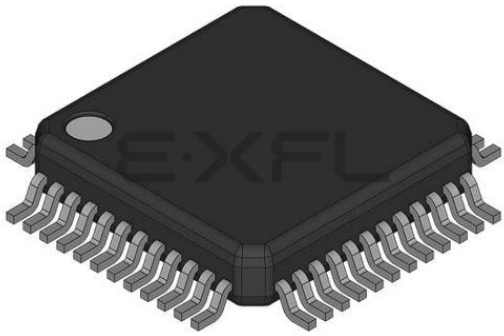

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	S08
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	LVD, POR, PWM, WDT
Number of I/O	41
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	External
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc9s08pt32vlf

Chapter 4
Memory map

4.1 Memory map

The HCS08 core processor can address 64 KB of memory space. The memory map, shown in the following figure, includes:

- User flash memory (flash)
  - MC9S08PT60: 60,864 bytes; 119 pages of 512 bytes each
  - MC9S08PT32: 32,768 bytes; 64 pages of 512 bytes each
- Random-access memory (RAM)
  - MC9S08PT60: 4,096 bytes
  - MC9S08PT32: 4,096 bytes
- Electrically erasable programmable read-only memory (EEPROM)
  - MC9S08PT60: 256 bytes; 128 pages of 2 bytes each
  - MC9S08PT32: 256 bytes; 128 pages of 2 bytes each
- Direct-page registers (0x0000 through 0x003F)
- High-page registers (0x3000 through 0x30FF)

Default protection settings as well as security information that allows the MCU to restrict access to the flash module are stored in the flash configuration field as described in the table below.

Table 4-11. Flash configuration field

Global address	Size (Bytes)	Description
0xFF70 — 0xFF77[1]	8	Backdoor comparison key. See Verify backdoor access key command and Unsecuring the MCU using backdoor key access.
0xFF78 — 0xFF7B	4	Reserved
0xFF7C[1]	1	Flash protection byte
0xFF7D[1]	1	EEPROM protection byte
0xFF7E[1]	1	Flash nonvolatile byte
0xFF7F[1]	1	Flash security byte

1. 0xFF78–0xFF7F for a flash phrase and must be programmed in a single command write sequence. Each byte in the 0xFF78-0xFF7B reserved field must be programmed to 0xFF.

The flash and EEPROM module provides protection to the MCU. During the reset sequence, the FPROT register is loaded with the contents of the flash protection byte in the flash configuration field at global address 0xFF7C in flash memory. The protection functions depend on the configuration of bit settings in FPORT register.

Table 4-12. Flash protection function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No flash protection
1	1	0	Protected low range
1	0	1	Protected high range
1	0	0	Protected high and low ranges
0	1	1	Full p-flash memory protected
0	1	0	Unprotected low range
0	0	1	Unprotected high range
0	0	0	Unprotected high and low ranges

1. For range sizes, see Table 4 and Table 5

The flash protection scheme can be used by applications requiring reprogramming in single chip mode while providing as much protection as possible if reprogramming is not required.

the flash and EEPROM memory, the FSEC[SEC] bits will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, flash memory and EEPROM memory will not be available for read access and 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, the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the verify backdoor access key command as explained in Verify backdoor access key command.
2. If the verify backdoor access key command is successful, the MCU is unsecured and the FSEC[SEC] bits are forced to the unsecure state of 10.

The verify backdoor access key command is monitored by the memory controller and an illegal key will prohibit future use of the verify backdoor access key command. A reset of the MCU is the only method to re-enable the verify backdoor access key command. The security as defined in the flash and EEPROM security byte (0xFF7F) is not changed by using the verify backdoor access key command sequence. The backdoor keys stored in addresses 0xFF70–0xFF77 are unaffected by the verify backdoor access key command sequence. The verify backdoor access key command sequence has no effect on the program and erase protections defined in the flash and EEPROM protection register, FPORT.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the flash and EEPROM security byte can be erased and the flash and EEPROM security byte can be reprogrammed 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 0xFF70–0xFF77 in the flash configuration field.

4.5.2.7.2 Unsecuring the MCU using BDM

A secured MCU can be unsecured by using the following method to erase the flash and EEPROM memory:

1. Reset the MCU.
2. Set FCDIV register as described in Writing the FCLKDIV register.

Table 4-25. Erase verify flash section command error handling (continued)

Register	Error bit	Error condition
		Set if command not available in current mode (see Table 4-9)
		Set if an invalid global address [23:0] is supplied (see Table 4-6)[1]
		Set if a misaligned long words address is supplied (global address[1:0] != 00)
		Set if the requested section crosses flash address boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read[2] or if blank check failed
	MGSTAT0	Set if any non-correctable errors have been encountered during the read[2] or if blank check failed

1. As defined by the memory map for NVM
2. As found in the memory map for NVM

4.5.2.9.4 Read once command

The read once command provides read access to a reserved 64 byte field (8 phrase) located in the nonvolatile information register of flash. The read once field can only be programmed once and can not be erased. It can be used to store the product ID or any other information that can be written only once. It is programmed using the program once command described in Program once command. To avoid code runaway, the read once command must not be executed from the flash block containing the program once reserved field.

Table 4-26. Read once command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x04	Not required
001	Read once phrase index (0x0000 – 0x0007)	
010	Read once word 0 value	
011	Read once word 1 value	
100	Read once word 2 value	
101	Read once word 3 value	

Upon clearing FSTAT[CCIF] to launch the read once command, a read once phrase is fetched and stored in the FCCOB indexed register. The FSTAT[CCIF] flag will set after the read once operation has completed. Valid phrase index values for the read once command range from 0x0000 to 0x0007. During execution of the read once command, any attempt to read addresses within flash block will return invalid data.

4.5.2.9.8 Erase flash block command

The erase flash block operation will erase all addresses in a flash or EEPROM block.

Table 4-34. Erase flash block command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x09	Global address [23:16] to identify flash block[1]
001	Global address[15:0] in flash block to be erased	

1. Global address [23] selects between flash (0) or EEPROM (1) block, that can otherwise eventually share the same address on the MCU global memory map.

Upon clearing FSTAT[CCIF] to launch the erase flash block command, the memory controller will erase the selected flash block and verify that it is erased. The FSTAT[CCIF] flag will set after the erase flash block operation has completed.

Table 4-35. Erase flash block command error handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 4-9)
		Set if an invalid global address [23:16] is supplied[1]
	FPVIOL	Set if an area of the selected flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation[2]
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[2]

1. As defined by the memory map for NVM
2. As found in the memory map for NVM

4.5.2.9.9 Erase flash sector command

The erase flash sector operation will erase all addresses in a flash sector.

Table 4-36. Erase flash sector command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x0A	Global address [23:16] to identify flash block to be erased
001	Global address [15:0] anywhere within the sector to be erased. Refer to Overview for the flash sector size	

Upon clearing FSTAT[CCIF] to launch the erase flash sector command, the memory controller will erase the selected flash sector and then verify that it is erased. The FSTAT[CCIF] flag will be set after the erase flash sector operation has completed.

4.5.2.9.11 Verify backdoor access key command

The verify backdoor access key command will execute only if it is enabled by the NVM_FSEC[KEYEN] bits. The verify backdoor access key command releases security if user-supplied keys match those stored in the flash security bytes of the flash configuration field. See Table 4-6 for details. The code that performs verifying backdoor access command must be running from RAM or EEPROM.

Table 4-40. Verify backdoor access key command FCCOB requirements

CCOBIX[2:0]	NVM_FCCOBHI parameters	NVM_FCCOBLO parameters
000	0x0C	Not required
001	Key 0	
010	Key 1	
011	Key 2	
100	Key 3	

Upon clearing NVM_FSTAT[CCIF] to launch the verify backdoor access key command, the memory controller will check the NVM_FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the memory controller sets the NVM_FSTAT[ACCERR] bit. If the command is enabled, the memory controller compares the key provided in FCCOB to the backdoor comparison key in the flash configuration field with Key 0 compared to 0xFF70, and so on. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the verify backdoor access key command are aborted (set NVM_FSTAT[ACCERR]) until a reset occurs. The NVM_FSTAT[CCIF] flag is set after the verify backdoor access key operation has completed.

Table 4-41. Verify backdoor access key command error handling

Register	Error bit	Error condition
NVM_FSTAT	ACCERR	Set if CCOBIX[2:0] ≠ 100 at command launch
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] ≠ 10
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

Table 4-46. Erase verify EEPROM section command error handling (continued)

Register	Error bit	Error condition
		Set if an invalid global address [23:0] is supplied
		Set if the requested section breaches the end of the EEPROM block
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read or if blank check failed
	MGSTAT0	Set if any non-correctable errors have been encountered during the read or it blank check failed.

4.5.2.9.14 Program EEPROM command

The program EEPROM operation programs one to four previously erased bytes in the EEPROM block. The program EEPROM operation will confirm that the targeted location(s) were successfully programmed upon completion.

Note

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

Table 4-47. Program EEPROM command FCCOB requirements

CCOBIX[2:0]	NVM_FCCOBHI parameters	NVM_FCCOBLO parameters
000	0x11	Global address [23:16] to identify the EEPROM block
001	Global address [15:0] of the first word to be verified	
010		Byte 0 program value
011		Byte 1 program value, if desired
100		Byte 2 program value, if desired
101		Byte 3 program value, if desired

Upon clearing NVM_FSTAT[CCIF] to launch the program EEPROM command, the user-supplied words will be transferred to the memory controller and be programmed if the area is unprotected. The CCOBIX index value at program EEPROM command launch determines how many bytes will be programmed in the EEPROM block. The NVM_FSTAT[CCIF] flag is set when the operation has completed.

Table 4-48. Program EEPROM command error handling

Register	Error Bit	Error condition
NVM_FSTAT	ACCERR	Set if CCOBIX[2:0] < 010 at command launch
		Set if CCOBIX[2:0] >101 at command launch

Table continues on the next page...

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow, when the value of CNTINH:L is loaded into the FTM counter. Additionally, it is forced low at the channel (n) match, when the FTM counter = CnVH:L. See the following figure.

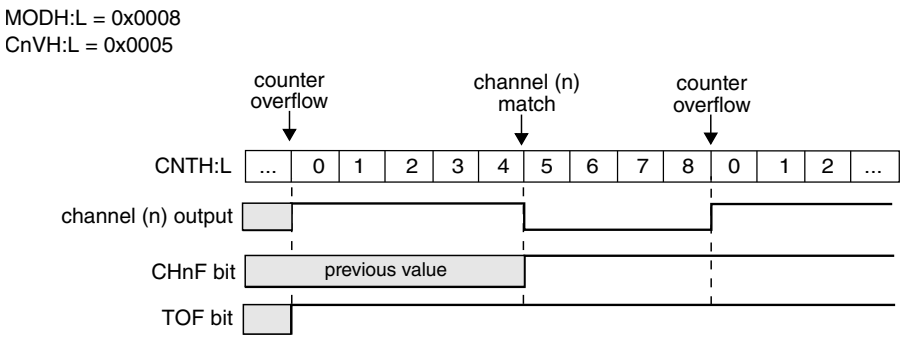

Figure 12-200. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow, when the value of CNTINH:L is loaded into the FTM counter. Additionally, it is forced high at the channel (n) match, when the FTM counter = CnVH:L. See the following figure.

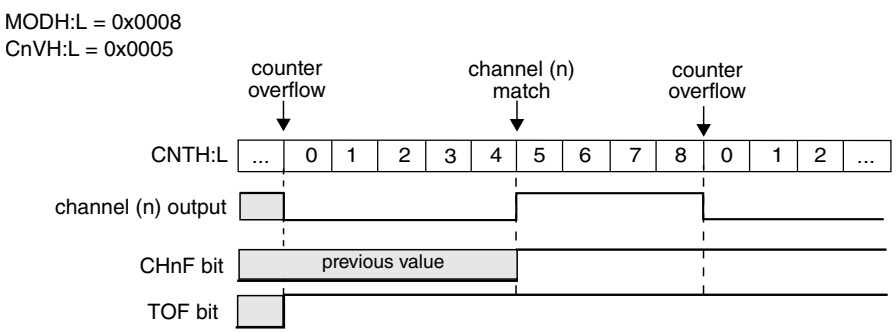

Figure 12-201. EPWM signal with ELSnB:ELSnA = X:1

If (CnVH:L = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHnF bit is not set, even when there is the channel (n) match. If (CnVH:L > MODH:L), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set, even when there is the channel (n) match. Therefore, MODH:MODL must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

Note

- EPWM mode is available only with (CNTINH:L = 0x0000).
- EPWM mode with (CNTINH:L ≠ 0x0000) is not recommended and its results are not guaranteed.

If CHnOM = 1, then the channel (n) output is forced to its inactive state, defined by the POLn bit in register POL. If CHnOM = 0, then the channel (n) output is unaffected by the output mask function.

When a CHnOM bit is cleared, the channel (n) output is enabled. See the following figure.

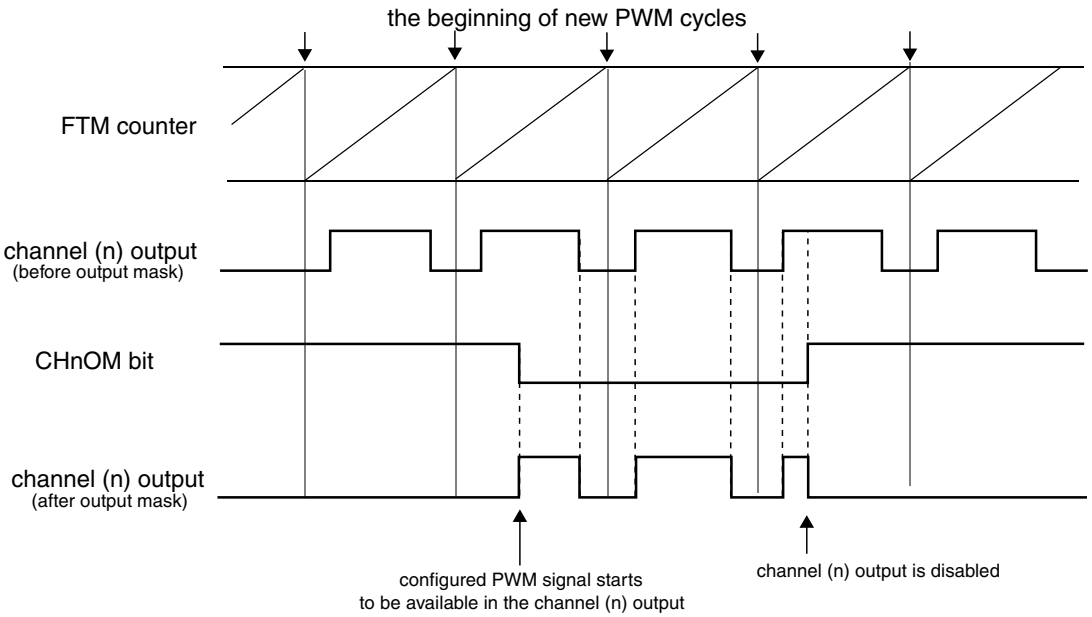

Figure 12-241. Output mask

The following table shows the output mask result before the polarity control.

Table 12-247. Output mask result for channel (n) before the polarity control

CHnOM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

Note

Output mask is available only in combine mode.

12.4.14 Fault control

The fault control is enabled if (FTMEN = 1) and (FAULTM[1:0] ≠ 0:0).

SCIx_BDH field descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt Enable (for LBKDIF) 0 Hardware interrupts from SCI_S2[LBKDIF] disabled (use polling). 1 Hardware interrupt requested when SCI_S2[LBKDIF] flag is 1.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable (for RXEDGIF) 0 Hardware interrupts from SCI_S2[RXEDGIF] disabled (use polling). 1 Hardware interrupt requested when SCI_S2[RXEDGIF] flag is 1.
5 SBNS	Stop Bit Number Select SBNS determines whether data characters are one or two stop bits. **NOTE:** For SCI2, this field is masked. 0 One stop bit. 1 Two stop bit.
SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR is cleared, the SCI baud rate generator is disabled to reduce supply current. When BR is 1 - 8191, the SCI baud rate equals BUSCLK/(16×BR).

15.3.2 SCI Baud Rate Register: Low (SCIx_BDL)

This register, along with SCI_BDH, control the prescale divisor for SCI baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to SCI_BDH to buffer the high half of the new value and then write to SCI_BDL. The working value in SCI_BDH does not change until SCI_BDL is written.

SCI_BDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled; that is, SCI_C2[RE] or SCI_C2[TE] bits are written to 1.

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read Write	SBR							
Reset	0	0	0	0	0	1	0	0

SCIx_BDL field descriptions

Field	Description
SBR	Baud Rate Modulo Divisor These 13 bits in SBR[12:0] are referred to collectively as BR. They set the modulo divide rate for the SCI baud rate generator. When BR is cleared, the SCI baud rate generator is disabled to reduce supply current. When BR is 1 - 8191, the SCI baud rate equals BUSCLK/(16×BR).

SCI_D[RT3], SCI_D[RT5], and SCI_D[RT7] are 0 even if one or all of the samples taken at SCI_D[RT8], SCI_D[RT9], and SCI_D[RT10] are 1s. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (SCI_S1[NF]) is set when the received character is transferred to the receive data buffer.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times. This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

In the case of a framing error, provided the received character was not a break character, the sampling logic that searches for a falling edge is filled with three logic 1 samples so that a new start bit can be detected almost immediately.

In the case of a framing error, the receiver is inhibited from receiving any new characters until the framing error flag is cleared. The receive shift register continues to function, but a complete character cannot transfer to the receive data buffer if SCI_S1[FE] remains set.

15.4.3.2 Receiver wake-up operation

Receiver wake-up is a hardware mechanism that allows an SCI receiver to ignore the characters in a message intended for a different SCI receiver. In such a system, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control field (SCI_C2[RWU]). When SCI_C2[RWU] is set, the status flags associated with the receiver, (with the exception of the idle bit, IDLE, when SCI_S2[RWUID] is set), are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force SCI_C2[RWU] to 0, so all receivers wake up in time to look at the first character(s) of the next message.

15.4.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, SCI_C2[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The SCI_C1[M] control field selects 8-bit or 9-bit data mode and SCI_BDH[SBNS] selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 or 11 or 12 bit times because of the start and stop bits.

SPI0_S field descriptions (continued)

Field	Description
5 SPTEF	SPI transmit buffer empty flag This bit is set when the transmit data buffer is empty. SPTEF is cleared by reading the S register with SPTEF set and then writing a data value to the transmit buffer at D. The S register must be read with SPTEF set to 1 before writing data to the D register; otherwise, the D write is ignored. SPTEF is automatically set when all data from the transmit buffer transfers into the transmit shift register. For an idle SPI, data written to D is transferred to the shifter almost immediately so that SPTEF is set within two bus cycles, allowing a second set of data to be queued into the transmit buffer. After completion of the transfer of the data in the shift register, the queued data from the transmit buffer automatically moves to the shifter, and SPTEF is set to indicate that room exists for new data in the transmit buffer. If no new data is waiting in the transmit buffer, SPTEF simply remains set and no data moves from the buffer to the shifter. If a transfer does not stop, the last data that was transmitted is sent out again. 0 SPI transmit buffer not empty 1 SPI transmit buffer empty
4 MODF	Master mode fault flag MODF is set if the SPI is configured as a master and the slave select input goes low, indicating some other SPI device is also configured as a master. The $\overline{SS}$ pin acts as a mode fault error input only when MSTR is 1, MODFEN is 1, and SSOE is 0; otherwise, MODF will never be set. MODF is cleared by reading MODF while it is 1 and then writing to the SPI control register 1 (C1). 0 No mode fault error 1 Mode fault error detected
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

16.3.5 SPI data register (SPIx_D)

This register is both the input and output register for SPI data. A write to the register writes to the transmit data buffer, allowing data to be queued and transmitted.

When the SPI is configured as a master, data queued in the transmit data buffer is transmitted immediately after the previous transmission has completed.

The SPTEF bit in the S register indicates when the transmit data buffer is ready to accept new data. The S register must be read when SPTEF is set before writing to the SPI data register; otherwise, the write is ignored.

Data may be read from the SPI data register any time after SPRF is set and before another transfer is finished. Failure to read the data out of the receive data buffer before a new transfer ends causes a receive overrun condition, and the data from the new transfer is lost. The new data is lost because the receive buffer still held the previous character and was not ready to accept the new data. There is no indication for a receive overrun condition, so the application system designer must ensure that previous data has been read from the receive buffer before a new transfer is initiated.

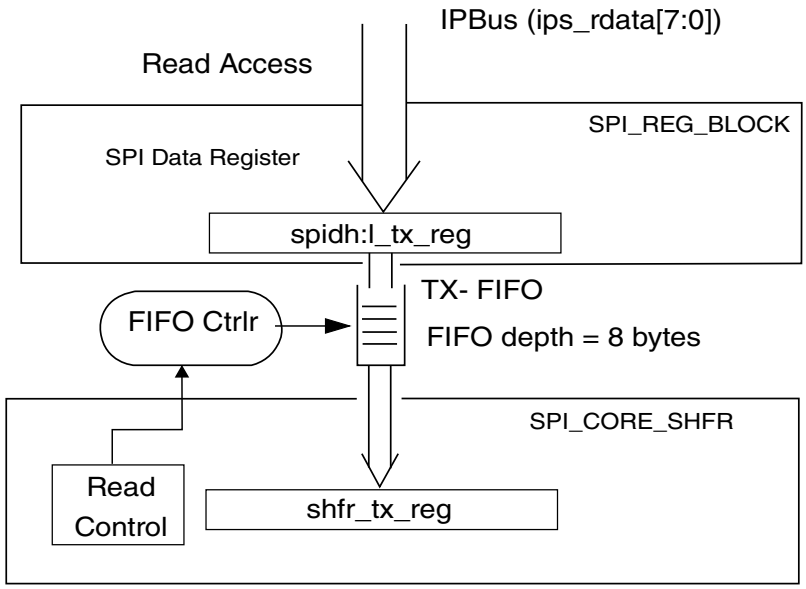

Figure 17-24. SPIH:L write side structural overview in FIFO mode

17.4.5 Data Transmission Length

The SPI can support data lengths of 8 or 16 bits. The length can be configured with the SPIMODE bit in the SPIx_C2 register.

In 8-bit mode (SPIMODE = 0), the SPI Data Register is comprised of one byte: SPIx_DL. The SPI Match Register is also comprised of only one byte: SPIx_ML. Reads of SPIx_DH and SPIx_MH will return zero. Writes to SPIx_DH and SPIx_MH will be ignored.

In 16-bit mode (SPIMODE = 1), the SPI Data Register is comprised of two bytes: SPIx_DH and SPIx_DL. Reading either byte (SPIx_DH or SPIx_DL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. Writing to either byte (SPIx_DH or SPIx_DL) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the transmit data buffer.

In 16-bit mode, the SPI Match Register is also comprised of two bytes: SPIx_MH and SPIx_ML. There is no buffer mechanism for the reading of SPIxMH and SPIxML since they can only be changed by writing at CPU side. Writing to either byte (SPIx_MH or SPIx_ML) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the SPI Match Register.

Any switching between 8- and 16-bit data transmission length (controlled by SPIMODE bit) in master mode will abort a transmission in progress, force the SPI system into idle state, and reset all status bits in the SPIx_S register. To initiate a transfer after writing to

After a repeated START condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them are addressed because R/$\overline{W}$ = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

Table 18-17. Master-receiver addresses a slave-transmitter with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Sr	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 1	A3	Data	A	...	Data	A	P

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

18.4.3 Address matching

All received addresses can be requested in 7-bit or 10-bit address format.

- AD[7:1] in Address Register 1, which contains the I2C primary slave address, always participates in the address matching process. It provides a 7-bit address.
- If the ADEXT bit is set, AD[10:8] in Control Register 2 participates in the address matching process. It extends the I2C primary slave address to a 10-bit address.

Additional conditions that affect address matching include:

- If the GCAEN bit is set, general call participates the address matching process.
- If the ALERTEN bit is set, alert response participates the address matching process.
- If the SIICAEN bit is set, Address Register 2 participates in the address matching process.
- If the RMEN bit is set, when the Range Address register is programmed to a nonzero value, any address within the range of values of Address Register 1 (excluded) and the Range Address register (included) participates in the address matching process. The Range Address register must be programmed to a value greater than the value of Address Register 1.

When the I2C module responds to one of these addresses, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the Data register after the first byte transfer to determine that the address is matched.

21.3.3 TSI Control and Status Register 2 (TSI_CS2)

Address: 8h base + 2h offset = Ah

Bit	7	6	5	4	3	2	1	0
Read Write	REFCHRG			DVOLT		EXTCHRG		
Reset	0	0	0	0	0	0	0	0

TSI_CS2 field descriptions

Field	Description
7–5 REFCHRG	REFCHRG These bits indicate the reference oscillator charge and discharge current value. 000 500 nA. 001 1 µ A. 010 2 µ A. 011 4 µ A. 100 8 µ A. 101 16 µ A. 110 32 µ A. 111 64 µ A.
4–3 DVOLT	DVOLT These bits indicate the oscillator's voltage rails as below. 00 $\Delta V = 1.03$ V; $V_P = 1.33$ V; $V_m = 0.30$ V. 01 $\Delta V = 0.73$ V; $V_P = 1.18$ V; $V_m = 0.45$ V. 10 $\Delta V = 0.43$ V; $V_P = 1.03$ V; $V_m = 0.60$ V. 11 $\Delta V = 0.29$ V; $V_P = 0.95$ V; $V_m = 0.67$ V.
EXTCHRG	EXTCHRG These bits indicate the electrode oscillator charge and discharge current value. 000 500 nA. 001 1 µ A. 010 2 µ A. 011 4 µ A. 100 8 µ A. 101 16 µ A. 110 32 µ A. 111 64 µ A.

$$F_{ref\_osc} = \frac{I_{ref}}{2 * C_{ref} * \Delta V}$$

Equation 3. TSI reference oscillator frequency

Where:

C_{ref}: Internal reference capacitor

I_{ref}: Reference oscillator current source

ΔV : Hysteresis delta voltage

Considering C_{ref} = 1.0 pF, I_{ref} = 16 µA and ΔV = 1.03 V, follows

$$F_{ref\_osc} = \frac{16\mu A}{2 * 1.0pF * 1.03V} = 7.8MHz$$

21.4.2 TSI measurement result

The capacitance measurement result is defined by the number of TSI reference oscillator periods during the sample time and is stored in the TSICHnCNT register.

TSICHnCNT = T_{cap_samp} * F_{ref_osc}

Using Equation 2 and Equation 1 follows:

$$TSICHnCNT = \frac{I_{ref} * PS * NSCN}{C_{ref} * I_{elec}} * C_{elec}$$

Equation 4. Capacitance result value

In the example where F_{ref_osc} = 7.8 MHz and T_{cap_samp} = 82.4 µs, TSICHnCNT = 640

21.4.3 Enable TSI module

The TSI module can be fully functional in run, wait and stop3 modes. The TSI_CS0[TSIEN] bit must be set to enable the TSI module in run and wait mode. When TSI_CS0[STPE] bit is set, it allows the TSI module to work in stop3 mode.

NOTE

To get better performance, set other channels as GPIO output mode and output 0 when scanning a channel.

2. Optional to enable reverse and complement function. Please see Bit reverse and Result complement for details.
3. Write 32-bit polynomial to CRC_P0:CRC_P3.
4. Set CRC_CTRL[WAS] bit to allow CRC_D0:CRC_D3 written by seed.
5. Write 32-bit seed to CRC_D0:CRC_D3.
6. Clear CRC_CTRL[WAS] bit to start 32-bit CRC calculation.
7. Dummy CRC_D3 with 8-bit CRC raw data.
8. Get the checksum from CRC_D0:CRC_D3 when all CRC raw data dummied.

22.6.3 Bit reverse

The bit reverse function allows the input and output data reversed by bit for different CRC standard and endian systems. The CRC_CTRL[TOT] bits control the reverse of input data and the CRC_CTRL[TOTR] bits control the reverse of output data. The following table shows how the CRC_CTRL[TOT] and CRC_CTRL[TOTR] bits work.

Table 22-11. TOT and TOTR bit and byte reverse function

TOT ROW	D0	D1	D2	D3
00	b31b30b29b28b27b26b25b24	b23b22b21b20b19b18b17b16	b15b14b13b12b11b10b9b8	b7b6b5b4b3b2b1b0
01	b24b25b26b27b28b29b30b31	b16b17b18b19b20b21b22b23	b8b9b10b11b12b13b14b15	b0b1b2b3b4b5b6b7

NOTE

00 is the default case that no bit is reversed.

22.6.4 Result complement

The result complement function allows to output the complement of the checksum in CRC data registers. When CRC_CTRL[FXOR] bit is set, the checksum is read by its complement. Otherwise, the raw checksum is accessed.

22.6.5 CCITT compliant CRC example

The following code segment shows CCITT CRC-16 compliant example.

Example: 22.6.5.1 CCITT CRC-16 compliant example

Chapter 24
Development support

24.1 Introduction

This chapter describes the single-wire background debug mode (BDM), which uses the on-chip background debug controller (BDC) module, and the independent on-chip real-time in-circuit emulation (ICE) system, which uses the on-chip debug (DBG) module.

24.1.1 Forcing active background

The method for forcing active background mode depends on the specific HCS08 derivative. For the 9S08xxxx, you can force active background after a power-on reset by holding the BKGD pin low as the device exits the reset condition. You can also force active background by driving BKGD low immediately after a serial background command that writes a one to the BDFR bit in the SBDFR register. Other causes of reset including an external pin reset or an internally generated error reset ignore the state of the BKGD pin and reset into normal user mode. If no debug pod is connected to the BKGD pin, the MCU will always reset into normal operating mode.

24.1.2 Features

Features of the BDC module include:

- Single pin for mode selection and background communications
- BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands for memory access
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from stop or wait modes
- One hardware address breakpoint built into BDC

DBG_FH field descriptions

Field	Description
F[15:8]	FIFO High Data Bits The FIFO High data bits provide access to bits [15:8] of data in the FIFO. This register is not used in event only modes and will read a $00 for valid FIFO words.

25.3.8 Debug FIFO Low Register (DBG_FL)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 3010h base + 7h offset = 3017h

Bit	7	6	5	4	3	2	1	0
Read	F[7:0]							
Write								
Reset	0	0	0	0	0	0	0	0

DBG_FL field descriptions

Field	Description
F[7:0]	FIFO Low Data Bits The FIFO Low data bits contain the least significant byte of data in the FIFO. When reading FIFO words, read DBGFX and DBGFH before reading DBGFL because reading DBGFL causes the FIFO pointers to advance to the next FIFO location. In event-only modes, there is no useful information in DBGFX and DBGFH so it is not necessary to read them before reading DBGFL.