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	Obsolete
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	119
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 24x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s12xa256vag

BFINS Bit Field Insert BFINS

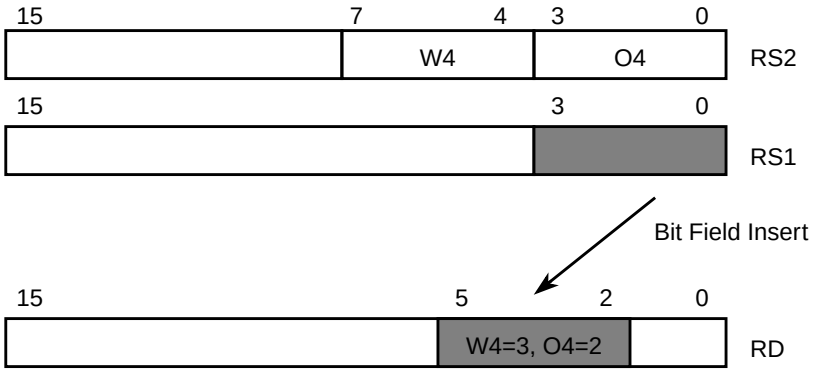
Operation

RS1[*w*:*0*] ⇒ RD[(*w*+*o*):o];

w = (RS2[7:4])

o = (RS2[3:0])

Extracts *w+1* bits from register RS1 starting at position 0 and writes them into register RD starting at position *o*. The remaining bits in RD are not affected. If (o+w) > 15 the upper bits are ignored. Using R0 as a RS1, this command can be used to clear bits.



CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: 0; cleared.

C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
BFINS RD, RS1, RS2	TRI	0	1	1	0	1	RD	RS1	RS2	1	1	P

BVC Branch if Overflow Cleared BVC

Operation

If V = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Overflow flag and branches if V = 0.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BVC REL9	REL9	0	0	1	0	1	1	0	REL9	PP/P

MOV — Move Register Content — MOV

Operation

RS ⇒ RD (translates to OR RD, R0, RS)

Copies the content of RS to RD.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

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

Code and CPU Cycles

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

Table 7-18. PACTL Field Descriptions (continued)

Field	Description
5 PAMOD	**Pulse Accumulator Mode** — This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1). 0 Event counter mode 1 Gated time accumulation mode
4 PEDGE	**Pulse Accumulator Edge Control** — This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1). Refer to Table 7-19. For PAMOD bit = 0 (event counter mode). 0 Falling edges on PT7 pin cause the count to be incremented 1 Rising edges on PT7 pin cause the count to be incremented For PAMOD bit = 1 (gated time accumulation mode). 0 PT7 input pin high enables bus clock divided by 64 to Pulse Accumulator and the trailing falling edge on PT7 sets the PAIF flag. 1 PT7 input pin low enables bus clock divided by 64 to Pulse Accumulator and the trailing rising edge on PT7 sets the PAIF flag. If the timer is not active (TEN = 0 in TSCR1), there is no divide-by-64 since the ÷64 clock is generated by the timer prescaler.
3:2 CLK[1:0]	**Clock Select Bits** — For the description of PACLK please refer to Figure 7-70. If the pulse accumulator is disabled (PAEN = 0), the prescaler clock from the timer is always used as an input clock to the timer counter. The change from one selected clock to the other happens immediately after these bits are written. Refer to Table 7-20.
2 PAOVI	**Pulse Accumulator A Overflow Interrupt Enable** 0 Interrupt inhibited 1 Interrupt requested if PAOVF is set
0 PAI	**Pulse Accumulator Input Interrupt Enable** 0 Interrupt inhibited 1 Interrupt requested if PAIF is set

Table 7-19. Pin Action

PAMOD	PEDGE	Pin Action
0	0	Falling edge
0	1	Rising edge
1	0	Divide by 64 clock enabled with pin high level
1	1	Divide by 64 clock enabled with pin low level

Table 7-20. Clock Selection

CLK1	CLK0	Clock Source
0	0	Use timer prescaler clock as timer counter clock
0	1	Use PACLK as input to timer counter clock
1	0	Use PACLK/256 as timer counter clock frequency
1	1	Use PACLK/65536 as timer counter clock frequency

7.3.2.24 Input Control System Control Register (ICSYS)

	7	6	5	4	3	2	1	0
R W	SH37	SH26	SH15	SH04	TFMOD	PACMX	BUFEN	LATQ
Reset	0	0	0	0	0	0	0	0

Figure 7-46. Input Control System Register (ICSYS)

Read: Anytime

Write: Once in normal modes

All bits reset to zero.

Table 7-30. ICSYS Field Descriptions

Field	Description
7:4 SHxy	**Share Input action of Input Capture Channels x and y** 0 Normal operation 1 The channel input ‘x’ causes the same action on the channel ‘y’. The port pin ‘x’ and the corresponding edge detector is used to be active on the channel ‘y’.
3 TFMOD	**Timer Flag Setting Mode** — Use of the TFMOD bit in conjunction with the use of the ICOVW register allows a timer interrupt to be generated after capturing two values in the capture and holding registers instead of generating an interrupt for every capture. By setting TFMOD in queue mode, when NOVWx bit is set and the corresponding capture and holding registers are emptied, an input capture event will first update the related input capture register with the main timer contents. At the next event, the TCx data is transferred to the TCxH register, the TCx is updated and the CxF interrupt flag is set. In all other input capture cases the interrupt flag is set by a valid external event on PTx. 0 The timer flags C3F–C0F in TFLG1 are set when a valid input capture transition on the corresponding port pin occurs. 1 If in queue mode (BUFEN = 1 and LATQ = 0), the timer flags C3F–C0F in TFLG1 are set only when a latch on the corresponding holding register occurs. If the queue mode is not engaged, the timer flags C3F–C0F are set the same way as for TFMOD = 0.
2 PACMX	**8-Bit Pulse Accumulators Maximum Count** 0 Normal operation. When the 8-bit pulse accumulator has reached the value 0x00FF, with the next active edge, it will be incremented to 0x0000. 1 When the 8-bit pulse accumulator has reached the value 0x00FF, it will not be incremented further. The value 0x00FF indicates a count of 255 or more.
1 BUFFEN	**IC Buffer Enable** 0 Input capture and pulse accumulator holding registers are disabled. 1 Input capture and pulse accumulator holding registers are enabled. The latching mode is defined by LATQ control bit.

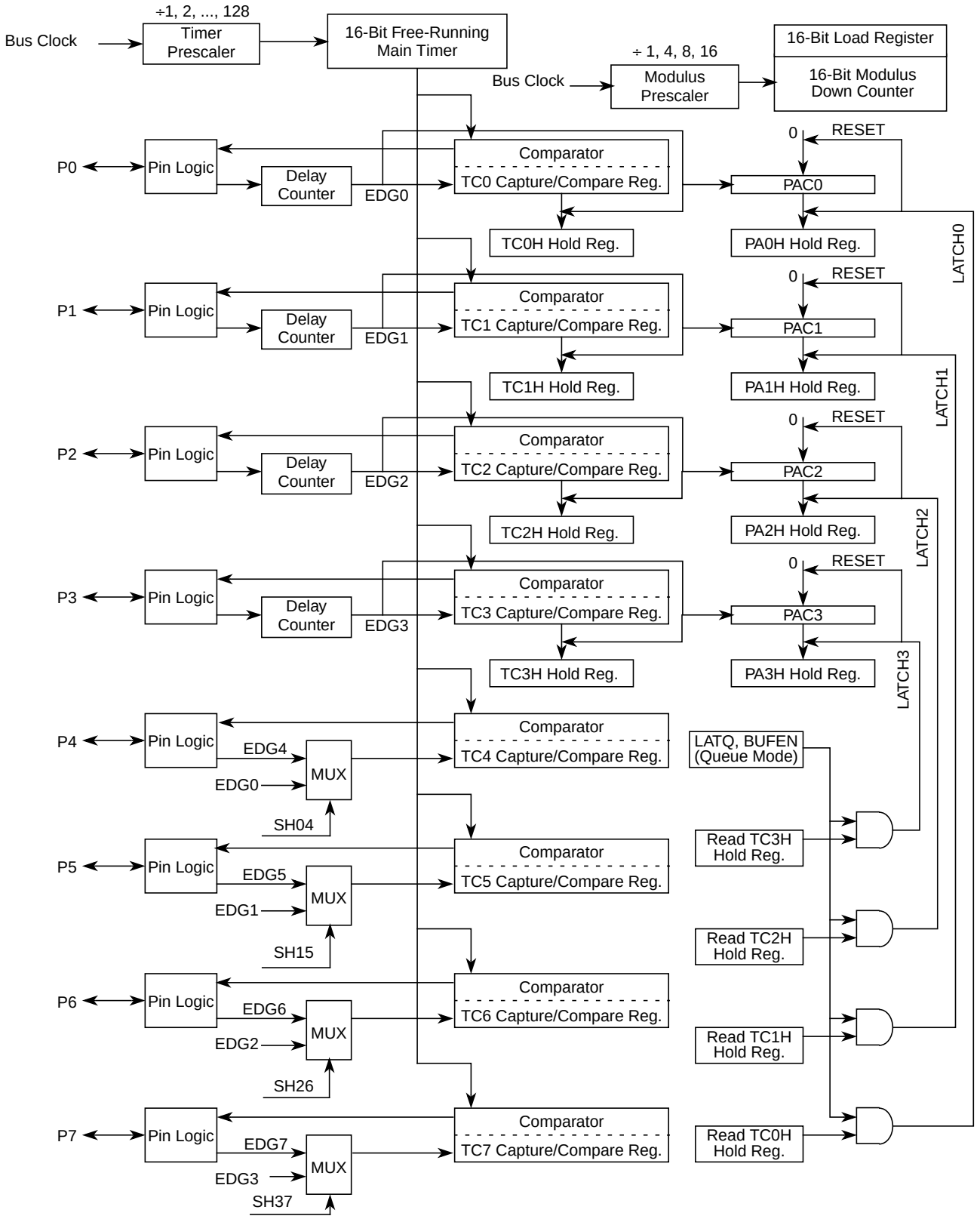


Figure 7-67. Detailed Timer Block Diagram in Queue Mode when PRNT = 0

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.

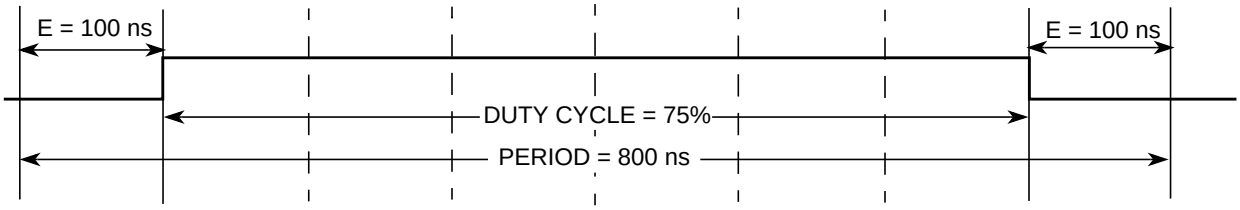


Figure 8-23. PWM Center Aligned Output Example Waveform

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.

If the MSCAN is configured for user request (BORM set in Section 10.3.2.2, "MSCAN Control Register 1 (CANCTL1)"), the recovery from bus-off starts after both independent events have become true:

- 128 occurrences of 11 consecutive recessive bits on the CAN bus have been monitored
- BOHOLD in Section 10.3.2.14, "MSCAN Miscellaneous Register (CANMISC) has been cleared by the user

These two events may occur in any order.

16.3.1.3 Interrupt Request Configuration Address Register (INT_CFADDR)

Address: 0x0127

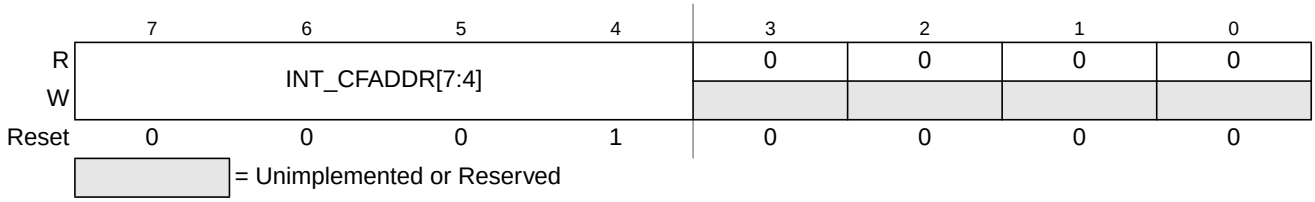


Figure 16-5. Interrupt Configuration Address Register (INT_CFADDR)

Read: Anytime

Write: Anytime

Table 16-5. INT_CFADDR Field Descriptions

Field	Description
7–4 INT_CFADDR[7:4]	**Interrupt Request Configuration Data Register Select Bits** — These bits determine which of the 128 configuration data registers are accessible in the 8 register window at INT_CFDATA0–7. The hexadecimal value written to this register corresponds to the upper nibble of the lower byte of the interrupt vector, i.e., writing 0xE0 to this register selects the configuration data register block for the 8 interrupt vector requests starting with vector (vector base + 0x00E0) to be accessible as INT_CFDATA0–7. **Note:** Writing all 0s selects non-existing configuration registers. In this case write accesses to INT_CFDATA0–7 will be ignored and read accesses will return all 0.

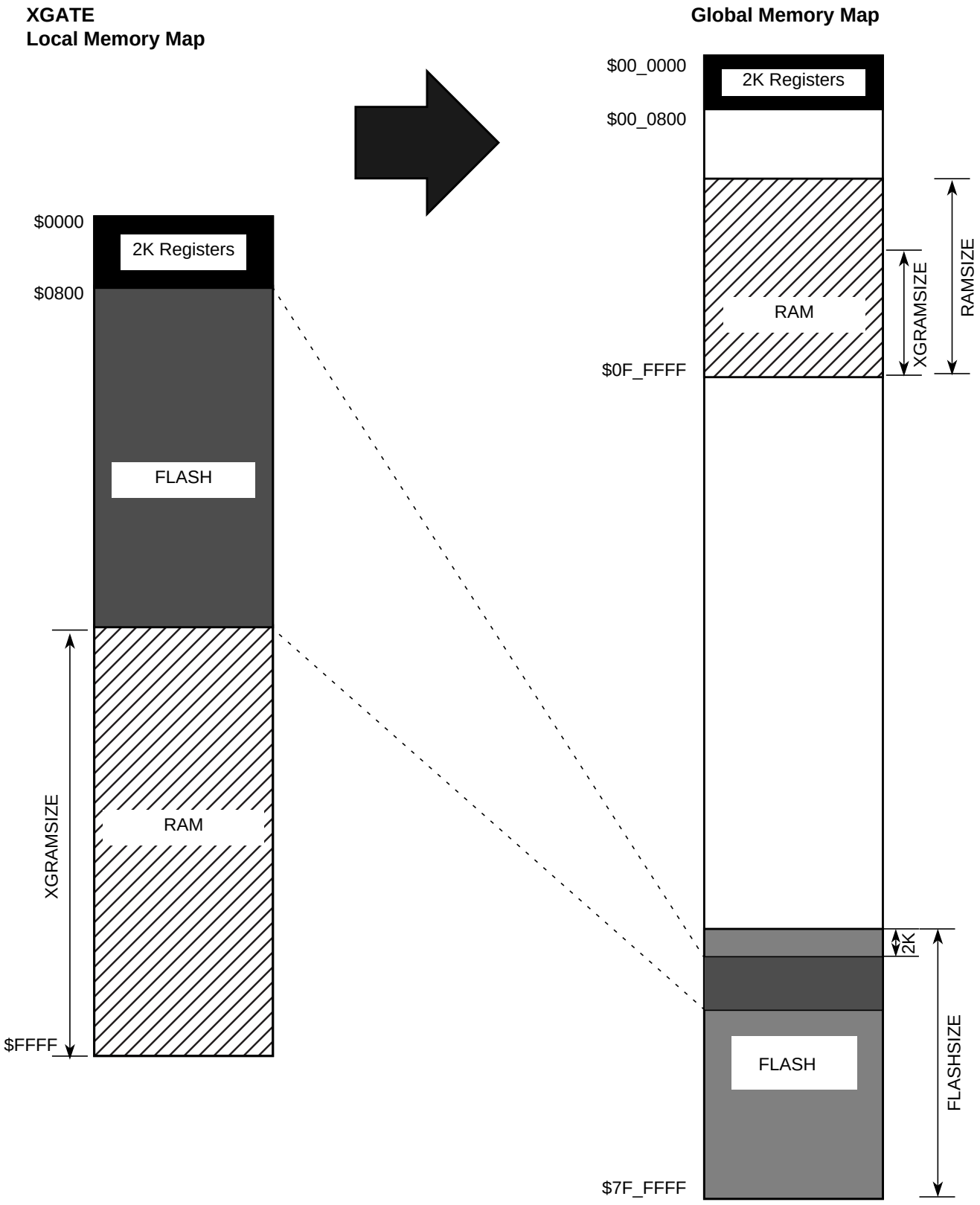


Figure 17-24. Local to Global Address Mapping (XGATE)

- Simultaneous accesses to different resources[1] (internal, external, and peripherals) (see)
- 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

18.1.3 S12X Memory Mapping

The S12X architecture implements a number of memory mapping schemes including

- a CPU 8 MByte global map, defined using a global page (GPAGE) register and dedicated 23-bit address load/store instructions.
- a BDM 8 MByte global map, defined using a global page (BDMGPR) register and dedicated 23-bit address load/store instructions.
- a (CPU or BDM) 64 KByte local map, defined using specific resource page (RPAGE, EPAGE and PPAGE) registers and the default instruction set. The 64 KBytes visible at any instant can be considered as the local map accessed by the 16-bit (CPU or BDM) address.
- The XGATE 64 Kbyte local map.

The MMC module performs translation of the different memory mapping schemes to the specific global (physical) memory implementation.

18.1.4 Modes of Operation

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

18.1.4.1 Power Saving Modes

- Run mode

  MMC is functional during normal run mode.
- Wait mode

  MMC is functional during wait mode.
- Stop mode

  MMC is inactive during stop mode.

1. Resources are also called targets.

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0029	DBGXAH	R	0	Bit 22	21	20	19	18	17	Bit 16
		W								
0x002A	DBGXAM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002B	DBGXAL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002C	DBGXDH	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002D	DBGXDL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002E	DBGXDHM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002F	DBGXDLM	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

= Unimplemented or Reserved

Figure 19-2. DBG Register Summary (continued)

19.3.1.1 Debug Control Register 1 (DBGC1)

0x0020

	7	6	5	4	3	2	1	0
R	ARM	0	XGSBPE	BDM	DBGBRK		COMRV	
W		TRIG						
Reset	0	0	0	0	0	0	0	0

Figure 19-3. Debug Control Register (DBGC1)

Read: Anytime

Write: Bits 7,1,0 anytime, Bit 6 can be written anytime but always reads back as 0.
Bits 5:2 anytime DBG is not armed.

NOTE

When disarming the DBG by clearing ARM with software, the contents of bits[5:2] are not affected by the write, since up until the write operation, ARM=1 preventing these bits from being written. These bits must be cleared using a second write if required.

19.3.1.11.2 Debug Comparator Address High Register (DBGXAH)

0x0029

	7	6	5	4	3	2	1	0
R	0	Bit 22	21	20	19	18	17	Bit 16
W								
Reset	0	0	0	0	0	0	0	0

Unimplemented or Reserved

Figure 19-15. Debug Comparator Address High Register (DBGXAH)

Read: Anytime

Write: Anytime when DBG not armed.

Table 19-29. DBGXAH Field Descriptions

Field	Description
6–0 Bits [22:16]	**Comparator Address High Compare Bits** — The comparator address high compare bits control whether the selected comparator will compare the address bus bits [22:16] to a logic 1 or logic 0. This register byte is ignored for XGATE compares. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

19.3.1.11.3 Debug Comparator Address Mid Register (DBGXAM)

0x002A

	7	6	5	4	3	2	1	0
R	Bit 15	14	13	12	11	10	9	Bit 8
W								
Reset	0	0	0	0	0	0	0	0

Figure 19-16. Debug Comparator Address Mid Register (DBGXAM)

Read: Anytime

Write: Anytime when DBG not armed.

Table 19-30. DBGXAM Field Descriptions

Field	Description
7–0 Bits [15:8]	**Comparator Address Mid Compare Bits** — The comparator address mid compare bits control whether the selected comparator will compare the address bus bits [15:8] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0 1 Compare corresponding address bit to a logic 1

Table 20-42. CXINF Field Descriptions (continued)

Field	Description
5 CRW	**Read Write Indicator** — This bit indicates if the corresponding stored address corresponds to a read or write access. This bit only contains valid information when tracing S12XCPU activity in Detail Mode. 0 Write Access 1 Read Access
4 COCF	**S12XCPU Opcode Fetch Indicator** — This bit indicates if the stored address corresponds to an opcode fetch cycle. This bit only contains valid information when tracing the XGATE accesses in Detail Mode. 0 Stored information does not correspond to opcode fetch cycle 1 Stored information corresponds to opcode fetch cycle
3 XACK	**XGATE Access Indicator** — This bit indicates if the stored XGATE address corresponds to a free cycle. This bit only contains valid information when tracing the S12XCPU accesses in Detail Mode. 0 Stored information corresponds to free cycle 1 Stored information does not correspond to free cycle
2 XSZ	**Access Type Indicator** — This bit indicates if the access was a byte or word size access. This bit only contains valid information when tracing XGATE activity in Detail Mode. 0 Word Access 1 Byte Access
1 XRW	**Read Write Indicator** — This bit indicates if the corresponding stored address corresponds to a read or write access. This bit only contains valid information when tracing XGATE activity in Detail Mode. 0 Write Access 1 Read Access
0 XOCF	**XGATE Opcode Fetch Indicator** — This bit indicates if the stored address corresponds to an opcode fetch cycle.This bit only contains valid information when tracing the S12XCPU accesses in Detail Mode. 0 Stored information does not correspond to opcode fetch cycle 1 Stored information corresponds to opcode fetch cycle

20.4.5.4 Reading Data from Trace Buffer

The data stored in the Trace Buffer can be read using either the background debug module (BDM) module or the S12XCPU provided the S12XDBG module is not armed, is configured for tracing (at least one TSOURCE bit is set) and the system not secured. When the ARM bit is written to 1 the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by a single aligned word write to DBGTB when the module is disarmed. Multiple writes to the DBGTB are not allowed since they increment the pointer.

The Trace Buffer can only be read through the DBGTB register using aligned word reads, any byte or misaligned reads return 0 and do not cause the trace buffer pointer to increment to the next trace buffer address. The Trace Buffer data is read out first-in first-out. By reading CNT in DBGCNT the number of valid 64-bit lines can be determined. DBGCNT will not decrement as data is read.

Whilst reading an internal pointer is used to determine the next line to be read. After a tracing session, the pointer points to the oldest data entry, thus if no overflow has occurred, the pointer points to line0, otherwise it points to the line with the oldest entry. The pointer is initialized by each aligned write to DBGTBH to point to the oldest data again. This enables an interrupted trace buffer read sequence to be easily restarted from the oldest data entry.

22.4.1.2 Input Register

This is a read-only register and always returns the buffered state of the pin (Figure 22-76).

22.4.1.3 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 22-76).

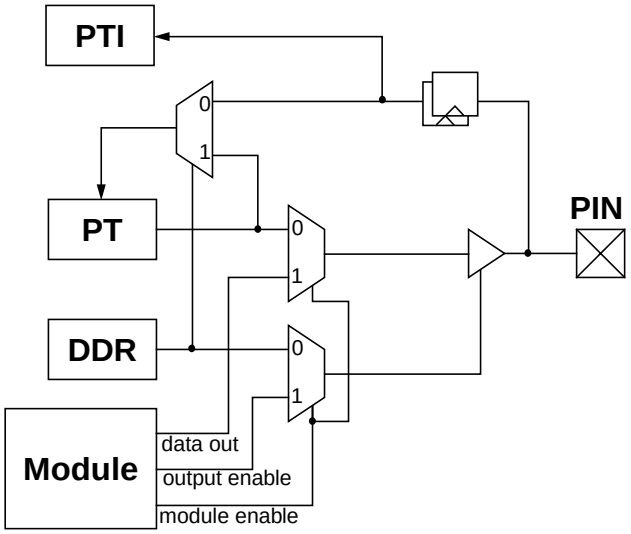


Figure 22-76. Illustration of I/O Pin Functionality

22.4.1.4 Reduced Drive Register

If the pin is used as an output this register allows the configuration of the drive strength.

22.4.1.5 Pull Device Enable Register

This register turns on a pull-up or pull-down device. It becomes active only if the pin is used as an input or as a wired-OR output.

22.4.1.6 Polarity Select Register

This register selects either a pull-up or pull-down device if enabled. It becomes active only if the pin is used as an input. A pull-up device can be activated if the pin is used as a wired-OR output. If the pin is used as an interrupt input this register selects the active interrupt edge.

22.4.1.7 Wired-OR Mode Register

If the pin is used as an output this register turns off the active high drive. This allows wired-OR type connections of outputs.

24.0.5.8 S12X_EBI Ports Reduced Drive Register (RDRIV)

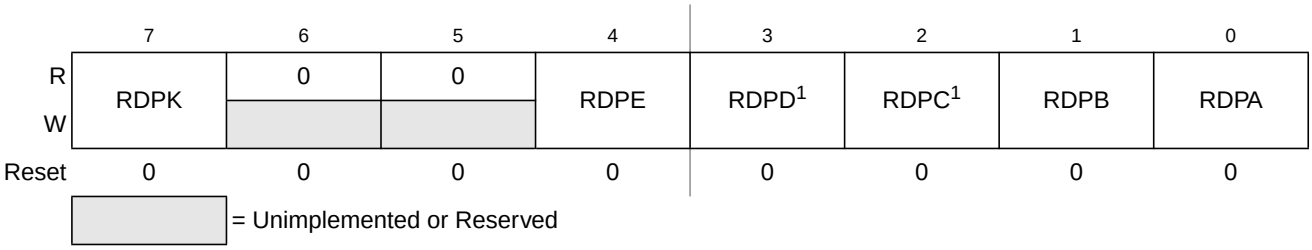


Figure 24-10. S12X_EBI Ports Reduced Drive Register (RDRIV)

1. Register implemented, function disabled: Written values can be read back.

Read: Anytime.

Write: Anytime.

This register is used to select reduced drive for the pins associated with ports A, B, E, and K. If enabled, the pins drive at about 1/6 of the full drive strength. The reduced drive function is independent of which function is being used on a particular pin.

The reduced drive functionality does not take effect on the pins in emulation modes.

Table 24-11. RDRIV Field Descriptions

Field	Description
7 RDPK	**Reduced Drive of Port K** 0 All port K output pins have full drive enabled. 1 All port K output pins have reduced drive enabled.
4 RDPE	**Reduced Drive of Port E** 0 All port E output pins have full drive enabled. 1 All port E output pins have reduced drive enabled.
1 RDPB	**Reduced Drive of Port B** 0 All port B output pins have full drive enabled. 1 All port B output pins have reduced drive enabled.
0 RDPA	**Reduced Drive of Ports A** 0 All Port A output pins have full drive enabled. 1 All port A output pins have reduced drive enabled.

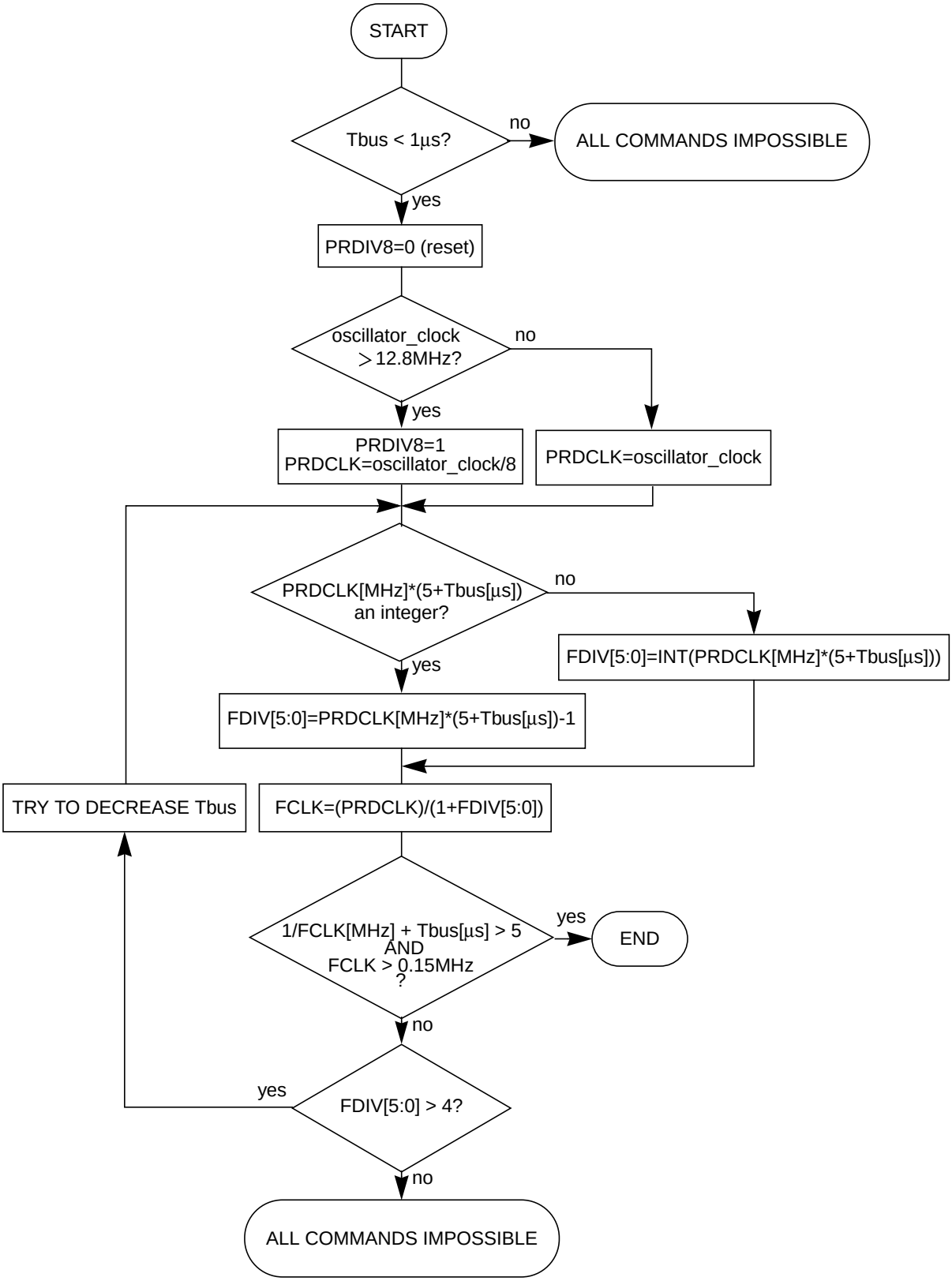


Figure 28-24. Determination Procedure for PRDIV8 and FDIV Bits

Table 29-8. FCNFG Field Descriptions

Field	Description
7 CBEIE	**Command Buffer Empty Interrupt Enable** — The CBEIE bit enables an interrupt in case of an empty command buffer in the Flash module. 0 Command buffer empty interrupt disabled. 1 An interrupt will be requested whenever the CBEIF flag (see Section 29.3.2.6, “Flash Status Register (FSTAT)”) is set.
6 CCIE	**Command Complete Interrupt Enable** — The CCIE bit enables an interrupt in case all commands have been completed in the Flash module. 0 Command complete interrupt disabled. 1 An interrupt will be requested whenever the CCIF flag (see Section 29.3.2.6, “Flash Status Register (FSTAT)”) is set.
5 KEYACC	**Enable Security Key Writing** 0 Flash writes are interpreted as the start of a command write sequence. 1 Writes to Flash array are interpreted as keys to open the backdoor. Reads of the Flash array return invalid data.

29.3.2.5 Flash Protection Register (FPROT)

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

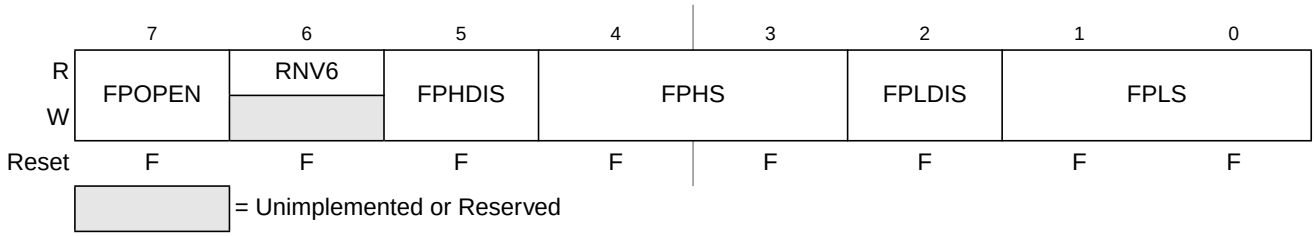


Figure 29-8. Flash Protection Register (FPROT)

All bits in the FPROT register are readable and writable with restrictions (see Section 29.3.2.5.1, “Flash Protection Restrictions”) except for RNV[6] which is only readable.

During the reset sequence, the FPROT register is loaded from the Flash Configuration Field at global address 0x7F_FF0D. To change the Flash protection that will be loaded during the reset sequence, the upper sector of the Flash memory must be unprotected, then the Flash Protect/Security byte located as described in Table 29-1 must be reprogrammed.

Trying to alter data in any protected area in the Flash memory will result in a protection violation error and the PVIOL flag will be set in the FSTAT register. The mass erase of a Flash block is not possible if any of the Flash sectors contained in the Flash block are protected.

between the written data and the backdoor key data stored in the Flash memory. If all four words of data are written to the correct addresses in the correct order and the data matches the backdoor keys stored in the Flash memory, the MCU will be unsecured. The data must be written to the backdoor keys sequentially starting with 0x7F_FF00–1 and ending with 0x7F_FF06–7. 0x0000 and 0xFFFF are not permitted as backdoor keys. While the KEYACC bit is set, reads of the Flash memory will return invalid data.

The user code stored in the Flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state (see Section 29.3.2.2, “Flash Security Register (FSEC)”), the MCU can be unsecured by the backdoor key access sequence described below:

1. Set the KEYACC bit in the Flash Configuration Register (FCNFG).
2. Write the correct four 16-bit words to Flash addresses 0xFF00–0xFF07 sequentially starting with 0x7F_FF00.
3. Clear the KEYACC bit. Depending on the user code used to write the backdoor keys, a wait cycle (NOP) may be required before clearing the KEYACC bit.
4. If all four 16-bit words match the backdoor keys stored in Flash addresses 0x7F_FF00–0x7F_FF07, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 1:0.

The backdoor key access sequence is monitored by an internal security state machine. An illegal operation during the backdoor key access sequence will cause the security state machine to lock, leaving the MCU in the secured state. A reset of the MCU will cause the security state machine to exit the lock state and allow a new backdoor key access sequence to be attempted. The following operations during the backdoor key access sequence will lock the security state machine:

1. If any of the four 16-bit words does not match the backdoor keys programmed in the Flash array.
2. If the four 16-bit words are written in the wrong sequence.
3. If more than four 16-bit words are written.
4. If any of the four 16-bit words written are 0x0000 or 0xFFFF.
5. If the KEYACC bit does not remain set while the four 16-bit words are written.
6. If any two of the four 16-bit words are written on successive MCU clock cycles.

After the backdoor keys have been correctly matched, the MCU will be unsecured. Once the MCU is unsecured, the Flash security byte can be programmed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0x7F_FF00–0x7F_FF07 in the Flash Configuration Field.

The security as defined in the Flash security byte (0x7F_FF0F) is not changed by using the backdoor key access sequence to unsecure. The backdoor keys stored in addresses 0x7F_FF00–0x7F_FF07 are unaffected by the backdoor key access sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte (0x7F_FF0F). The backdoor key access sequence has no effect on the program and erase protections defined in the Flash protection register.

It is not possible to unsecure the MCU in special single chip mode by using the backdoor key access sequence in background debug mode (BDM).

0x0100–0x010F Flash Control Register (FTX512K4) Map (continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x010C	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x010D	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x010E	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x010F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0110–0x011B EEPROM Control Register (EETX4K) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0110	ECLKDIV	R	EDIVLD	PRDIV8	EDIV5	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
		W								
0x0111	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0112	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0113	ECNFG	R	CBEIE	CCIE	0	0	0	0	0	0
		W								
0x0114	EPROT	R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
		W								
0x0115	ESTAT	R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
		W								
0x0116	ECMD	R	0	CMDB[6:0]						
		W								
0x0117	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0118	EADDRHI	R	0	0	0	0	0	EABHI		
		W								
0x0119	EADDRLO	R	EABLO							
		W								
0x011A	EDATAHI	R	EDHI							
		W								
0x011B	EDATALO	R	EDLO							
		W								