Maskable Interrupt Pin

The $\overline{\text{IRQ}}$ input provides a means of applying asynchronous interrupt requests to the MCU. Either falling edge-sensitive triggering or level-sensitive triggering is program selectable (INTCR register). $\overline{\text{IRQ}}$ is always enabled and configured to level-sensitive triggering out of reset. It can be disabled by clearing IRQEN bit (INTCR register). When the MCU is reset the $\overline{\text{IRQ}}$ function is masked in the condition code register. This pin is always an input and can always be read. There is an active pull-up on this pin while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register.

#### 1.3.4.15 PE0 / $\overline{\text{XIRQ}}$ — Port E input Pin [0] / Non Maskable Interrupt Pin

The $\overline{\text{XIRQ}}$ input provides a means of requesting a non-maskable interrupt after reset initialization. During reset, the X bit in the condition code register (CCR) is set and any interrupt is masked until MCU software enables it. Because the $\overline{\text{XIRQ}}$ input is level sensitive, it can be connected to a multiple-source wired-OR network. This pin is always an input and can always be read. There is an active pull-up on this pin while in reset and immediately out of reset. The pull-up can be turned off by clearing PUPEE in the PUCR register.

#### 1.3.4.16 PAD[7:0] / AN[7:0] — Port AD I/O Pins [7:0]

PAD7–PAD0 are general purpose I/O pins and also analog inputs for the analog to digital converter. In order to use a PAD pin as a standard input, the corresponding ATDDIEN register bit must be set. These bits are cleared out of reset to configure the PAD pins for A/D operation.

When the A/D converter is active in multi-channel mode, port inputs are scanned and converted irrespective of Port AD configuration. Thus Port AD pins that are configured as digital inputs or digital outputs are also converted in the A/D conversion sequence.

#### 1.3.4.17 PP[7] / KWP[7] — Port P I/O Pin [7]

PP7 is a general purpose input or output pin, shared with the keypad interrupt function. When configured as an input, it can generate interrupts causing the MCU to exit stop or wait mode. This pin is not available in the 48- / 52-pin package versions.

#### 1.3.4.18 PP[6] / KWP[6]/ROMCTL — Port P I/O Pin [6]

PP6 is a general purpose input or output pin, shared with the keypad interrupt function. When configured as an input, it can generate interrupts causing the MCU to exit stop or wait mode. This pin is not available in the 48- / 52-pin package versions. During MCU expanded modes of operation, this pin is used to enable

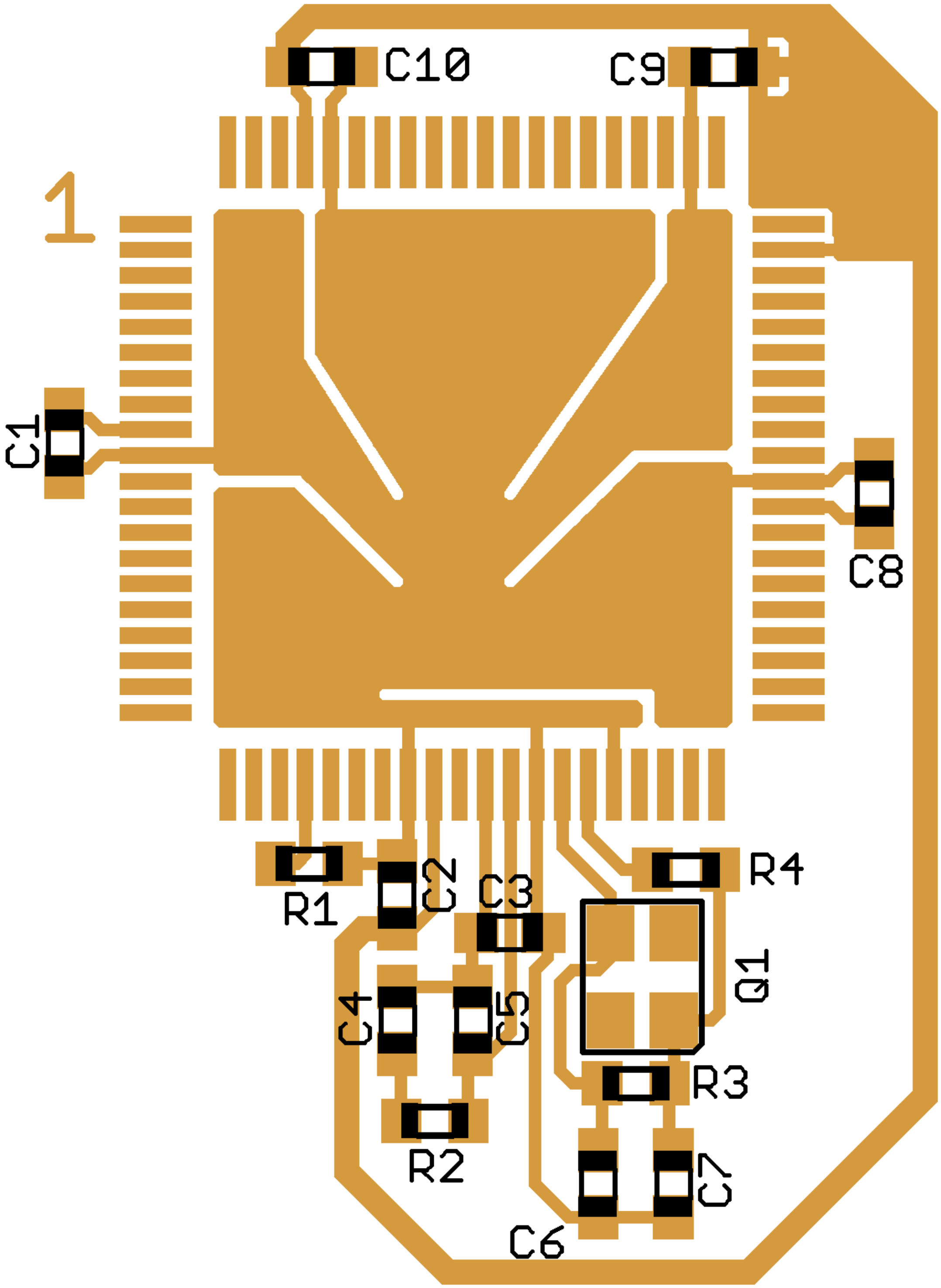


**Figure 1-20. Recommended PCB Layout for 80QFP Pierce Oscillator**

## 2.1.2 Block Diagram

Figure 2-1 is a block diagram of the PIM.

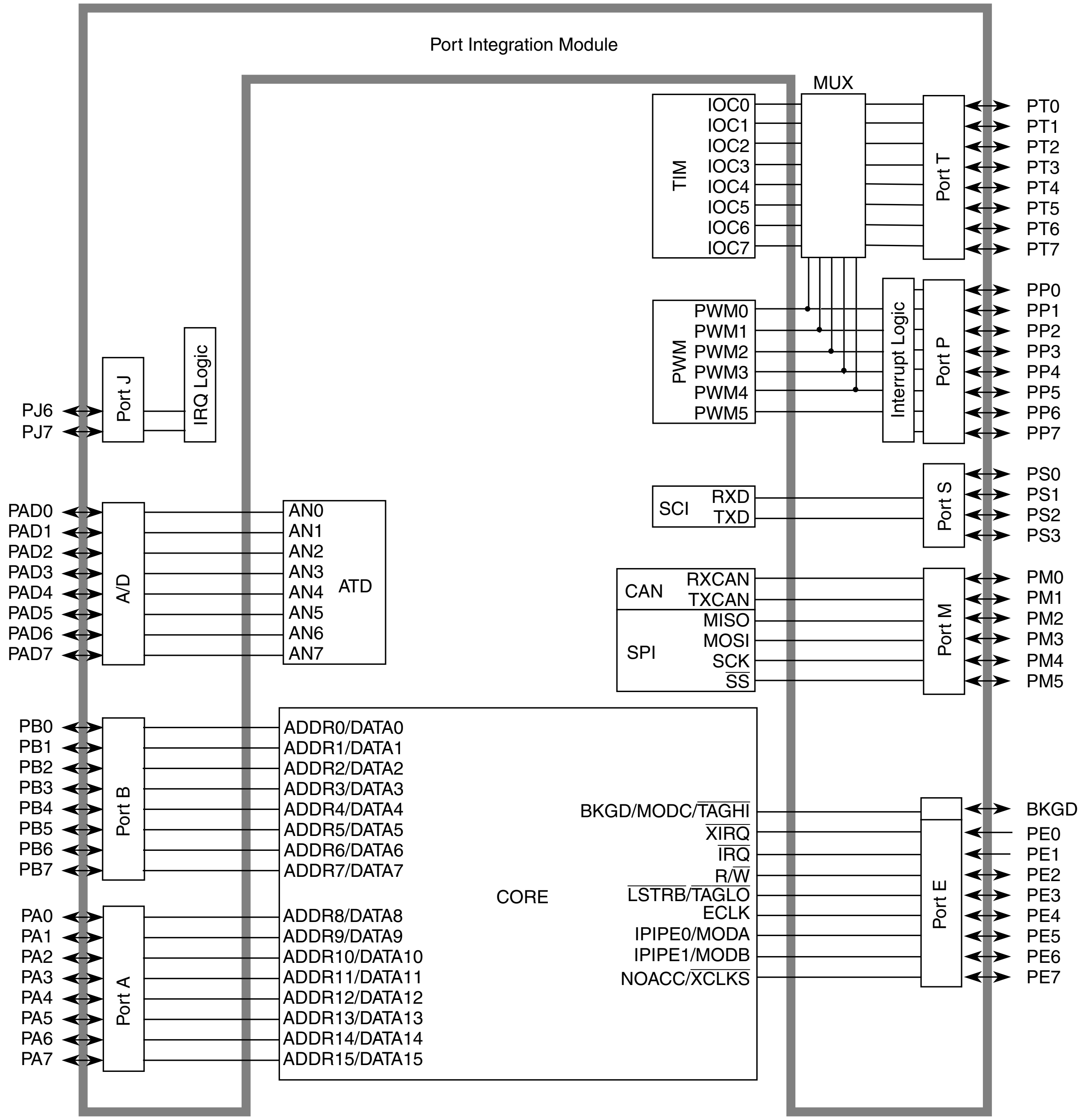


**Figure 2-1. PIM Block Diagram**

Note: The MODRR register within the PIM allows for mapping of PWM channels to Port T in the absence of Port P pins for the low pin count packages. For the 80QFP package option it is recommended not to use MODRR since this is intended to support PWM channel availability in low pin count packages. Note that

## 2.4 Functional Description

Each pin can act as general purpose I/O. In addition the pin can act as an output from a peripheral module or an input to a peripheral module.

A set of configuration registers is common to all ports. All registers can be written at any time, however a specific configuration might not become active.

Example: Selecting a pull-up resistor. This resistor does not become active while the port is used as a push-pull output.

### 2.4.1 Registers

#### 2.4.1.1 I/O Register

This register holds the value driven out to the pin if the port is used as a general purpose I/O. Writing to this register has only an effect on the pin if the port is used as general purpose output. When reading this address, the value of the pins are returned if the data direction register bits are set to 0.

If the data direction register bits are set to 1, the contents of the I/O register is returned. This is independent of any other configuration (Figure 2-46).

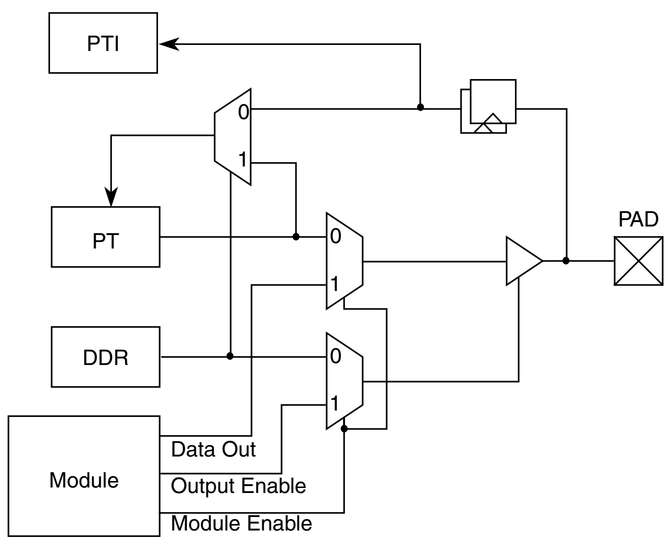


**Figure 2-46. Illustration of I/O Pin Functionality**

#### 2.4.1.2 Input Register

This is a read-only register and always returns the value of the pin (Figure 2-46).

#### 2.4.1.3 Data Direction Register

This register defines whether the pin is used as an input or an output. If a peripheral module controls the pin the contents of the data direction register is ignored (Figure 2-46).

# Chapter 3
# Module Mapping Control (MMCV4) Block Description

## 3.1 Introduction

This section describes the functionality of the module mapping control (MMC) sub-block of the S12 core platform.

The block diagram of the MMC is shown in Figure 3-1.

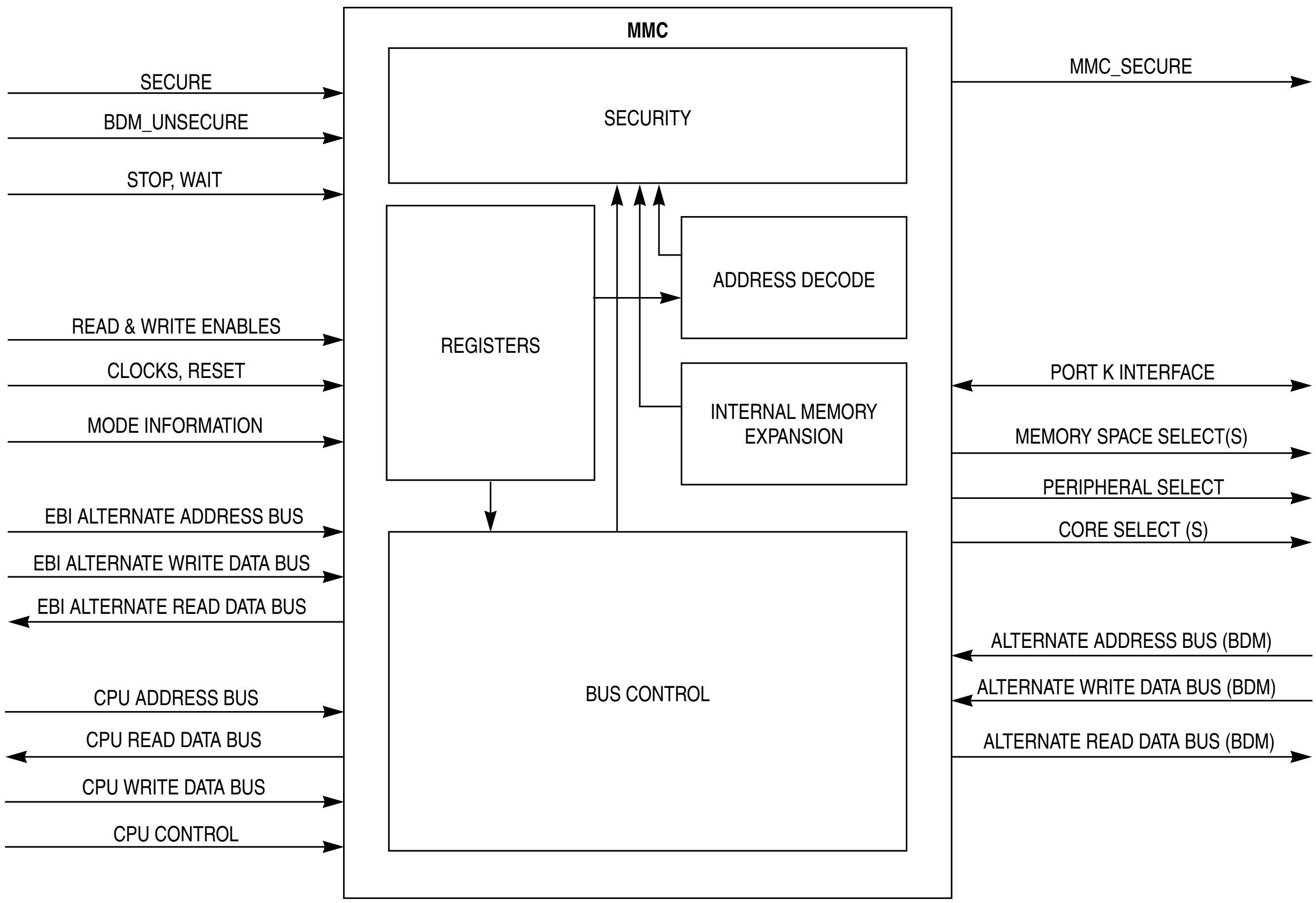


**Figure 3-1. MMC Block Diagram**

The MMC is the sub-module which controls memory map assignment and selection of internal resources and external space. Internal buses between the core and memories and between the core and peripherals is controlled in this module. The memory expansion is generated in this module.

**Table 3-9. Allocated RAM Memory Space (continued)**

| ram_sw2:ram_sw0 | Allocated RAM Space | RAM Mappable Region | INITRM Bits Used | RAM Reset Base Address[1] |
|---|---|---|---|---|
| 011 | 8K bytes | 8K bytes | RAM[15:13] | 0x0000 |
| 100 | 10K bytes | 16K bytes [2] | RAM[15:14] | 0x1800 |
| 101 | 12K bytes | 16K bytes [2] | RAM[15:14] | 0x1000 |
| 110 | 14K bytes | 16K bytes [2] | RAM[15:14] | 0x0800 |
| 111 | 16K bytes | 16K bytes | RAM[15:14] | 0x0000 |

1. The RAM Reset BASE Address is based on the reset value of the INITRM register, 0x0009.
2. Alignment of the Allocated RAM space within the RAM mappable region is dependent on the value of RAMHAL.

**NOTE**

As stated, the bits in this register provide read visibility to the system physical memory space allocations defined at system integration. The actual array size for any given type of memory block may differ from the allocated size. Please refer to the device overview chapter for actual sizes.

### 3.3.2.8 Memory Size Register 1 (MEMSIZ1)

Module Base + 0x001D
Starting address location affected by INITRG register setting.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | ROM_SW1 | ROM_SW0 | 0 | 0 | 0 | 0 | PAG_SW1 | PAG_SW0 |
| W | | | | | | | | |
| Reset | — | — | — | — | — | — | — | — |

= Unimplemented or Reserved

**Figure 3-10. Memory Size Register 1 (MEMSIZ1)**

Read: Anytime

Write: Writes have no effect

Reset: Defined at chip integration, see device overview section.

The MEMSIZ1 register reflects the state of the FLASH or ROM physical memory space and paging switches at the core boundary which are configured at system integration. This register allows read visibility to the state of these switches.

vector spaces, expansion windows, and on-chip memory are mapped so that their address ranges do not overlap. The MMC will make only one select signal active at any given time. This activation is based upon the priority outlined in Table 3-15. If two or more blocks share the same address space, only the select signal for the block with the highest priority will become active. An example of this is if the registers and the RAM are mapped to the same space, the registers will have priority over the RAM and the portion of RAM mapped in this shared space will not be accessible. The expansion windows have the lowest priority. This means that registers, vectors, and on-chip memory are always visible to a program regardless of the values in the page select registers.

**Table 3-15. Select Signal Priority**

| Priority | Address Space |
|---|---|
| Highest | BDM (internal to core) firmware or register space |
| ... | Internal register space |
| ... | RAM memory block |
| ... | EEPROM memory block |
| ... | On-chip FLASH or ROM |
| Lowest | Remaining external space |

In expanded modes, all address space not used by internal resources is by default external memory space. The data registers and data direction registers for ports A and B are removed from the on-chip memory map and become external accesses. If the EME bit in the MODE register (see MEBI block description chapter) is set, the data and data direction registers for port E are also removed from the on-chip memory map and become external accesses.

In special peripheral mode, the first 16 registers associated with bus expansion are removed from the on-chip memory map (PORTA, PORTB, DDRA, DDRB, PORTE, DDRE, PEAR, MODE, PUCR, RDRIV, and the EBI reserved registers).

In emulation modes, if the EMK bit in the MODE register (see MEBI block description chapter) is set, the data and data direction registers for port K are removed from the on-chip memory map and become external accesses.

### 3.4.2.2 Emulation Chip Select Signal

When the EMK bit in the MODE register (see MEBI block description chapter) is set, port K bit 7 is used as an active-low emulation chip select signal, $\overline{\text{ECS}}$. This signal is active when the system is in emulation mode, the EMK bit is set and the FLASH or ROM space is being addressed subject to the conditions outlined in Section 3.4.3.2, "Extended Address (XAB19:14) and ECS Signal Functionality." When the EMK bit is clear, this pin is used for general purpose I/O.

### 3.4.2.3 External Chip Select Signal

When the EMK bit in the MODE register (see MEBI block description chapter) is set, port K bit 6 is used as an active-low external chip select signal, $\overline{\text{XCS}}$. This signal is active only when the $\overline{\text{ECS}}$ signal described above is not active and when the system is addressing the external address space. Accesses to

#### 7.4.2.6 Capture Modes

The DBG in DBG mode can operate in four capture modes. These modes are described in the following subsections.

##### 7.4.2.6.1 Normal Mode

In normal mode, the DBG module uses comparator A and B as triggering devices. Change-of-flow information or data will be stored depending on TRG in DBGSC.

##### 7.4.2.6.2 Loop1 Mode

The intent of loop1 mode is to prevent the trace buffer from being filled entirely with duplicate information from a looping construct such as delays using the DBNE instruction or polling loops using BRSET/BRCLR instructions. Immediately after address information is placed in the trace buffer, the DBG module writes this value into the C comparator and the C comparator is placed in ignore address mode. This will prevent duplicate address entries in the trace buffer resulting from repeated bit-conditional branches. Comparator C will be cleared when the ARM bit is set in loop1 mode to prevent the previous contents of the register from interfering with loop1 mode operation. Breakpoints based on comparator C are disabled.

Loop1 mode only inhibits duplicate source address entries that would typically be stored in most tight looping constructs. It will not inhibit repeated entries of destination addresses or vector addresses, because repeated entries of these would most likely indicate a bug in the user's code that the DBG module is designed to help find.

**NOTE**

In certain very tight loops, the source address will have already been fetched again before the C comparator is updated. This results in the source address being stored twice before further duplicate entries are suppressed. This condition occurs with branch-on-bit instructions when the branch is fetched by the first P-cycle of the branch or with loop-construct instructions in which the branch is fetched with the first or second P cycle. See examples below:

```
LOOP  INCX                   ; 1-byte instruction fetched by 1st P-cycle of BRCLR
      BRCLR CMPTMP,#$0c,LOOP ; the BRCLR instruction also will be fetched by 1st P-cycle of BRCLR

LOOP2 BRN   *                ; 2-byte instruction fetched by 1st P-cycle of DBNE
      NOP                    ; 1-byte instruction fetched by 2nd P-cycle of DBNE
      DBNE  A,LOOP2          ; this instruction also fetched by 2nd P-cycle of DBNE
```

**NOTE**

Loop1 mode does not support paged memory, and inhibits duplicate entries in the trace buffer based solely on the CPU address. There is a remote possibility of an erroneous address match if program flow alternates between paged and unpaged memory space.

**Table 9-5. PLLCTL Field Descriptions (continued)**

| Field | Description |
|---|---|
| 5<br>AUTO | **Automatic Bandwidth Control Bit** — AUTO selects either the high bandwidth (acquisition) mode or the low bandwidth (tracking) mode depending on how close to the desired frequency the VCO is running. Write anytime except when PLLWAI=1, because PLLWAI sets the AUTO bit to 1.<br>0 Automatic mode control is disabled and the PLL is under software control, using ACQ bit.<br>1 Automatic mode control is enabled and ACQ bit has no effect. |
| 4<br>ACQ | **Acquisition Bit** — Write anytime. If AUTO=1 this bit has no effect.<br>0 Low bandwidth filter is selected.<br>1 High bandwidth filter is selected. |
| 2<br>PRE | **RTI Enable during Pseudo-Stop Bit** — PRE enables the RTI during pseudo-stop mode. Write anytime.<br>0 RTI stops running during pseudo-stop mode.<br>1 RTI continues running during pseudo-stop mode.<br>**Note:** If the PRE bit is cleared the RTI dividers will go static while pseudo-stop mode is active. The RTI dividers will not initialize like in wait mode with RTIWAI bit set. |
| 1<br>PCE | **COP Enable during Pseudo-Stop Bit** — PCE enables the COP during pseudo-stop mode. Write anytime.<br>0 COP stops running during pseudo-stop mode<br>1 COP continues running during pseudo-stop mode<br>**Note:** If the PCE bit is cleared the COP dividers will go static while pseudo-stop mode is active. The COP dividers will *not* initialize like in wait mode with COPWAI bit set. |
| 0<br>SCME | **Self-Clock Mode Enable Bit** — Normal modes: Write once —Special modes: Write anytime — SCME can not be cleared while operating in self-clock mode (SCM=1).<br>0 Detection of crystal clock failure causes clock monitor reset (see Section 9.5.1, “Clock Monitor Reset”).<br>1 Detection of crystal clock failure forces the MCU in self-clock mode (see Section 9.4.7.2, “Self-Clock Mode”). |

#### 9.3.2.8 CRG RTI Control Register (RTICTL)

This register selects the timeout period for the real-time interrupt.

Module Base + 0x0007

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | RTR6 | RTR5 | RTR4 | RTR3 | RTR2 | RTR1 | RTR0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 9-11. CRG RTI Control Register (RTICTL)**

Read: anytime

Write: anytime

**NOTE**

A write to this register initializes the RTI counter.

### 9.3.2.9 CRG COP Control Register (COPCTL)

This register controls the COP (computer operating properly) watchdog.

Module Base + 0x0008

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

= Unimplemented or Reserved

**Figure 9-12. CRG COP Control Register (COPCTL)**

Read: anytime

Write: WCOP, CR2, CR1, CR0: once in user mode, anytime in special mode

Write: RSBCK: once

**Table 9-8. COPCTL Field Descriptions**

| Field | Description |
|---|---|
| 7<br>WCOP | **Window COP Mode Bit** — When set, a write to the ARMCOP register must occur in the last 25% of the selected period. A write during the first 75% of the selected period will reset the part. As long as all writes occur during this window, 0x0055 can be written as often as desired. As soon as 0x00AA is written after the 0x0055, the time-out logic restarts and the user must wait until the next window before writing to ARMCOP. Table 9-9 shows the exact duration of this window for the seven available COP rates.<br>0 Normal COP operation<br>1 Window COP operation |
| 6<br>RSBCK | COP and RTI Stop in Active BDM Mode Bit<br>0 Allows the COP and RTI to keep running in active BDM mode.<br>1 Stops the COP and RTI counters whenever the part is in active BDM mode. |
| 2:0<br>CR[2:0] | **COP Watchdog Timer Rate Select** — These bits select the COP time-out rate (see Table 9-9). The COP time-out period is OSCCLK period divided by CR[2:0] value. Writing a nonzero value to CR[2:0] enables the COP counter and starts the time-out period. A COP counter time-out causes a system reset. This can be avoided by periodically (before time-out) reinitializing the COP counter via the ARMCOP register. |

**Table 9-9. COP Watchdog Rates[1]**

| CR2 | CR1 | CR0 | OSCCLK Cycles to Time Out |
|---|---|---|---|
| 0 | 0 | 0 | COP disabled |
| 0 | 0 | 1 | $2^{14}$ |
| 0 | 1 | 0 | $2^{16}$ |
| 0 | 1 | 1 | $2^{18}$ |
| 1 | 0 | 0 | $2^{20}$ |
| 1 | 0 | 1 | $2^{22}$ |
| 1 | 1 | 0 | $2^{23}$ |
| 1 | 1 | 1 | $2^{24}$ |

1. OSCCLK cycles are referenced from the previous COP time-out reset (writing 0x0055/0x00AA to the ARMCOP register)

**Table 10-8. Time Segment 1 Values**

| TSEG13 | TSEG12 | TSEG11 | TSEG10 | Time segment 1 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 Tq clock cycle[(1)] |
| 0 | 0 | 0 | 1 | 2 Tq clock cycles[1] |
| 0 | 0 | 1 | 0 | 3 Tq clock cycles[1] |
| 0 | 0 | 1 | 1 | 4 Tq clock cycles |
| : | : | : | : | : |
| 1 | 1 | 1 | 0 | 15 Tq clock cycles |
| 1 | 1 | 1 | 1 | 16 Tq clock cycles |

1. This setting is not valid. Please refer to Table 10-34 for valid settings.

The bit time is determined by the oscillator frequency, the baud rate prescaler, and the number of time quanta (Tq) clock cycles per bit (as shown in Table 10-7 and Table 10-8).

*Eqn. 10-1*

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{CANCLK}} \bullet (1 + \text{TimeSegment1} + \text{TimeSegment2})$$

### 10.3.2.5 MSCAN Receiver Flag Register (CANRFLG)

A flag can be cleared only by software (writing a 1 to the corresponding bit position) when the condition which caused the setting is no longer valid. Every flag has an associated interrupt enable bit in the CANRIER register.

Module Base + 0x0004

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | WUPIF | CSCIF | RSTAT1 | RSTAT0 | TSTAT1 | TSTAT0 | OVRIF | RXF |
| W | | | | | | | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 10-8. MSCAN Receiver Flag Register (CANRFLG)**

**NOTE**

The CANRFLG register is held in the reset state[1] when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable again as soon as the initialization mode is exited (INITRQ = 0 and INITAK = 0).

Read: Anytime

Write: Anytime when out of initialization mode, except RSTAT[1:0] and TSTAT[1:0] flags which are read-only; write of 1 clears flag; write of 0 is ignored.

1. The RSTAT[1:0], TSTAT[1:0] bits are not affected by initialization mode.

**NOTE**

Reading this register when in any other mode other than sleep or initialization mode may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

Writing to this register when in special modes can alter the MSCAN functionality.

#### 10.3.2.15 MSCAN Transmit Error Counter (CANTXERR)

This register reflects the status of the MSCAN transmit error counter.

Module Base + 0x000F

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | TXERR7 | TXERR6 | TXERR5 | TXERR4 | TXERR3 | TXERR2 | TXERR1 | TXERR0 |
| W | | | | | | | | |
| Reset: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented

**Figure 10-18. MSCAN Transmit Error Counter (CANTXERR)**

Read: Only when in sleep mode (SLPRQ = 1 and SLPAK = 1) or initialization mode (INITRQ = 1 and INITAK = 1)

Write: Unimplemented

**NOTE**

Reading this register when in any other mode other than sleep or initialization mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

Writing to this register when in special modes can alter the MSCAN functionality.

| Register Name | | Bit 7 | 6 | 5 | 4 | 3 | 2 | 1 | Bit0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x00X0 IDR0 | R<br>W | ID28 | ID27 | ID26 | ID25 | ID24 | ID23 | ID22 | ID21 |
| 0x00X1 IDR1 | R<br>W | ID20 | ID19 | ID18 | SRR (=1) | IDE (=1) | ID17 | ID16 | ID15 |
| 0x00X2 IDR2 | R<br>W | ID14 | ID13 | ID12 | ID11 | ID10 | ID9 | ID8 | ID7 |
| 0x00X3 IDR3 | R<br>W | ID6 | ID5 | ID4 | ID3 | ID2 | ID1 | ID0 | RTR |
| 0x00X4 DSR0 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00X5 DSR1 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00X6 DSR2 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00X7 DSR3 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00X8 DSR4 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00X9 DSR5 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00XA DSR6 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00XB DSR7 | R<br>W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 |
| 0x00XC DLR | R<br>W | | | | | DLC3 | DLC2 | DLC1 | DLC0 |

= Unused, always read 'x'

**Figure 10-23. Receive/Transmit Message Buffer — Extended Identifier Mapping**

Read: For transmit buffers, anytime when TXEx flag is set (see Section 10.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 10.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)"). For receive buffers, only when RXF flag is set (see Section 10.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)").

**Table 12-9. PWMCTL Field Descriptions**

| Field | Description |
|---|---|
| 6<br>CON45 | **Concatenate Channels 4 and 5**<br>0 Channels 4 and 5 are separate 8-bit PWMs.<br>1 Channels 4 and 5 are concatenated to create one 16-bit PWM channel. Channel 4 becomes the high-order byte and channel 5 becomes the low-order byte. Channel 5 output pin is used as the output for this 16-bit PWM (bit 5 of port PWMP). Channel 5 clock select control bit determines the clock source, channel 5 polarity bit determines the polarity, channel 5 enable bit enables the output and channel 5 center aligned enable bit determines the output mode. |
| 5<br>CON23 | **Concatenate Channels 2 and 3**<br>0 Channels 2 and 3 are separate 8-bit PWMs.<br>1 Channels 2 and 3 are concatenated to create one 16-bit PWM channel. Channel 2 becomes the high-order byte and channel 3 becomes the low-order byte. Channel 3 output pin is used as the output for this 16-bit PWM (bit 3 of port PWMP). Channel 3 clock select control bit determines the clock source, channel 3 polarity bit determines the polarity, channel 3 enable bit enables the output and channel 3 center aligned enable bit determines the output mode. |
| 4<br>CON01 | **Concatenate Channels 0 and 1**<br>0 Channels 0 and 1 are separate 8-bit PWMs.<br>1 Channels 0 and 1 are concatenated to create one 16-bit PWM channel. Channel 0 becomes the high-order byte and channel 1 becomes the low-order byte. Channel 1 output pin is used as the output for this 16-bit PWM (bit 1 of port PWMP). Channel 1 clock select control bit determines the clock source, channel 1 polarity bit determines the polarity, channel 1 enable bit enables the output and channel 1 center aligned enable bit determines the output mode. |
| 3<br>PSWAI | **PWM Stops in Wait Mode** — Enabling this bit allows for lower power consumption in wait mode by disabling the input clock to the prescaler.<br>0 Allow the clock to the prescaler to continue while in wait mode.<br>1 Stop the input clock to the prescaler whenever the MCU is in wait mode. |
| 2<br>PFRZ | **PWM Counters Stop in Freeze Mode** — In freeze mode, there is an option to disable the input clock to the prescaler by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode the input clock to the prescaler is disabled. This feature is useful during emulation as it allows the PWM function to be suspended. In this way, the counters of the PWM can be stopped while in freeze mode so that after normal program flow is continued, the counters are re-enabled to simulate real-time operations. Because the registers remain accessible in this mode, to re-enable the prescaler clock, either disable the PFRZ bit or exit freeze mode.<br>0 Allow PWM to continue while in freeze mode.<br>1 Disable PWM input clock to the prescaler whenever the part is in freeze mode. This is useful for emulation. |

Table 12-12 is used to summarize which channels are used to set the various control bits when in 16-bit mode.

**Table 12-12. 16-bit Concatenation Mode Summary**

| CONxx | PWMEx | PPOLx | PCLKx | CAEx | PWMx Output |
|---|---|---|---|---|---|
| CON45 | PWME5 | PPOL5 | PCLK5 | CAE5 | PWM5 |
| CON23 | PWME3 | PPOL3 | PCLK3 | CAE3 | PWM3 |
| CON01 | PWME1 | PPOL1 | PCLK1 | CAE1 | PWM1 |

### 12.4.2.8 PWM Boundary Cases

Table 12-13 summarizes the boundary conditions for the PWM regardless of the output mode (left aligned or center aligned) and 8-bit (normal) or 16-bit (concatenation):

**Table 12-13. PWM Boundary Cases**

| PWMDTYx | PWMPERx | PPOLx | PWMx Output |
|---|---|---|---|
| 0x0000 (indicates no duty) | >0x0000 | 1 | Always Low |
| 0x0000 (indicates no duty) | >0x0000 | 0 | Always High |
| XX | 0x0000[1] (indicates no period) | 1 | Always High |
| XX | 0x0000[1] (indicates no period) | 0 | Always Low |
| >= PWMPERx | XX | 1 | Always High |
| >= PWMPERx | XX | 0 | Always Low |

1. Counter = 0x0000 and does not count.

## 12.5 Resets

The reset state of each individual bit is listed within the register description section (see Section 12.3, "Memory Map and Register Definition," which details the registers and their bit-fields. All special functions or modes which are initialized during or just following reset are described within this section.

- The 8-bit up/down counter is configured as an up counter out of reset.
- All the channels are disabled and all the counters don't count.

## 12.6 Interrupts

The PWM8B6CV1 module has only one interrupt which is generated at the time of emergency shutdown, if the corresponding enable bit (PWMIE) is set. This bit is the enable for the interrupt. The interrupt flag PWMIF is set whenever the input level of the PWM5 channel changes while PWM5ENA=1 or when PWMENA is being asserted while the level at PWM5 is active.

A description of the registers involved and affected due to this interrupt is explained in Section 12.3.2.15, "PWM Shutdown Register (PWMSDN)."

# Chapter 18
# 32 Kbyte Flash Module (S12FTS32KV1)

## 18.1 Introduction

The FTS32K module implements a 32 Kbyte Flash (nonvolatile) memory. The Flash memory contains one array of 32 Kbytes organized as 512 rows of 64 bytes with an erase sector size of eight rows (512 bytes). The Flash array may be read as either bytes, aligned words, or misaligned words. Read access time is one bus cycle for byte and aligned word, and two bus cycles for misaligned words.

The Flash array is ideal for program and data storage for single-supply applications allowing for field reprogramming without requiring external voltage sources for program or erase. Program and erase functions are controlled by a command driven interface. The Flash module supports both mass erase and sector erase. An erased bit reads 1 and a programmed bit reads 0. The high voltage required to program and erase is generated internally. It is not possible to read from a Flash array while it is being erased or programmed.

**CAUTION**

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

### 18.1.1 Glossary

**Command Write Sequence** — A three-step MCU instruction sequence to program, erase, or erase verify the Flash array memory.

### 18.1.2 Features

- 32 Kbytes of Flash memory comprised of one 32 Kbyte array divided into 64 sectors of 512 bytes
- Automated program and erase algorithm
- Interrupts on Flash command completion and command buffer empty
- Fast sector erase and word program operation
- 2-stage command pipeline for faster multi-word program times
- Flexible protection scheme to prevent accidental program or erase
- Single power supply for Flash program and erase operations
- Security feature to prevent unauthorized access to the Flash array memory

#### 21.3.2.4 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash interrupts and gates the security backdoor key writes.

Module Base + 0x0003

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | CBEIE | CCIE | KEYACC | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

**Figure 21-7. Flash Configuration Register (FCNFG)**

CBEIE, CCIE, and KEYACC are readable and writable while remaining bits read 0 and are not writable. KEYACC is only writable if the KEYEN bit in the FSEC register is set to the enabled state (see Section 21.3.2.2).

**Table 21-7. FCNFG Field Descriptions**

| Field | Description |
|---|---|
| 7<br>CBEIE | **Command Buffer Empty Interrupt Enable** — The CBEIE bit enables the interrupts in case of an empty command buffer in the Flash module.<br>0 Command Buffer Empty interrupts disabled<br>1 An interrupt will be requested whenever the CBEIF flag is set (see Section 21.3.2.6) |
| 6<br>CCIE | **Command Complete Interrupt Enable** — The CCIE bit enables the interrupts in case of all commands being completed in the Flash module.<br>0 Command Complete interrupts disabled<br>1 An interrupt will be requested whenever the CCIF flag is set (see Section 21.3.2.6) |
| 5<br>KEYACC | **Enable Security Key Writing**.<br>0 Flash writes are interpreted as the start of a command write sequence<br>1 Writes to the Flash array are interpreted as a backdoor key while reads of the Flash array return invalid data |

#### 21.3.2.5 Flash Protection Register (FPROT)

The FPROT register defines which Flash sectors are protected against program or erase.

Module Base + 0x0004

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R<br>W | FPOPEN | NV6 | FPHDIS | FPHS1 | FPHS0 | FPLDIS | FPLS1 | FPLS0 |
| Reset | F | F | F | F | F | F | F | F |

**Figure 21-8. Flash Protection Register (FPROT)**

The FPROT register is readable in normal and special modes. FPOPEN can only be written from a 1 to a 0. FPLS[1:0] can be written anytime until FPLDIS is cleared. FPHS[1:0] can be written anytime until

Module Base + 0x000C

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

= Unimplemented or Reserved

**Figure 21-17. RESERVED3**

All bits read 0 and are not writable.

### 21.3.2.12 RESERVED4

This register is reserved for factory testing and is not accessible to the user.

Module Base + 0x000D

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

= Unimplemented or Reserved

**Figure 21-18. RESERVED4**

All bits read 0 and are not writable.

### 21.3.2.13 RESERVED5

This register is reserved for factory testing and is not accessible to the user.

Module Base + 0x000E

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

= Unimplemented or Reserved

**Figure 21-19. RESERVED5**

All bits read 0 and are not writable.

### 21.3.2.14 RESERVED6

This register is reserved for factory testing and is not accessible to the user.

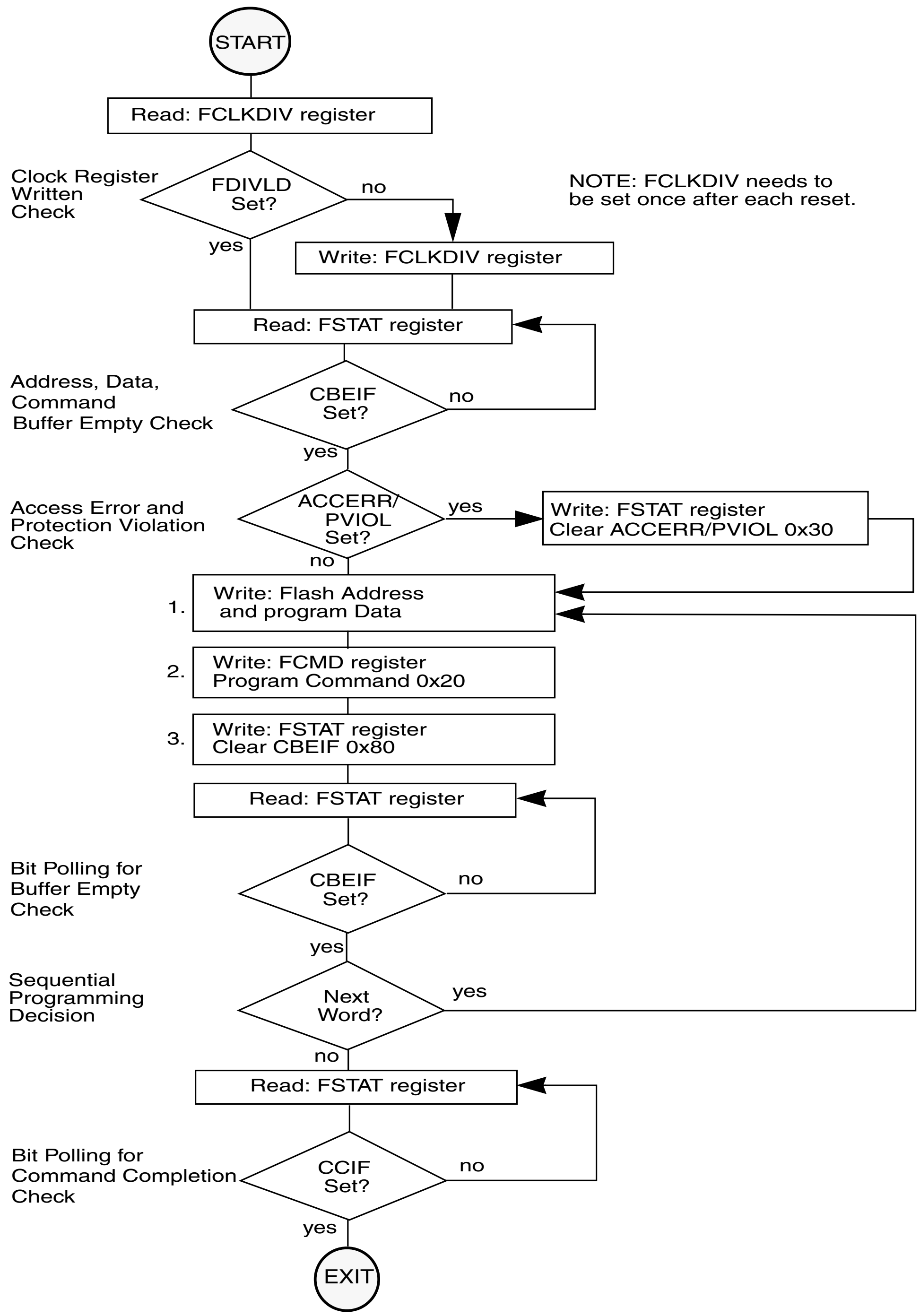


**Figure 21-23. Example Program Command Flow**