

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	S08
Core Size	8-Bit
Speed	40MHz
Connectivity	CANbus, I ² C, LINbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	48KB (48K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 10x12b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	32-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s08dv48mlc

Revision History

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

<http://freescale.com/>

The following revision history table summarizes changes contained in this document.

Revision Number	Revision Date	Description of Changes
1	6/2006	Advance Information version for alpha samples customers
2	9/2007	Product Launch. Removed the 64-pin QFN package. Changed from standard to extended mode for MSCAN registers in register summary. Corrected Block diagrams for SCI. Updated the latest Temp Sensor information. Made FTSTMOD reserved. Updated device to use the ADC 12-bit module. Revised the MCG module. Updated the TPM block module to version 3. Added the TPM block module version 2 as an appendix for devices using 3M05C (or earlier) mask sets. Heavily revised the Electricals appendix.
3	6/2008	Sustaining Update. Incorporated PS Issues # 2765, 3177, 3236, 3292, 3301, 3311, 3312, 3326, 3335, 3345, 3382, 2795, 3382 and 3386 PLL Jitter Spec update. Also, added internal reference clock trim adjustment statement to Features page. Updated the TPM module to the latest version. Adjusted values in Table A-13 Control Timing row 2 and in Table A-6 DC Characteristics row 24 so that it references 5.0 V instead of 3.0 V.

© Freescale Semiconductor, Inc., 2007-2008. All rights reserved.

This product incorporates SuperFlash[®] Technology licensed from SST.

Section Number	Title	Page
4.5.2	Program and Erase Times	53
4.5.3	Program and Erase Command Execution	53
4.5.4	Burst Program Execution	55
4.5.5	Sector Erase Abort	57
4.5.6	Access Errors	58
4.5.7	Block Protection	59
4.5.8	Vector Redirection	59
4.5.9	Security	59
4.5.10	Flash Registers and Control Bits	61

Chapter 5 Resets, Interrupts, and General System Control

5.1	Introduction	67
5.2	Features	67
5.3	MCU Reset	67
5.4	Computer Operating Properly (COP) Watchdog	68
5.5	Interrupts	69
5.5.1	Interrupt Stack Frame	70
5.5.2	External Interrupt Request (IRQ) Pin	70
5.5.3	Interrupt Vectors, Sources, and Local Masks	71
5.6	Low-Voltage Detect (LVD) System	73
5.6.1	Power-On Reset Operation	73
5.6.2	Low-Voltage Detection (LVD) Reset Operation	73
5.6.3	Low-Voltage Warning (LVW) Interrupt Operation	73
5.7	MCLK Output	73
5.8	Reset, Interrupt, and System Control Registers and Control Bits	74
5.8.1	Interrupt Pin Request Status and Control Register (IRQSC)	75
5.8.2	System Reset Status Register (SRS)	76
5.8.3	System Background Debug Force Reset Register (SBDFR)	77
5.8.4	System Options Register 1 (SOPT1)	78
5.8.5	System Options Register 2 (SOPT2)	79
5.8.6	System Device Identification Register (SDIDH, SDIDL)	80
5.8.7	System Power Management Status and Control 1 Register (SPMSC1)	81
5.8.8	System Power Management Status and Control 2 Register (SPMSC2)	82

Chapter 6 Parallel Input/Output Control

6.1	Port Data and Data Direction	83
6.2	Pull-up, Slew Rate, and Drive Strength	84
6.3	Pin Interrupts	85
6.3.1	Edge Only Sensitivity	85
6.3.2	Edge and Level Sensitivity	86

4.5.4 Burst Program Execution

The burst program command is used to program sequential bytes of data in less time than would be required using the standard program command. This is possible because the high voltage to the Flash array does not need to be disabled between program operations. Ordinarily, when a program or erase command is issued, an internal charge pump associated with the Flash memory must be enabled to supply high voltage to the array. Upon completion of the command, the charge pump is turned off. When a burst program command is issued, the charge pump is enabled and remains enabled after completion of the burst program operation if these two conditions are met:

- The next burst program command sequence has begun before the FCCF bit is set.
- The next sequential address selects a byte on the same burst block as the current byte being programmed. A burst block in this Flash memory consists of 32 bytes. A new burst block begins at each 32-byte address boundary.

The first byte of a series of sequential bytes being programmed in burst mode will take the same amount of time to program as a byte programmed in standard mode. Subsequent bytes will program in the burst program time provided that the conditions above are met. If the next sequential address is the beginning of a new row, the program time for that byte will be the standard time instead of the burst time. This is because the high voltage to the array must be disabled and then enabled again. If a new burst command has not been queued before the current command completes, then the charge pump will be disabled and high voltage removed from the array.

A flowchart to execute the burst program operation is shown in [Figure 4-3](#).

7.2 Programmer's Model and CPU Registers

Figure 7-1 shows the five CPU registers. CPU registers are not part of the memory map.

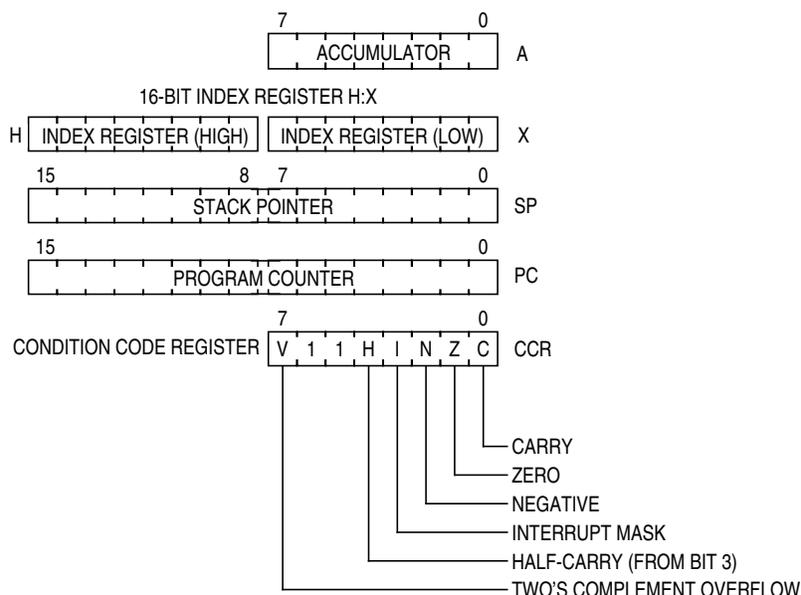


Figure 7-1. CPU Registers

7.2.1 Accumulator (A)

The A accumulator is a general-purpose 8-bit register. One operand input to the arithmetic logic unit (ALU) is connected to the accumulator and the ALU results are often stored into the A accumulator after arithmetic and logical operations. The accumulator can be loaded from memory using various addressing modes to specify the address where the loaded data comes from, or the contents of A can be stored to memory using various addressing modes to specify the address where data from A will be stored.

Reset has no effect on the contents of the A accumulator.

7.2.2 Index Register (H:X)

This 16-bit register is actually two separate 8-bit registers (H and X), which often work together as a 16-bit address pointer where H holds the upper byte of an address and X holds the lower byte of the address. All indexed addressing mode instructions use the full 16-bit value in H:X as an index reference pointer; however, for compatibility with the earlier M68HC05 Family, some instructions operate only on the low-order 8-bit half (X).

Many instructions treat X as a second general-purpose 8-bit register that can be used to hold 8-bit data values. X can be cleared, incremented, decremented, complemented, negated, shifted, or rotated. Transfer instructions allow data to be transferred from A or transferred to A where arithmetic and logical operations can then be performed.

For compatibility with the earlier M68HC05 Family, H is forced to 0x00 during reset. Reset has no effect on the contents of X.

Table 7-2. Instruction Set Summary (Sheet 7 of 9)

Source Form	Operation	Address Mode	Object Code	Cycles	Cyc-by-Cyc Details	Affect on CCR	
						V 1 1 H	I N Z C
RSP	Reset Stack Pointer (Low Byte) SPL ← \$FF (High Byte Not Affected)	INH	9C	1	p	- 1 1 -	- - - -
RTI	Return from Interrupt SP ← (SP) + \$0001; Pull (CCR) SP ← (SP) + \$0001; Pull (A) SP ← (SP) + \$0001; Pull (X) SP ← (SP) + \$0001; Pull (PCH) SP ← (SP) + \$0001; Pull (PCL)	INH	80	9	uuuuufppp	↑ 1 1 ↓	↑ ↓ ↑ ↓
RTS	Return from Subroutine SP ← SP + \$0001; Pull (PCH) SP ← SP + \$0001; Pull (PCL)	INH	81	5	ufppp	- 1 1 -	- - - -
SBC #opr8i SBC opr8a SBC opr16a SBC oprx16,X SBC oprx8,X SBC ,X SBC oprx16,SP SBC oprx8,SP	Subtract with Carry A ← (A) – (M) – (C)	IMM DIR EXT IX2 IX1 IX SP2 SP1	A2 ii B2 dd C2 hh ll D2 ee ff E2 ff F2 9E D2 ee ff 9E E2 ff	2 3 4 4 3 3 5 4	pp rpp prpp prpp rpp rff pprpp prpp	↑ 1 1 -	- ↑ ↓ ↓
SEC	Set Carry Bit (C ← 1)	INH	99	1	p	- 1 1 -	- - - 1
SEI	Set Interrupt Mask Bit (I ← 1)	INH	9B	1	p	- 1 1 -	1 - - -
STA opr8a STA opr16a STA oprx16,X STA oprx8,X STA ,X STA oprx16,SP STA oprx8,SP	Store Accumulator in Memory M ← (A)	DIR EXT IX2 IX1 IX SP2 SP1	B7 dd C7 hh ll D7 ee ff E7 ff F7 9E D7 ee ff 9E E7 ff	3 4 4 3 2 5 4	wpp pwpp pwpp wpp wp ppwpp pwpp	0 1 1 -	- ↓ ↓ -
STHX opr8a STHX opr16a STHX oprx8,SP	Store H:X (Index Reg.) (M:M + \$0001) ← (H:X)	DIR EXT SP1	35 dd 96 hh ll 9E FF ff	4 5 5	wpp pwpp pwpp	0 1 1 -	- ↓ ↓ -
STOP	Enable Interrupts: Stop Processing Refer to MCU Documentation I bit ← 0; Stop Processing	INH	8E	2	fp...	- 1 1 -	0 - - -
STX opr8a STX opr16a STX oprx16,X STX oprx8,X STX ,X STX oprx16,SP STX oprx8,SP	Store X (Low 8 Bits of Index Register) in Memory M ← (X)	DIR EXT IX2 IX1 IX SP2 SP1	BF dd CF hh ll DF ee ff EF ff FF 9E DF ee ff 9E EF ff	3 4 4 3 2 5 4	wpp pwpp pwpp wpp wp ppwpp pwpp	0 1 1 -	- ↓ ↓ -

9.1.2 Features

The ACMP has the following features:

- Full rail to rail supply operation.
- Selectable interrupt on rising edge, falling edge, or either rising or falling edges of comparator output.
- Option to compare to fixed internal bandgap reference voltage.
- Option to allow comparator output to be visible on a pin, ACMPxO.

9.1.3 Modes of Operation

This section defines the ACMP operation in wait, stop, and background debug modes.

9.1.3.1 ACMP in Wait Mode

The ACMP continues to run in wait mode if enabled before executing the appropriate instruction. Therefore, the ACMP can be used to bring the MCU out of wait mode if the ACMP interrupt is enabled (ACIE is set). For lowest possible current consumption, the ACMP should be disabled by software if not required as an interrupt source during wait mode.

9.1.3.2 ACMP in Stop Modes

The ACMP is disabled in all stop modes, regardless of the settings before executing the stop instruction. Therefore, the ACMP cannot be used as a wake up source from stop modes.

During stop2 mode, the ACMP module is fully powered down. Upon wake-up from stop2 mode, the ACMP module is in the reset state.

During stop3 mode, clocks to the ACMP module are halted. No registers are affected. In addition, the ACMP comparator circuit enters a low-power state. No compare operation occurs while in stop3.

If stop3 is exited with a reset, the ACMP is put into its reset state. If stop3 is exited with an interrupt, the ACMP continues from the state it was in when stop3 was entered.

9.1.3.3 ACMP in Active Background Mode

When the microcontroller is in active background mode, the ACMP continues to operate normally.

In 10-bit mode, the ADCCVH register holds the upper two bits of the 10-bit compare value (ADCV[9:8]). These bits are compared to the upper two bits of the result following a conversion in 10-bit mode when the compare function is enabled.

In 8-bit mode, ADCCVH is not used during compare.

10.3.6 Compare Value Low Register (ADCCVL)

This register holds the lower 8 bits of the 12-bit or 10-bit compare value or all 8 bits of the 8-bit compare value. When the compare function is enabled, bits ADCV[7:0] are compared to the lower 8 bits of the result following a conversion in 12-bit, 10-bit or 8-bit mode.

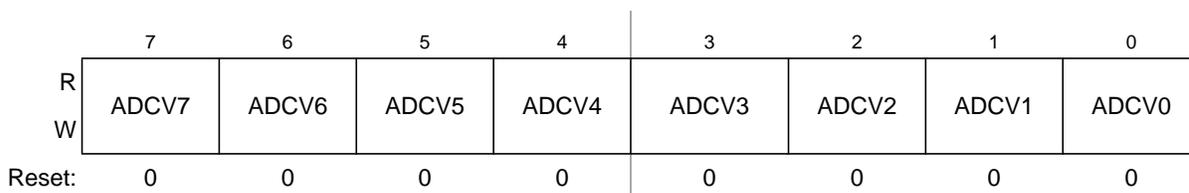


Figure 10-8. Compare Value Low Register (ADCCVL)

10.3.7 Configuration Register (ADCCFG)

ADCCFG selects the mode of operation, clock source, clock divide, and configures for low power and long sample time.

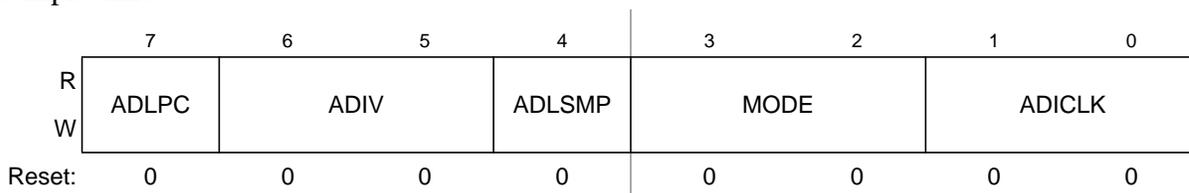


Figure 10-9. Configuration Register (ADCCFG)

Table 10-6. ADCCFG Register Field Descriptions

Field	Description
7 ADLPC	Low-Power Configuration. ADLPC controls the speed and power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 High speed configuration 1 Low power configuration: The power is reduced at the expense of maximum clock speed.
6:5 ADIV	Clock Divide Select. ADIV selects the divide ratio used by the ADC to generate the internal clock ADCK. Table 10-7 shows the available clock configurations.
4 ADLSMP	Long Sample Time Configuration. ADLSMP selects between long and short sample time. This adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. 0 Short sample time 1 Long sample time

11.1.1 Features

The IIC includes these distinctive features:

- Compatible with IIC bus standard
- Multi-master operation
- Software programmable for one of 64 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation
- Acknowledge bit generation/detection
- Bus busy detection
- General call recognition
- 10-bit address extension

11.1.2 Modes of Operation

A brief description of the IIC in the various MCU modes is given here.

- **Run mode** — This is the basic mode of operation. To conserve power in this mode, disable the module.
- **Wait mode** — The module continues to operate while the MCU is in wait mode and can provide a wake-up interrupt.
- **Stop mode** — The IIC is inactive in stop3 mode for reduced power consumption. The stop instruction does not affect IIC register states. Stop2 resets the register contents.

Table 11-4. IIC Divider and Hold Values

ICR (hex)	SCL Divider	SDA Hold Value	SCL Hold (Start) Value	SDA Hold (Stop) Value
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
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

ICR (hex)	SCL Divider	SDA Hold Value	SCL Hold (Start) Value	SCL Hold (Stop) Value
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
3E	3072	513	1534	1537
3F	3840	513	1918	1921

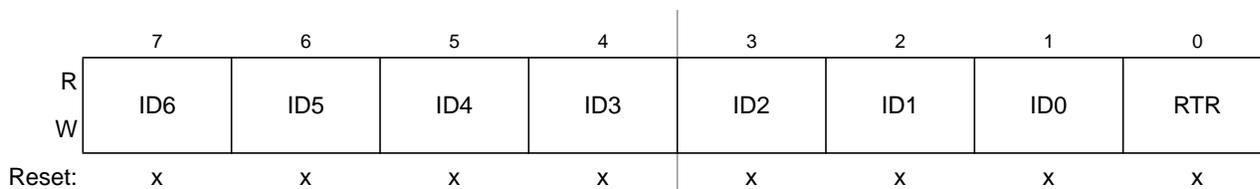


Figure 12-28. Identifier Register 3 (IDR3) — Extended Identifier Mapping

Table 12-28. IDR3 Register Field Descriptions — Extended

Field	Description
7:1 ID[6:0]	Extended Format Identifier — The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.
0 RTR	Remote Transmission Request — This flag reflects the status of the remote transmission request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent. 0 Data frame 1 Remote frame

12.4.2 IDR0–IDR3 for Standard Identifier Mapping

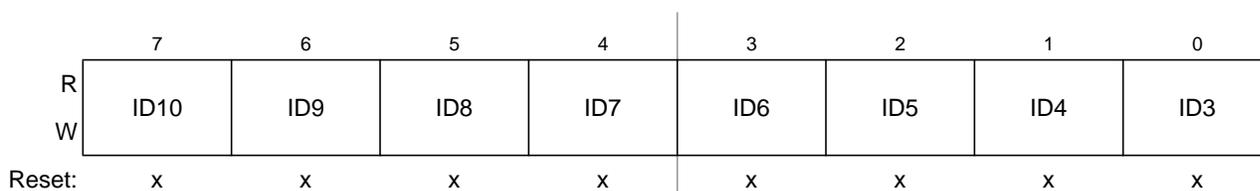
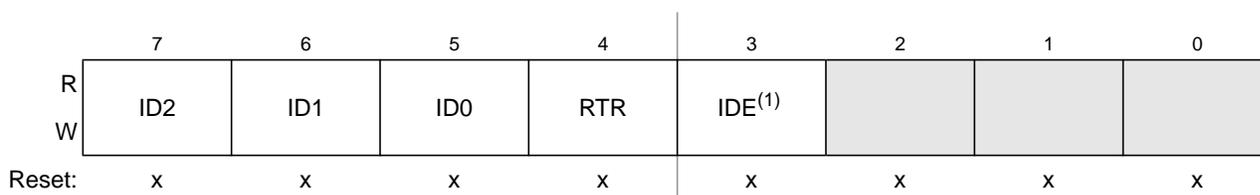


Figure 12-29. Identifier Register 0 — Standard Mapping

Table 12-29. IDR0 Register Field Descriptions — Standard

Field	Description
7:0 ID[10:3]	Standard Format Identifier — The identifiers consist of 11 bits (ID[10:0]) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number. See also ID bits in Table 12-30 .



= Unused; always read 'x'

Figure 12-30. Identifier Register 1 — Standard Mapping

¹ IDE is 0.

Table 12-33. Data Length Codes

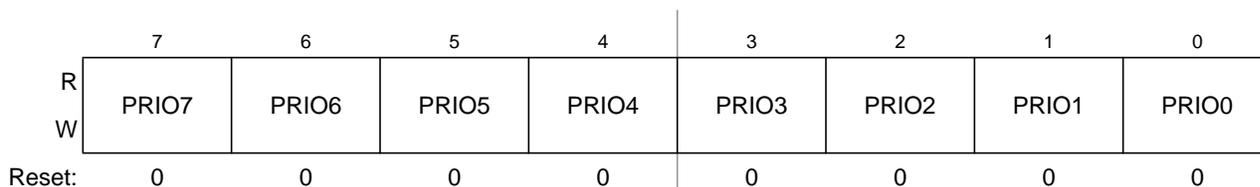
Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8

12.4.5 Transmit Buffer Priority Register (TBPR)

This register defines the local priority of the associated message transmit buffer. The local priority is used for the internal prioritization process of the MSCAN and is defined to be highest for the smallest binary number. The MSCAN implements the following internal prioritization mechanisms:

- All transmission buffers with a cleared TXEx flag participate in the prioritization immediately before the SOF (start of frame) is sent.
- The transmission buffer with the lowest local priority field wins the prioritization.

In cases of more than one buffer having the same lowest priority, the message buffer with the lower index number wins.


Figure 12-35. Transmit Buffer Priority Register (TBPR)

Read: Anytime when TXEx flag is set (see Section 12.3.6, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.10, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”).

Write: Anytime when TXEx flag is set (see Section 12.3.6, “MSCAN Transmitter Flag Register (CANTFLG)”) and the corresponding transmit buffer is selected in CANTBSEL (see Section 12.3.10, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”).

12.4.6 Time Stamp Register (TSRH–TSRL)

If the TIME bit is enabled, the MSCAN will write a time stamp to the respective registers in the active transmit or receive buffer as soon as a message has been acknowledged on the CAN bus (see

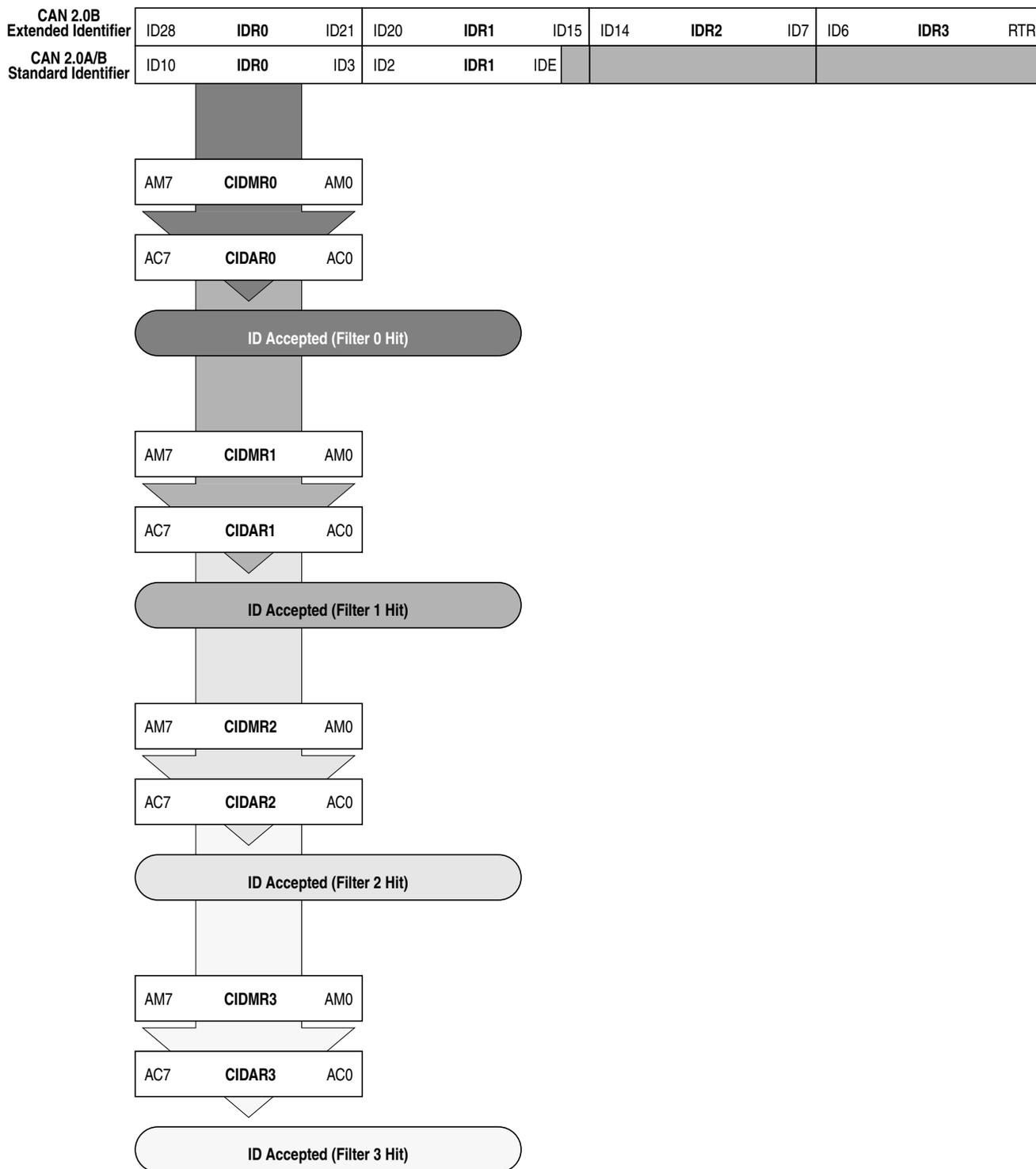


Figure 12-41. 8-bit Maskable Identifier Acceptance Filters

MSCAN filter uses three sets of registers to provide the filter configuration. Firstly, the CANIDAC register determines the configuration of the banks into filter sizes and number of filters. Secondly, registers CANIDMR0/1/2/3 determine those bits on which the filter will operate by placing a '0' at the appropriate

Table 14-8. SCIxC3 Field Descriptions (continued)

Field	Description
4 TXINV ¹	Transmit Data Inversion — Setting this bit reverses the polarity of the transmitted data output. 0 Transmit data not inverted 1 Transmit data inverted
3 ORIE	Overrun Interrupt Enable — This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0 OR interrupts disabled (use polling). 1 Hardware interrupt requested when OR = 1.
2 NEIE	Noise Error Interrupt Enable — This bit enables the noise flag (NF) to generate hardware interrupt requests. 0 NF interrupts disabled (use polling). 1 Hardware interrupt requested when NF = 1.
1 FEIE	Framing Error Interrupt Enable — This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0 FE interrupts disabled (use polling). 1 Hardware interrupt requested when FE = 1.
0 PEIE	Parity Error Interrupt Enable — This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0 PF interrupts disabled (use polling). 1 Hardware interrupt requested when PF = 1.

¹ Setting TXINV inverts the TxD output for all cases: data bits, start and stop bits, break, and idle.

14.2.7 SCI Data Register (SClxD)

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for the SCI status flags.

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

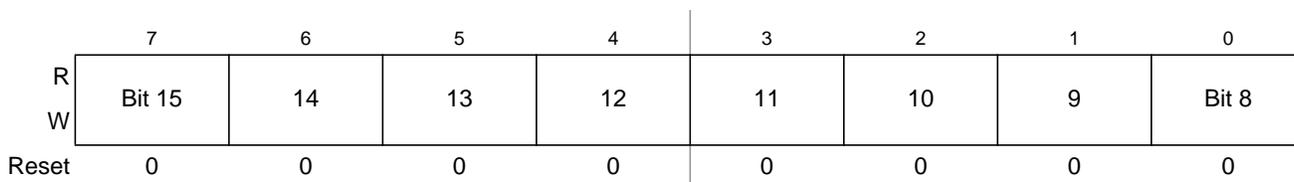
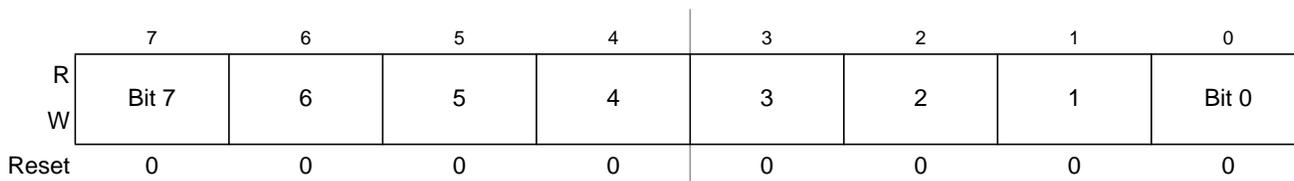
Figure 14-11. SCI Data Register (SClxD)

14.3 Functional Description

The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The SCI comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. During normal operation, the MCU monitors the status of the SCI, writes the data to be transmitted, and processes received data. The following describes each of the blocks of the SCI.

14.3.1 Baud Rate Generation

As shown in [Figure 14-12](#), the clock source for the SCI baud rate generator is the bus-rate clock.


Figure 16-13. TPM Channel Value Register High (TPMxCnVH)

Figure 16-14. TPM Channel Value Register Low (TPMxCnVL)

In input capture mode, reading either byte (TPMxCnVH or TPMxCnVL) latches the contents of both bytes into a buffer where they remain latched until the other half is read. This latching mechanism also resets (becomes unlatched) when the TPMxCnSC register is written (whether BDM mode is active or not). Any write to the channel registers will be ignored during the input capture mode.

When BDM is active, the coherency mechanism is frozen (unless reset by writing to TPMxCnSC register) such that the buffer latches remain in the state they were in when the BDM became active, even if one or both halves of the channel register are read while BDM is active. This assures that if the user was in the middle of reading a 16-bit register when BDM became active, it will read the appropriate value from the other half of the 16-bit value after returning to normal execution. The value read from the TPMxCnVH and TPMxCnVL registers in BDM mode is the value of these registers and not the value of their read buffer.

In output compare or PWM modes, writing to either byte (TPMxCnVH or TPMxCnVL) latches the value into a buffer. After both bytes are written, they are transferred as a coherent 16-bit value into the timer-channel registers according to the value of CLKS_B:CLKS_A bits and the selected mode, so:

- If (CLKS_B:CLKS_A = 0:0), then the registers are updated when the second byte is written.
- If (CLKS_B:CLKS_A not = 0:0 and in output compare mode) then the registers are updated after the second byte is written and on the next change of the TPM counter (end of the prescaler counting).
- If (CLKS_B:CLKS_A not = 0:0 and in EPWM or CPWM modes), then the registers are updated after the both bytes were written, and the TPM counter changes from (TPMxMODH:TPMxMODL - 1) to (TPMxMODH:TPMxMODL). If the TPM counter is a free-running counter then the update is made when the TPM counter changes from 0xFFFFE to 0xFFFF.

The latching mechanism may be manually reset by writing to the TPMxCnSC register (whether BDM mode is active or not). This latching mechanism allows coherent 16-bit writes in either big-endian or little-endian order which is friendly to various compiler implementations.

When BDM is active, the coherency mechanism is frozen such that the buffer latches remain in the state they were in when the BDM became active even if one or both halves of the channel register are written while BDM is active. Any write to the channel registers bypasses the buffer latches and directly write to the channel register while BDM is active. The values written to the channel register while BDM is active

A-Only — Trigger when the address matches the value in comparator A

A OR B — Trigger when the address matches either the value in comparator A or the value in comparator B

A Then B — Trigger when the address matches the value in comparator B but only after the address for another cycle matched the value in comparator A. There can be any number of cycles after the A match and before the B match.

A AND B Data (Full Mode) — This is called a full mode because address, data, and R/W (optionally) must match within the same bus cycle to cause a trigger event. Comparator A checks address, the low byte of comparator B checks data, and R/W is checked against RWA if RWAEN = 1. The high-order half of comparator B is not used.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

A AND NOT B Data (Full Mode) — Address must match comparator A, data must not match the low half of comparator B, and R/W must match RWA if RWAEN = 1. All three conditions must be met within the same bus cycle to cause a trigger.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

Event-Only B (Store Data) — Trigger events occur each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

A Then Event-Only B (Store Data) — After the address has matched the value in comparator A, a trigger event occurs each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

Inside Range ($A \leq \text{Address} \leq B$) — A trigger occurs when the address is greater than or equal to the value in comparator A and less than or equal to the value in comparator B at the same time.

Outside Range ($\text{Address} < A$ or $\text{Address} > B$) — A trigger occurs when the address is either less than the value in comparator A or greater than the value in comparator B.

17.4.3.5 Debug FIFO High Register (DBGFH)

This register provides read-only access to the high-order eight bits of the FIFO. Writes to this register have no meaning or effect. In the event-only trigger modes, the FIFO only stores data into the low-order byte of each FIFO word, so this register is not used and will read 0x00.

Reading DBGFH does not cause the FIFO to shift to the next word. When reading 16-bit words out of the FIFO, read DBGFH before reading DBGFL because reading DBGFL causes the FIFO to advance to the next word of information.

17.4.3.6 Debug FIFO Low Register (DBGFL)

This register provides read-only access to the low-order eight bits of the FIFO. Writes to this register have no meaning or effect.

Reading DBGFL causes the FIFO to shift to the next available word of information. When the debug module is operating in event-only modes, only 8-bit data is stored into the FIFO (high-order half of each FIFO word is unused). When reading 8-bit words out of the FIFO, simply read DBGFL repeatedly to get successive bytes of data from the FIFO. It isn't necessary to read DBGFH in this case.

Do not attempt to read data from the FIFO while it is still armed (after arming but before the FIFO is filled or ARMF is cleared) because the FIFO is prevented from advancing during reads of DBGFL. This can interfere with normal sequencing of reads from the FIFO.

Reading DBGFL while the debugger is not armed causes the address of the most-recently fetched opcode to be stored to the last location in the FIFO. By reading DBGFH then DBGFL periodically, external host software can develop a profile of program execution. After eight reads from the FIFO, the ninth read will return the information that was stored as a result of the first read. To use the profiling feature, read the FIFO eight times without using the data to prime the sequence and then begin using the data to get a delayed picture of what addresses were being executed. The information stored into the FIFO on reads of DBGFL (while the FIFO is not armed) is the address of the most-recently fetched opcode.

Table A-3. Thermal Characteristics

Num	C	Rating	Symbol	Value	Unit	Temp. Code	
1	D	Operating temperature range (packaged)	T_A	-40 to 125 -40 to 105 -40 to 85	°C	M V C	
2	T	Maximum Junction Temperature ¹	T_J	135	°C	—	
3	D	Thermal resistance ²					
		Single-layer board					
		64-pin LQFP	θ_{JA}	69	°C/W		
		48-pin LQFP	θ_{JA}	75	°C/W		
		32-pin LQFP	θ_{JA}	80	°C/W		
		Four-Layer board					
		64-pin LQFP	θ_{JA}	51	°C/W		
48-pin LQFP	θ_{JA}	51	°C/W				
32-pin LQFP	θ_{JA}	52	°C/W				

¹ Junction temperature is a function of die size, on-chip power dissipation, package thermal resistance, mounting site (board) temperature, ambient temperature, air flow, power dissipation of other components on the board, and board thermal resistance.

² Junction to Ambient Natural Convection

The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \times \theta_{JA}) \quad \text{Eqn. A-1}$$

where:

T_A = Ambient temperature, °C

θ_{JA} = Package thermal resistance, junction-to-ambient, °C/W

$P_D = P_{int} + P_{I/O}$

$P_{int} = I_{DD} \times V_{DD}$, Watts — chip internal power

$P_{I/O}$ = Power dissipation on input and output pins — user determined

For most applications, $P_{I/O} \ll P_{int}$ and can be neglected. An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K \div (T_J + 273^\circ\text{C}) \quad \text{Eqn. A-2}$$

Solving equations 1 and 2 for K gives:

$$K = P_D \times (T_A + 273^\circ\text{C}) + \theta_{JA} \times (P_D)^2 \quad \text{Eqn. A-3}$$

at the transition from the value set in the modulus register and the next lower count value. This corresponds to the end of a PWM period. (The 0x0000 count value corresponds to the center of a period.)

B.4.3 Channel Event Interrupt Description

The meaning of channel interrupts depends on the current mode of the channel (input capture, output compare, edge-aligned PWM, or center-aligned PWM).

When a channel is configured as an input capture channel, the ELSnB:ELSnA control bits select rising edges, falling edges, any edge, or no edge (off) as the edge that triggers an input capture event. When the selected edge is detected, the interrupt flag is set. The flag is cleared by the 2-step sequence described in [Section B.4.1, “Clearing Timer Interrupt Flags.”](#)

When a channel is configured as an output compare channel, the interrupt flag is set each time the main timer counter matches the 16-bit value in the channel value register. The flag is cleared by the 2-step sequence described in [Section B.4.1, “Clearing Timer Interrupt Flags.”](#)

B.4.4 PWM End-of-Duty-Cycle Events

For channels that are configured for PWM operation, there are two possibilities:

- When the channel is configured for edge-aligned PWM, the channel flag is set when the timer counter matches the channel value register that marks the end of the active duty cycle period.
- When the channel is configured for center-aligned PWM, the timer count matches the channel value register twice during each PWM cycle. In this CPWM case, the channel flag is set at the start and at the end of the active duty cycle, which are the times when the timer counter matches the channel value register.

The flag is cleared by the 2-step sequence described in [Section B.4.1, “Clearing Timer Interrupt Flags.”](#)