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	HCS12
Core Size	16-Bit
Speed	25MHz
Connectivity	CANbus, EBI/EMI, I²C, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	59
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2.35V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-QFP
Supplier Device Package	80-QFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s12kg128cfue

  - Low power mode capability
  - Low Voltage Reset (LVR) and Low Voltage Interrupt (LVI)
- 20 key wake up inputs
  - Rising or falling edge triggered interrupt capability
  - Digital filter to prevent short pulses from triggering interrupts
  - Programmable pull ups and pull downs
- Operating frequency for ambient temperatures (T_A -40°C to 125°C)
  - 50MHz equivalent to 25MHz Bus Speed
- 112-Pin LQFP or 80-Pin QFP package
  - I/O lines with 3.3V/5V input and drive capability
  - 3.3V/5V A/D converter inputs

1.1.2 Modes of Operation

- Normal modes
  - Normal Single-Chip Mode
  - Normal Expanded Wide Mode
  - Normal Expanded Narrow Mode
  - Emulation Expanded Wide Mode
  - Emulation Expanded Narrow Mode
- Special Operating Modes
  - Special Single-Chip Mode with active Background Debug Mode
  - Special Test Mode (Freescale use only)
  - Special Peripheral Mode (Freescale use only)
- Each of the above modes of operation can be configured for three Low power submodes
  - Stop Mode
  - Pseudo Stop Mode
  - Wait Mode
- Secure operation, preventing the unauthorized read and write of the memory contents.

HCS12 core PPAGE register is used to map the logical middle page ranging from address 0x8000 to 0xBFFF to any physical 16 Kbyte page in the Flash memory. By placing 0x3E or 0x3F in the HCS12 Core PPAGE register, the associated 16 Kbyte pages appear twice in the MCU memory map.

The FPROT register, described in Section 2.3.2.5, “Flash Protection Register (FPROT)”, can be set to globally protect a Flash block. However, three separate memory regions, one growing upward from the first address in the next-to-last page in the Flash block (called the lower region), one growing downward from the last address in the last page in the Flash block (called the higher region), and the remaining addresses in the Flash block, can be activated for protection. The Flash locations of these protectable regions are shown in Table 2-3. The higher address region is mainly targeted to hold the boot loader code because it covers the vector space. The lower address region can be used for EEPROM emulation in an MCU without an EEPROM module because it can remain unprotected while the remaining addresses are protected from program or erase.

Security information that allows the MCU to restrict access to the Flash module is stored in the Flash configuration field, described in Table 2-2.

Table 2-2. Flash Configuration Field

Unpaged Flash Address	Paged Flash Address (PPAGE 0x3F)	Size (bytes)	Description
0xFF00 - 0xFF07	0xBF00-0xBF07	8	Backdoor Comparison Key Refer to Section Section 2.6.1, “Unsecuring the MCU using Backdoor Key Access”
0xFF08 - 0xFF0C	0xBF08-0xBF0C	5	Reserved
0xFF0D	0xBF0D	1	Flash Protection byte Refer to Section 2.3.2.5, “Flash Protection Register (FPROT)”
0xFF0E	0xBF0E	1	Reserved
0xFF0F	0xBF0F	1	Flash Security byte Refer to Section 2.3.2.2, “Flash Security Register (FSEC)”

If we define:

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

Then, FCLKDIV register bits PRDIV8 and FDIV[5:0] are to be set as described in Figure 2-23.

For example, if the oscillator clock frequency is 950 kHz and the bus clock frequency is 10 MHz, FCLKDIV bits FDIV[5:0] must be set to 4 (000100) and bit PRDIV8 set to 0. The resulting FCLK frequency is then 190 kHz. As a result, the Flash program and erase algorithm timings are increased over the optimum target by:

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

CAUTION

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

If the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. Flash commands will not be executed if the FCLKDIV register has not been written to.

Table 3-6. EEPROM Address Range Protection

EP[2:0]	Protected Address Range	Protected Size
000	0x07C0-0x07FF	64 bytes
001	0x0780-0x07FF	128 bytes
010	0x0740-0x07FF	192 bytes
011	0x0700-0x07FF	256 bytes
100	0x06C0-0x07FF	320 bytes
101	0x0680-0x07FF	384 bytes
110	0x0640-0x07FF	448 bytes
111	0x0600-0x07FF	512 bytes

3.3.2.6 EEPROM Status Register (ESTAT)

The ESTAT register defines the EEPROM state machine command status and EEPROM array access, protection and erase verify status.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-9. EEPROM Status Register (ESTAT - Normal Mode)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	FAIL	DONE
W								
Reset	1	1	0	0	0	0	0	1

= Unimplemented or Reserved

Figure 3-10. EEPROM Status Register (ESTAT - Special Mode)

CBEIF, PVIOL, and ACCERR bits are readable and writable, CCIF and BLANK bits are readable but 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 when starting a command write sequence. DONE is readable but not writable in special mode.

4.3.3.7 Port M Wired-OR Mode Register (WOMM)

Module Base + 0x0016

	7	6	5	4	3	2	1	0
R W	WOMM7	WOMM6	WOMM5	WOMM4	WOMM3	WOMM2	WOMM1	WOMM0
Reset	0	0	0	0	0	0	0	0

Figure 4-21. Port M Wired-OR Mode Register (WOMM)

Read: Anytime. Write: Anytime.

This register configures the output pins as wired-OR. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven. It applies also to the CAN outputs and allows a multipoint connection of several serial modules. This bit has no influence on pins used as inputs.

Table 4-16. WOMM Field Descriptions

Field	Description
7–0 WOMM[7:0]	**Wired-OR Mode Port M** 0 Output buffers operate as push-pull outputs. 1 Output buffers operate as open-drain outputs.

4.3.3.8 Module Routing Register (MODRR)

Module Base + 0x0017

	7	6	5	4	3	2	1	0
R	0	MODRR6	MODRR5	MODRR4	MODRR3	MODRR2	MODRR1	MODRR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-22. Module Routing Register (MODRR)

Read: Anytime. Write: Anytime.

This register configures the re-routing of CAN0, CAN4, SPI0, SPI1 and SPI2 on defined port pins.

Table 4-17. MODRR Field Descriptions

Field	Description
6 MODRR6	**SPI2 Routing Bit** — See Table 4-22.
5 MODRR5	**SPI1 Routing Bit** — See Table 4-21.
4 MODRR4	**SPI0 Routing Bit** — See Table 4-20.

4.3.6.7 Port J Interrupt Enable Register (PIEJ)

Module Base + 0x002E

	7	6	5	4	3	2	1	0
R	PIEJ7	PIEJ6	0	0	0	0	PIEJ1	PIEJ0
W								
Reset	0	0	—	—	—	—	0	0

= Unimplemented or Reserved

Figure 4-45. Port J Interrupt Enable Register (PIEJ)

Read: Anytime. Write: Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port J.

Table 4-40. PIEJ Field Descriptions

Field	Description
7, 6, 1, 0 PIEJ[7:6] PIEJ[1:0]	**Interrupt Enable Port J** 0 Interrupt is disabled (interrupt flag masked). 1 Interrupt is enabled.

4.3.6.8 Port J Interrupt Flag Register (PIFJ)

Module Base + 0x002F

	7	6	5	4	3	2	1	0
R	PIFJ7	PIFJ6	0	0	0	0	PIFJ1	PIFJ0
W								
Reset	0	0	—	—	—	—	0	0

= Unimplemented or Reserved

Figure 4-46. Port J Interrupt Flag Register (PIFJ)

Read: Anytime. Write: Anytime.

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

Table 4-41. PIFJ Field Descriptions

Field	Description
7, 6, 1, 0 PIFJ[7:6] PIFJ[1:0]	**Interrupt Flags Port J** 0 No active edge pending. Writing a “0” has no effect. 1 Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set). Writing a “1” clears the associated flag.

4.4 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. Table 4-42 summarizes the priority in case of multiple enabled modules trying to control a shared port.

Table 4-42. Summary of Functional Priority

Port	Priority[1]
T	TIMER > GPIO
S	SCI0, SCI1, SPI0 > GPIO
M	CAN0 > GPIO CAN0 (routed) > SPI0 (routed) > GPIO CAN0 (routed) > CAN4 (routed) > SPI0 (routed) > GPIO CAN4 (routed) > GPIO
P	PWM > SPI1, SPI2 > GPIO
H	SPI1, SPI2 > GPIO
J	CAN4 > IIC > GPIO

1 Highest priority >... > lowest priority

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:

A selected pull-up resistor does not become active while the port is used as a push-pull output.

4.4.1 I/O Register

The I/O Register holds the value driven out to the pin if the port is used as a general-purpose I/O. Writing to the I/O Register only has an effect on the pin if the port is used as general-purpose output.

When reading the I/O Register, the value of each pin is returned if the corresponding Data Direction Register bit is set to 0 (pin configured as input). If the data direction register bits is set to 1, the content of the I/O Register bit is returned. This is independent of any other configuration (Figure 4-47).

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on the I/O Register when changing the data direction register.

4.4.2 Input Register

The Input Register is a read-only register and generally returns the value of the pin (Figure 4-47). It can be used to detect overload or short circuit conditions.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on the Input Register when changing the Data Direction Register.

5.3.2.2 CRG Reference Divider Register (REFDV)

The REFDV register provides a finer granularity for the PLL multiplier steps. The count in the reference divider divides OSCCLK frequency by REFDV + 1.

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-5. CRG Reference Divider Register (REFDV)

Read: anytime

Write: anytime except when PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit and the track detector bit.

5.3.2.3 Reserved Register (CTFLG)

This register is reserved for factory testing of the CRGV4 module and is not available in normal modes.

Module Base + 0x0002

	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 5-6. CRG Reserved Register (CTFLG)

Read: always reads 0x0000 in normal modes

Write: unimplemented in normal modes

NOTE

Writing to this register when in special mode can alter the CRGV4 functionality.

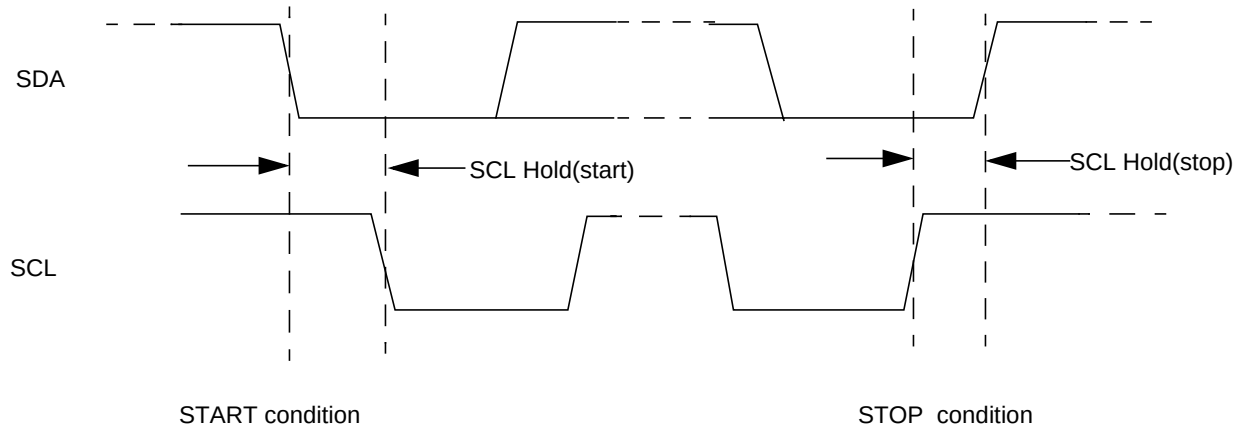

Figure 8-5. SCL Divider and SDA Hold

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)}

The SDA hold delay is equal to the CPU clock period multiplied by the SDA Hold value shown in Table 8-5. The equation used to generate the SDA Hold value from the IBFD bits is:

SDA Hold = MUL x {scl2tap + [(SDA_Tap - 1) x tap2tap] + 3}

The equation for SCL Hold values to generate the start and stop conditions from the IBFD bits is:

SCL Hold(start) = MUL x [scl2start + (SCL_Tap - 1) x tap2tap]

SCL Hold(stop) = MUL x [scl2stop + (SCL_Tap - 1) x tap2tap]

Table 8-5. IIC Divider and Hold Values (Sheet 1 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
MUL=1				
00	20	7	6	11
01	22	7	7	12
02	24	8	8	13
03	26	8	9	14
04	28	9	10	15
05	30	9	11	16
06	34	10	13	18
07	40	10	16	21
08	28	7	10	15
09	32	7	12	17
0A	36	9	14	19
0B	40	9	16	21
0C	44	11	18	23
0D	48	11	20	25
0E	56	13	24	29
0F	68	13	30	35
10	48	9	18	25

Table 8-5. IIC Divider and Hold Values (Sheet 2 of 5)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281

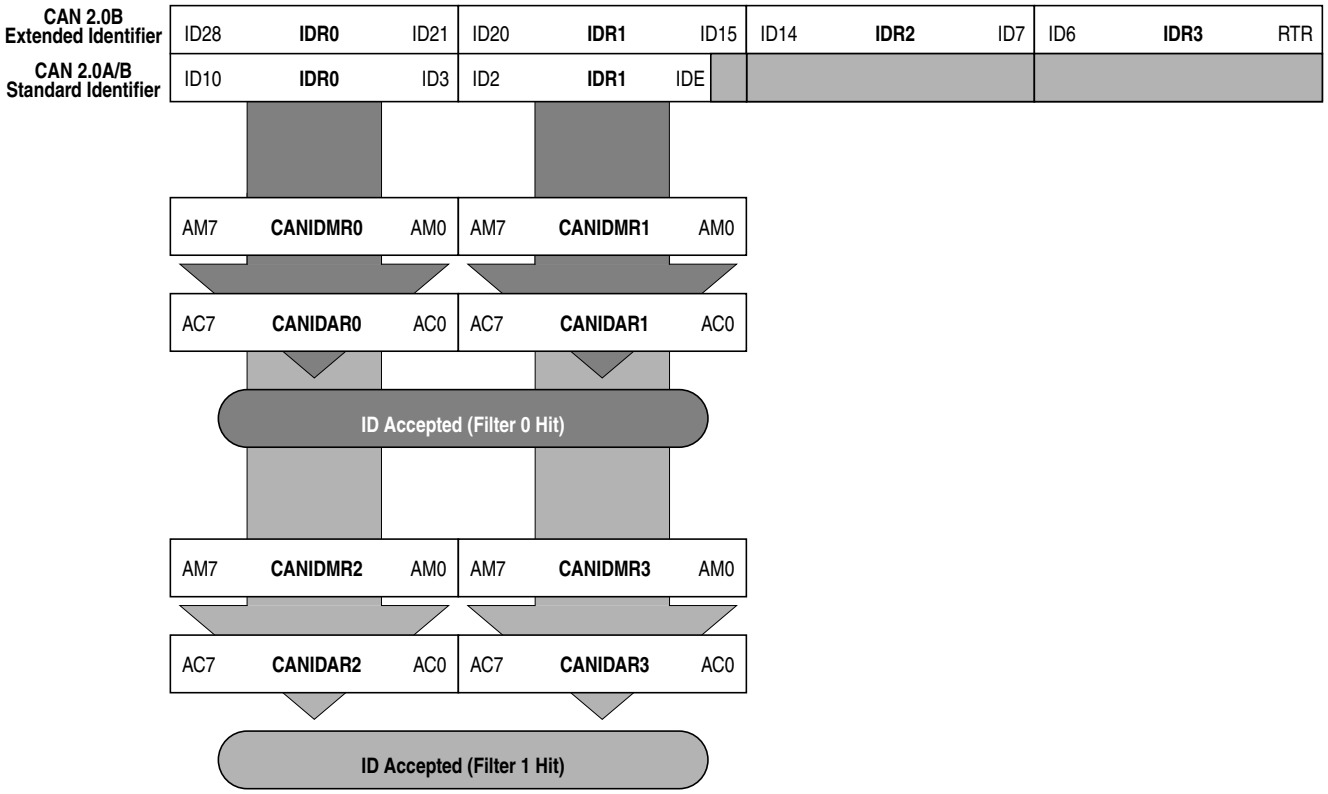

Figure 9-40. 16-bit Maskable Identifier Acceptance Filters

10.3.2.5 SCI Status Register 2 (SCISR2)

Module Base + 0x_0005

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	BK13	TXDIR	RAF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-7. SCI Status Register 2 (SCISR2)

Read: Anytime

Write: Anytime; writing accesses SCI status register 2; writing to any bits except TXDIR and BRK13 (SCISR2[1] & [2]) has no effect

Table 10-6. SCISR2 Field Descriptions

Field	Description
2 BK13	**Break Transmit Character Length** — This bit determines whether the transmit break character is 10 or 11 bit respectively 13 or 14 bits long. The detection of a framing error is not affected by this bit. 0 Break Character is 10 or 11 bit long 1 Break character is 13 or 14 bit long
1 TXDIR	**Transmitter Pin Data Direction in Single-Wire Mode.** — This bit determines whether the TXD pin is going to be used as an input or output, in the Single-Wire mode of operation. This bit is only relevant in the Single-Wire mode of operation. 0 TXD pin to be used as an input in Single-Wire mode 1 TXD pin to be used as an output in Single-Wire mode
0 RAF	**Receiver Active Flag** — RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 No reception in progress 1 Reception in progress

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x001A PWMPER6	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001B PWMPER7	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001C PWMDTY0	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001D PWMDTY1	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001E PWMDTY2	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x001F PWMDTY3	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0010 PWMDTY4	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0021 PWMDTY5	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0022 PWMDTY6	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0023 PWMDTY7	R	Bit 7	6	5	4	3	2	1	Bit 0
	W								
0x0024 PWMSDN	R	PWMIF	PWMIE	0	PWMLVL	0	PWM7IN	PWM7INL	PWM7ENA
	W			PWMRSTRT					

= Unimplemented or Reserved

Figure 12-2. PWM Register Summary (Sheet 3 of 3)

1 Intended for factory test purposes only.

12.3.2.1 PWM Enable Register (PWME)

Each PWM channel has an enable bit (PWMEx) to start its waveform output. When any of the PWMEx bits are set (PWMEx = 1), the associated PWM output is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWMEx and the clock source.

NOTE

The first PWM cycle after enabling the channel can be irregular.

Table 13-22. PAFLG Field Descriptions

Field	Description
1 PAOVF	**Pulse Accumulator Overflow Flag** — Set when the 16-bit pulse accumulator overflows from 0xFFFF to 0x0000. Clearing this bit requires wirting a one to this bit in the PAFLG register while TEN bit of TSCR1 register is set to one.
0 PAIF	**Pulse Accumulator Input edge Flag** — Set when the selected edge is detected at the IOC7 input pin.In event mode the event edge triggers PAIF and in gated time accumulation mode the trailing edge of the gate signal at the IOC7 input pin triggers PAIF. Clearing this bit requires writing a one to this bit in the PAFLG register while TEN bit of TSCR1 register is set to one. Any access to the PACNT register will clear all the flags in this register when TFFCA bit in register TSCR(0x0006) is set.

1 The DBG module is designed for backwards compatibility to existing BKP modules. Register and bit names have changed from the BKP module. This column shows the DBG register name, as well as the BKP register name for reference.

2 Comparator C can be used to enhance the BKP mode by providing a third breakpoint.

16.3.2.1 Debug Control Register 1 (DBGC1)

NOTE

All bits are used in DBG mode only.

Module Base + 0x0020
Starting address location affected by INITRG register setting.

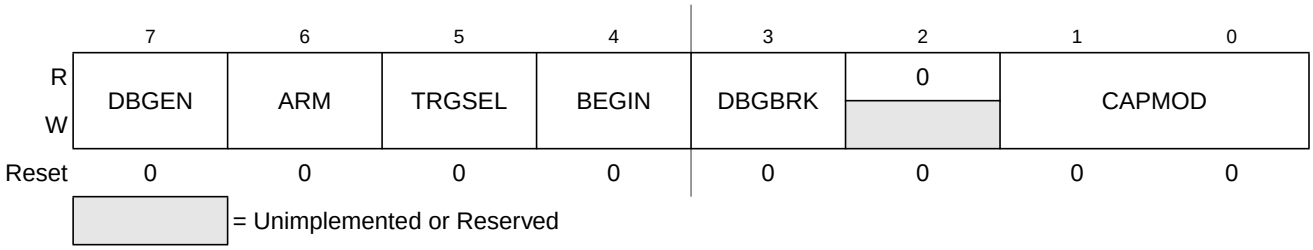

Figure 16-4. Debug Control Register (DBGC1)

NOTE

This register cannot be written if BKP mode is enabled (BKABEN in DBGC2 is set).

Table 16-3. DBGC1 Field Descriptions

Field	Description
7 DBGEN	**DBG Mode Enable Bit** — The DBGEN bit enables the DBG module for use in DBG mode. This bit cannot be set if the MCU is in secure mode. 0 DBG mode disabled 1 DBG mode enabled
6 ARM	**Arm Bit** — The ARM bit controls whether the debugger is comparing and storing data in the trace buffer. See Section 16.4.2.4, “Arming the DBG Module,” for more information. 0 Debugger unarmed 1 Debugger armed **Note:** This bit cannot be set if the DBGEN bit is not also being set at the same time. For example, a write of 01 to DBGEN[7:6] will be interpreted as a write of 00.
5 TRGSEL	**Trigger Selection Bit** — The TRGSEL bit controls the triggering condition for comparators A and B in DBG mode. It serves essentially the same function as the TAGAB bit in the DBGC2 register does in BKP mode. See Section 16.4.2.1.2, “Trigger Selection,” for more information. TRGSEL may also determine the type of breakpoint based on comparator A and B if enabled in DBG mode (DBGBRK = 1). Please refer to Section 16.4.3.1, “Breakpoint Based on Comparator A and B.” 0 Trigger on any compare address match 1 Trigger before opcode at compare address gets executed (tagged-type)

Module Base + 0x002F
Starting address location affected by INITRG register setting.

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

Figure 16-21. Debug Comparator B Register Low (DBGCBL)

Table 16-23. DBGCB Field Descriptions

Field	Description
15:0 15:0	**Comparator B Compare Bits** — The comparator B compare bits control whether comparator B compares the address bus bits [15:0] or data bus bits [15:0] to a logic 1 or logic 0. See Table 16-20. 0 Compare corresponding address bit to a logic 0, compares to data if in Full mode 1 Compare corresponding address bit to a logic 1, compares to data if in Full mode

16.4 Functional Description

This section provides a complete functional description of the DBG module. The DBG module can be configured to run in either of two modes, BKP or DBG. BKP mode is enabled by setting BKABEN in DBGC2. DBG mode is enabled by setting DBGEN in DBGC1. Setting BKABEN in DBGC2 overrides the DBGEN in DBGC1 and prevents DBG mode. If the part is in secure mode, DBG mode cannot be enabled.

16.4.1 DBG Operating in BKP Mode

In BKP mode, the DBG will be fully backwards compatible with the existing BKP_ST12_A module. The DBGC2 register has four additional bits that were not available on existing BKP_ST12_A modules. As long as these bits are written to either all 1s or all 0s, they should be transparent to the user. All 1s would enable comparator C to be used as a breakpoint, but tagging would be enabled. The match address register would be all 0s if not modified by the user. Therefore, code executing at address 0x0000 would have to occur before a breakpoint based on comparator C would happen.

The DBG module in BKP mode supports two modes of operation: dual address mode and full breakpoint mode. Within each of these modes, forced or tagged breakpoint types can be used. Forced breakpoints occur at the next instruction boundary if a match occurs and tagged breakpoints allow for breaking just before the tagged instruction executes. The action taken upon a successful match can be to either place the CPU in background debug mode or to initiate a software interrupt.

The breakpoint can operate in dual address mode or full breakpoint mode. Each of these modes is discussed in the subsections below.

16.4.1.1 Dual Address Mode

When dual address mode is enabled, two address breakpoints can be set. Each breakpoint can cause the system to enter background debug mode or to initiate a software interrupt based upon the state of BDM in

16.4.2.3 Begin- and End-Trigger

The definitions of begin- and end-trigger as used in the DBG module are as follows:

- Begin-trigger: Storage in trace buffer occurs after the trigger and continues until 64 locations are filled.
- End-trigger: Storage in trace buffer occurs until the trigger, with the least recent data falling out of the trace buffer if more than 64 words are collected.

16.4.2.4 Arming the DBG Module

In DBG mode, arming occurs by setting DBGEN and ARM in DBGC1. The ARM bit in DBGC1 is cleared when the trigger condition is met in end-trigger mode or when the Trace Buffer is filled in begin-trigger mode. The TBC logic determines whether a trigger condition has been met based on the trigger mode and the trigger selection.

16.4.2.5 Trigger Modes

The DBG module supports nine trigger modes. The trigger modes are encoded as shown in Table 16-6. The trigger mode is used as a qualifier for either starting or ending the storing of data in the trace buffer. When the match condition is met, the appropriate flag A or B is set in DBGSC. Arming the DBG module clears the A, B, and C flags in DBGSC. In all trigger modes except for the event-only modes and DETAIL capture mode, change-of-flow addresses are stored in the trace buffer. In the event-only modes only the value on the data bus at the trigger event B will be stored. In DETAIL capture mode address and data for all cycles except program fetch (P) and free (f) cycles are stored in trace buffer.

16.4.2.5.1 A Only

In the A only trigger mode, if the match condition for A is met, the A flag in DBGSC is set and a trigger occurs.

16.4.2.5.2 A or B

In the A or B trigger mode, if the match condition for A or B is met, the corresponding flag in DBGSC is set and a trigger occurs.

16.4.2.5.3 A then B

In the A then B trigger mode, the match condition for A must be met before the match condition for B is compared. When the match condition for A or B is met, the corresponding flag in DBGSC is set. The trigger occurs only after A then B have matched.

NOTE

When tagging and using A then B, if addresses A and B are close together, then B may not complete the trigger sequence. This occurs when A and B are in the instruction queue at the same time. Basically the A trigger has not yet occurred, so the B instruction is not tagged. Generally, if address B is at

18.3.2.5 Reserved Registers

Module Base + 0x0004
Starting address location affected by INITRG register setting.

	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 18-6. Reserved Register

Module Base + 0x0005
Starting address location affected by INITRG register setting.

	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 18-7. Reserved Register

Module Base + 0x0006
Starting address location affected by INITRG register setting.

	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 18-8. Reserved Register

Module Base + 0x0007
Starting address location affected by INITRG register setting.

	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 18-9. Reserved Register

18.3.2.16 Port K Data Direction Register (DDRK)

Module Base + 0x0033
Starting address location affected by INITRG register setting.

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

Figure 18-20. Port K Data Direction Register (DDRK)

Read: Anytime

Write: Anytime

This register determines the primary direction for each port K pin configured as general-purpose I/O. This register is not in the map in peripheral or expanded modes while the EMK control bit in MODE register is set. Therefore, these accesses will be echoed externally.

Table 18-14. EBICTL Field Descriptions

Field	Description
7:0 DDRK	**Data Direction Port K Bits** 0 Associated pin is a high-impedance input 1 Associated pin is an output **Note:** It is unwise to write PORTK and DDRK as a word access. If you are changing port K pins from inputs to outputs, the data may have extra transitions during the write. It is best to initialize PORTK before enabling as outputs. **Note:** To ensure that you read the correct value from the PORTK pins, always wait at least one cycle after writing to the DDRK register before reading from the PORTK register.

18.4 Functional Description

18.4.1 Detecting Access Type from External Signals

The external signals $\overline{\text{LSTRB}}$, R/$\overline{\text{W}}$, and AB0 indicate the type of bus access that is taking place. Accesses to the internal RAM module are the only type of access that would produce $\overline{\text{LSTRB}}$ = AB0 = 1, because the internal RAM is specifically designed to allow misaligned 16-bit accesses in a single cycle. In these cases the data for the address that was accessed is on the low half of the data bus and the data for address + 1 is on the high half of the data bus. This is summarized in Table 18-15.

Table 18-15. Access Type vs. Bus Control Pins

$\overline{\text{LSTRB}}$	AB0	R/$\overline{\text{W}}$	Type of Access
1	0	1	8-bit read of an even address
0	1	1	8-bit read of an odd address
1	0	0	8-bit write of an even address
0	1	0	8-bit write of an odd address

Table A-15. 3.3V ATD Conversion Performance

Conditions are shown in Table A-4 unless otherwise noted $V_{REF} = V_{RH} - V_{RL} = 3.328V$. Resulting to one 8 bit count = 13mV and one 10 bit count = 3.25mV. $f_{ATDCLK} = 2.0MHz$							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	10-Bit Resolution	LSB	—	3.25	—	mV
2	P	10-Bit Differential Nonlinearity	DNL	–1.5		1.5	Counts
3	P	10-Bit Integral Nonlinearity	INL	–3.5	±1.5	3.5	Counts
4	P	10-Bit Absolute Error[1]	AE	–5	±2.5	5	Counts
5	C	10-Bit Absolute Error at f_{ATDCLK}= 4MHz	AE	—	±7.0	—	Counts
6	P	8-Bit Resolution	LSB	—	13	—	mV
7	P	8-Bit Differential Nonlinearity	DNL	–0.5	—	—	Counts
8	P	8-Bit Integral Nonlinearity	INL	–1.5	±0.1	1.5	Counts
9	P	8-Bit Absolute Error[1]	AE	–2.0	±1.5	2.0	Counts

1 These values include the quantization error which is inherently 1/2 count for any A/D converter.

For the following definitions see also Figure A-2.

Differential Non-Linearity (DNL) is defined as the difference between two adjacent switching steps.

$$DNL(i) = \frac{V_i - V_{i-1}}{1LSB} - 1$$

The Integral Non-Linearity (INL) is defined as the sum of all DNLs:

$$INL(n) = \sum_{i=1}^{n} DNL(i) = \frac{V_n - V_0}{1LSB} - n$$