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	HCS12X
Core Size	16-Bit
Speed	40MHz
Connectivity	CANbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	44
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	1.72V ~ 5.5V
Data Converters	A/D 8x12b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	64-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/s9s12xs128j1maer

2.3.63 Port J Polarity Select Register (PPSJ)

Address 0x026D Access: User read/write[1]

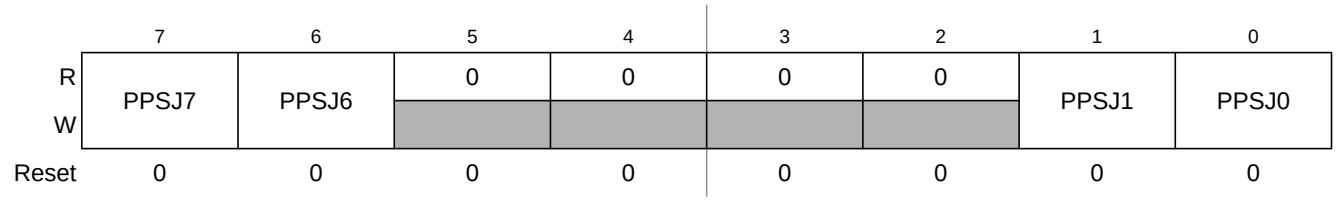


Figure 2-61. Port J Polarity Select Register (PPSJ)

[1] Read: Anytime
Write: Anytime

Table 2-60. PPSJ Register Field Descriptions

Field	Description
7-6, 1-0 PPSJ	**Port J pull device select**—Configure pull device and pin interrupt edge polarity on input pin This bit selects a pull-up or a pull-down device if enabled on the associated port input pin. This bit also selects the polarity of the active pin interrupt edge. 1 A pull-down device selected; rising edge selected 0 A pull-up device selected; falling edge selected

2.3.64 Port J Interrupt Enable Register (PIEJ)

Address 0x026E Access: User read/write[1]

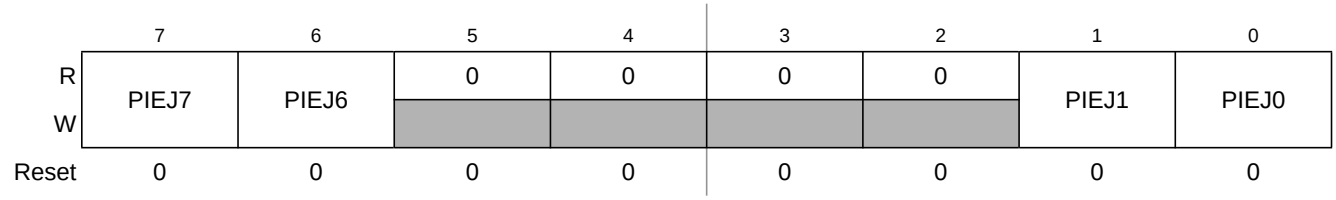


Figure 2-62. Port J Interrupt Enable Register (PIEJ)

[1] Read: Anytime
Write: Anytime

Table 2-61. PIEJ Register Field Descriptions

Field	Description
7-6, 1-0 PIEJ	**Port J interrupt enable**— This bit enables or disables on the edge sensitive pin interrupt on the associated pin. 1 Interrupt enabled 0 Interrupt disabled (interrupt flag masked)

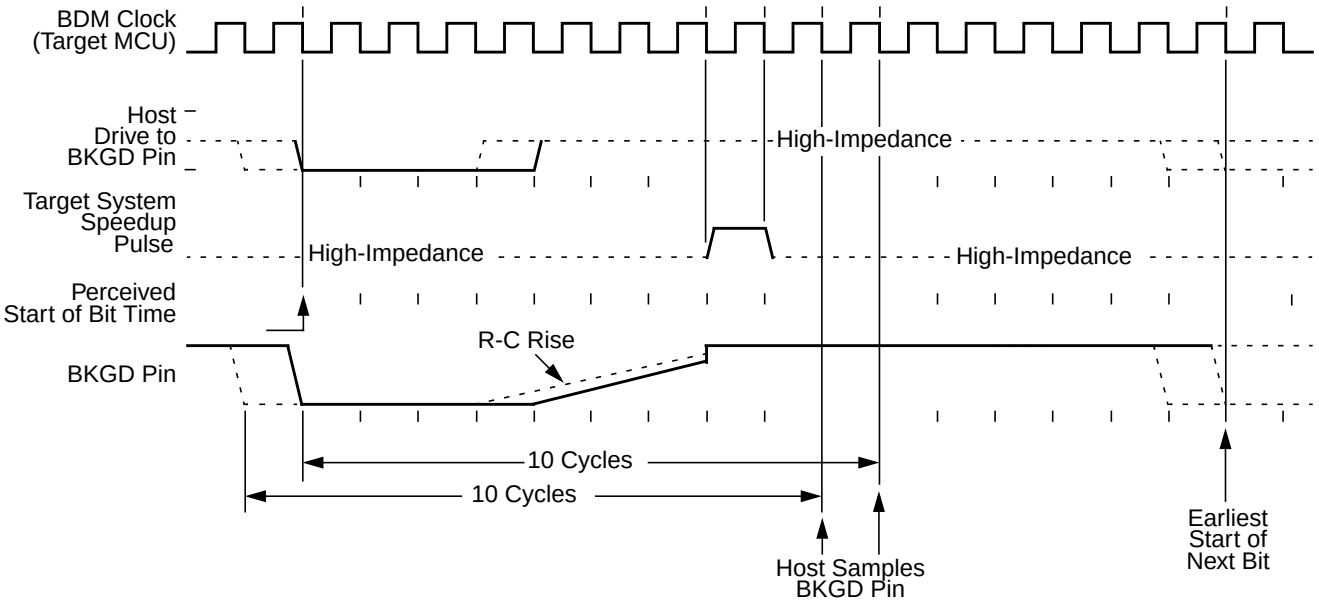


Figure 5-9. BDM Target-to-Host Serial Bit Timing (Logic 1)

The ACK_ENABLE sends an ACK pulse when the command has been completed. This feature could be used by the host to evaluate if the target supports the hardware handshake protocol. If an ACK pulse is issued in response to this command, the host knows that the target supports the hardware handshake protocol. If the target does not support the hardware handshake protocol the ACK pulse is not issued. In this case, the ACK_ENABLE command is ignored by the target since it is not recognized as a valid command.

The BACKGROUND command will issue an ACK pulse when the CPU changes from normal to background mode. The ACK pulse related to this command could be aborted using the SYNC command.

The GO command will issue an ACK pulse when the CPU exits from background mode. The ACK pulse related to this command could be aborted using the SYNC command.

The GO_UNTIL command is equivalent to a GO command with exception that the ACK pulse, in this case, is issued when the CPU enters into background mode. This command is an alternative to the GO command and should be used when the host wants to trace if a breakpoint match occurs and causes the CPU to enter active background mode. Note that the ACK is issued whenever the CPU enters BDM, which could be caused by a breakpoint match or by a BGND instruction being executed. The ACK pulse related to this command could be aborted using the SYNC command.

The TRACE1 command has the related ACK pulse issued when the CPU enters background active mode after one instruction of the application program is executed. The ACK pulse related to this command could be aborted using the SYNC command.

5.4.9 SYNC — Request Timed Reference Pulse

The SYNC command is unlike other BDM commands because the host does not necessarily know the correct communication speed to use for BDM communications until after it has analyzed the response to the SYNC command. To issue a SYNC command, the host should perform the following steps:

1. Drive the BKGD pin low for at least 128 cycles at the lowest possible BDM serial communication frequency (the lowest serial communication frequency is determined by the crystal oscillator or the clock chosen by CLKSW.)
2. Drive BKGD high for a brief speedup pulse to get a fast rise time (this speedup pulse is typically one cycle of the host clock.)
3. Remove all drive to the BKGD pin so it reverts to high impedance.
4. Listen to the BKGD pin for the sync response pulse.

Upon detecting the SYNC request from the host, the target performs the following steps:

1. Discards any incomplete command received or bit retrieved.
2. Waits for BKGD to return to a logic one.
3. Delays 16 cycles to allow the host to stop driving the high speedup pulse.
4. Drives BKGD low for 128 cycles at the current BDM serial communication frequency.
5. Drives a one-cycle high speedup pulse to force a fast rise time on BKGD.
6. Removes all drive to the BKGD pin so it reverts to high impedance.

The host measures the low time of this 128 cycle SYNC response pulse and determines the correct speed for subsequent BDM communications. Typically, the host can determine the correct communication speed

Table 6-44. Breakpoint Mapping Summary

1	1	0	X	Breakpoint to SWI
1	1	1	0	Breakpoint to BDM
1	1	1	1	No Breakpoint

BDM cannot be entered from a breakpoint unless the ENABLE bit is set in the BDM. If entry to BDM via a BGND instruction is attempted and the ENABLE bit in the BDM is cleared, the CPU12X actually executes the BDM firmware code. It checks the ENABLE and returns if ENABLE is not set. If not serviced by the monitor then the breakpoint is re-asserted when the BDM returns to normal CPU12X flow.

If the comparator register contents coincide with the SWI/BDM vector address then an SWI in user code and DBG breakpoint could occur simultaneously. The CPU12X ensures that BDM requests have a higher priority than SWI requests. Returning from the BDM/SWI service routine care must be taken to avoid re triggering a breakpoint.

NOTE

When program control returns from a tagged breakpoint using an RTI or BDM GO command without program counter modification it will return to the instruction whose tag generated the breakpoint. To avoid re triggering a breakpoint at the same location reconfigure the S12XDBG module in the SWI routine, if configured for an SWI breakpoint, or over the BDM interface by executing a TRACE command before the GO to increment the program flow past the tagged instruction.

7.1.5 Unsecuring the Microcontroller

Unsecuring the microcontroller can be done by three different methods:

1. Backdoor key access
2. Reprogramming the security bits
3. Complete memory erase (special modes)

7.1.5.1 Unsecuring the MCU Using the Backdoor Key Access

In normal modes (single chip and expanded), security can be temporarily disabled using the backdoor key access method. This method requires that:

- The backdoor key at 0xFF00–0xFF07 (= global addresses 0x7F_FF00–0x7F_FF07) has been programmed to a valid value.
- The KEYEN[1:0] bits within the Flash options/security byte select ‘enabled’.
- In single chip mode, the application program programmed into the microcontroller must be designed to have the capability to write to the backdoor key locations.

The backdoor key values themselves would not normally be stored within the application data, which means the application program would have to be designed to receive the backdoor key values from an external source (e.g. through a serial port).

The backdoor key access method allows debugging of a secured microcontroller without having to erase the Flash. This is particularly useful for failure analysis.

NOTE

No word of the backdoor key is allowed to have the value 0x0000 or 0xFFFF.

7.1.6 Reprogramming the Security Bits

In normal single chip mode (NS), security can also be disabled by erasing and reprogramming the security bits within Flash options/security byte to the unsecured value. Because the erase operation will erase the entire sector from 0xFE00–0xFFFF (0x7F_FE00–0x7F_FFFF), the backdoor key and the interrupt vectors will also be erased; this method is not recommended for normal single chip mode. The application software can only erase and program the Flash options/security byte if the Flash sector containing the Flash options/security byte is not protected (see Flash protection). Thus Flash protection is a useful means of preventing this method. The microcontroller will enter the unsecured state after the next reset following the programming of the security bits to the unsecured value.

This method requires that:

- The application software previously programmed into the microcontroller has been designed to have the capability to erase and program the Flash options/security byte, or security is first disabled using the backdoor key method, allowing BDM to be used to issue commands to erase and program the Flash options/security byte.
- The Flash sector containing the Flash options/security byte is not protected.

8.4 Functional Description

8.4.1 Functional Blocks

8.4.1.1 Phase Locked Loop with Internal Filter (IPLL)

The IPLL is used to run the MCU from a different time base than the incoming OSCCLK. Figure 8-15 shows a block diagram of the IPLL.

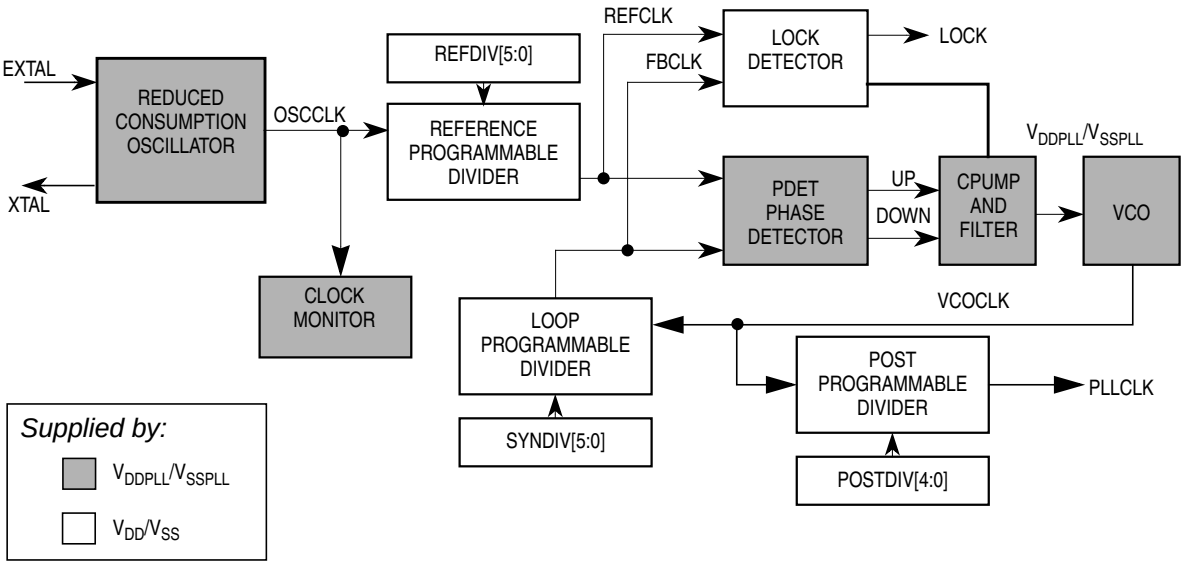


Figure 8-15. IPLL Functional Diagram

For increased flexibility, OSCCLK can be divided in a range of 1 to 64 to generate the reference frequency REFCLK using the REFDIV[5:0] bits. This offers a finer multiplication granularity. Based on the SYNDIV[5:0] bits the IPLL generates the VCOCLK by multiplying the reference clock by a multiple of 2, 4, 6,... 126, 128. Based on the POSTDIV[4:0] bits the VCOCLK can be divided in a range of 1,2,4,6,8,... to 62 to generate the PLLCLK.

$$f_{PLL} = 2 \times f_{OSC} \times \frac{SYNDIV + 1}{[REFDIV + 1][2 \times POSTDIV]}$$

NOTE

Although it is possible to set the dividers to command a very high clock frequency, do not exceed the specified bus frequency limit for the MCU.

If (PLLSEL = 1) then $f_{BUS} = f_{PLL} / 2$.

IF POSTDIV = $00 the f_{PLL} is identical to f_{VCO} (divide by one)

Several examples of IPLL divider settings are shown in Table 8-14. Shaded rows indicated that these settings are not recommended. The following rules help to achieve optimum stability and shortest lock time:

- Use lowest possible f_{VCO} / f_{REF} ratio (SYNDIV value).
- Use highest possible REFCLK frequency f_{REF}.

from the EXTAL input frequency. In full stop mode (PSTP = 0), the EXTAL pin is pulled down by an internal resistor of typical 200 kΩ.

NOTE

Freescale recommends an evaluation of the application board and chosen resonator or crystal by the resonator or crystal supplier.

Loop controlled circuit is not suited for overtone resonators and crystals.

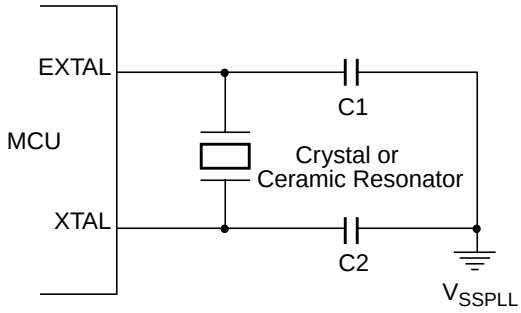
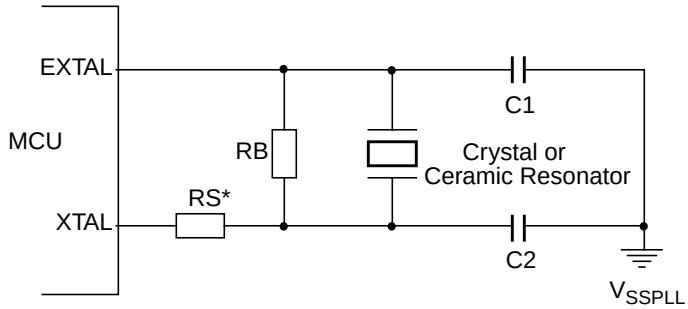


Figure 9-2. Loop Controlled Pierce Oscillator Connections (LCP mode selected)

NOTE

Full swing Pierce circuit is not suited for overtone resonators and crystals without a careful component selection.



* R_S can be zero (shorted) when use with higher frequency crystals. Refer to manufacturer's data.

Figure 9-3. Full Swing Pierce Oscillator Connections (FSP mode selected)

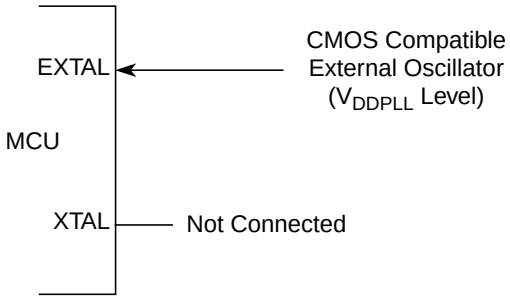


Figure 9-4. External Clock Connections (FSP mode selected)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x001E	ATDDR7	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0020	ATDDR8	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0022	ATDDR9	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0024	ATDDR10	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0026	ATDDR11	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0028	ATDDR12	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x002A	ATDDR13	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x002C	ATDDR14	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x002E	ATDDR15	R W	See Section 10.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 10.3.2.12.2, “Right Justified Result Data (DJM=1)”							

= Unimplemented or Reserved

Figure 10-3. ADC12B16C Register Summary (Sheet 2 of 2)

10.3.2 Register Descriptions

This section describes in address order all the ADC12B16C registers and their individual bits.

10.3.2.1 ATD Control Register 0 (ATDCTL0)

Writes to this register will abort current conversion sequence.

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	Reserved	0	0	0	WRAP3	WRAP2	WRAP1	WRAP0
W								
Reset	0	0	0	0	1	1	1	1

= Unimplemented or Reserved

Figure 10-4. ATD Control Register 0 (ATDCTL0)

Read: Anytime

Write: Anytime, in special modes always write 0 to Reserved Bit 7.

Table 10-2. ATDCTL0 Field Descriptions

Field	Description
3-0 WRAP[3-0]	**Wrap Around Channel Select Bits** — These bits determine the channel for wrap around when doing multi-channel conversions. The coding is summarized in Table 10-3.

Table 10-7. ATDCTL2 Field Descriptions (continued)

Field	Description
4 ETRIGLE	**External Trigger Level/Edge Control** — This bit controls the sensitivity of the external trigger signal. See Table 10-8 for details.
3 ETRIGP	**External Trigger Polarity** — This bit controls the polarity of the external trigger signal. See Table 10-8 for details.
2 ETRIGE	**External Trigger Mode Enable** — This bit enables the external trigger on one of the AD channels or one of the ETRIG3-0 inputs as described in Table 10-6. If external trigger source is one of the AD channels, the digital input buffer of this channel is enabled. The external trigger allows to synchronize the start of conversion with external events. External trigger will not work while converting in stop mode. 0 Disable external trigger 1 Enable external trigger
1 ASCIE	**ATD Sequence Complete Interrupt Enable** 0 ATD Sequence Complete interrupt requests are disabled. 1 ATD Sequence Complete interrupt will be requested whenever SCF=1 is set.
0 ACMPIE	**ATD Compare Interrupt Enable** — If automatic compare is enabled for conversion *n* (CMPE[*n*]=1 in ATDCMPE register) this bit enables the compare interrupt. If the CCF[*n*] flag is set (showing a successful compare for conversion *n*), the compare interrupt is triggered. 0 ATD Compare interrupt requests are disabled. 1 For the conversions in a sequence for which automatic compare is enabled (CMPE[*n*]=1), ATD Compare Interrupt will be requested whenever any of the respective CCF flags is set.

Table 10-8. External Trigger Configurations

ETRIGLE	ETRIGP	External Trigger Sensitivity
0	0	Falling edge
0	1	Rising edge
1	0	Low level
1	1	High level

10.3.2.4 ATD Control Register 3 (ATDCTL3)

Writes to this register will abort current conversion sequence.

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	DJM	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
Reset	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-7. ATD Control Register 3 (ATDCTL3)

Read: Anytime

Write: Anytime

Table 10-10. Examples of ideal decimal ATD Results

Input Signal V_{RL} = 0 Volts V_{RH} = 5.12 Volts	8-Bit Codes (resolution=20mV)	10-Bit Codes (resolution=5mV)	12-Bit Codes (transfer curve has 1.25mV offset) (resolution=1.25mV)
5.120 Volts	255	1023	4095
...	...	...	...
0.022	1	4	17
0.020	1	4	16
0.018	1	4	14
0.016	1	3	12
0.014	1	3	11
0.012	1	2	9
0.010	1	2	8
0.008	0	2	6
0.006	0	1	4
0.004	0	1	3
0.003	0	0	2
0.002	0	0	1
0.000	0	0	0

Table 10-11. Conversion Sequence Length Coding

S8C	S4C	S2C	S1C	Number of Conversions per Sequence
0	0	0	0	16
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
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

Table 10-12. ATD Behavior in Freeze Mode (Breakpoint)

FRZ1	FRZ0	Behavior in Freeze Mode
0	0	Continue conversion
0	1	Reserved
1	0	Finish current conversion, then freeze

Figure 11-24. Receive/Transmit Message Buffer — Extended Identifier Mapping

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

Module Base + 0x00X2

	7	6	5	4	3	2	1	0
R W	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
Reset:	x	x	x	x	x	x	x	x

Figure 11-28. Identifier Register 2 (IDR2) — Extended Identifier Mapping

Table 11-29. IDR2 Register Field Descriptions — Extended

Field	Description
7-0 ID[14:7]	**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.

Module Base + 0x00X3

	7	6	5	4	3	2	1	0
R W	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
Reset:	x	x	x	x	x	x	x	x

Figure 11-29. Identifier Register 3 (IDR3) — Extended Identifier Mapping

Table 11-30. 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

14.3.2.9 SCI Data Registers (SCIDRH, SCIDRL)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	R8	T8	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 14-12. SCI Data Registers (SCIDRH)

Module Base + 0x0007

	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 14-13. SCI Data Registers (SCIDRL)

Read: Anytime; reading accesses SCI receive data register

Write: Anytime; writing accesses SCI transmit data register; writing to R8 has no effect

Table 14-13. SCIDRH and SCIDRL Field Descriptions

Field	Description
SCIDRH 7 R8	**Received Bit 8** — R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1).
SCIDRH 6 T8	**Transmit Bit 8** — T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1).
SCIDRL 7:0 R[7:0] T[7:0]	**R7:R0** — Received bits seven through zero for 9-bit or 8-bit data formats **T7:T0** — Transmit bits seven through zero for 9-bit or 8-bit formats

NOTE

If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten.The same value is transmitted until T8 is rewritten

In 8-bit data format, only SCI data register low (SCIDRL) needs to be accessed.

When transmitting in 9-bit data format and using 8-bit write instructions, write first to SCI data register high (SCIDRH), then SCIDRL.

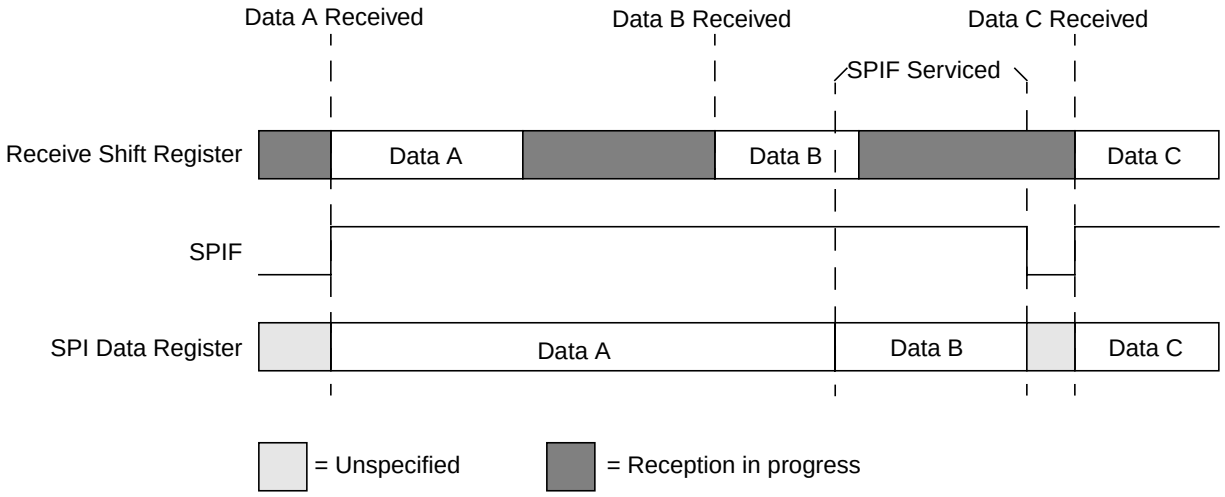


Figure 15-9. Reception with SPIF serviced in Time

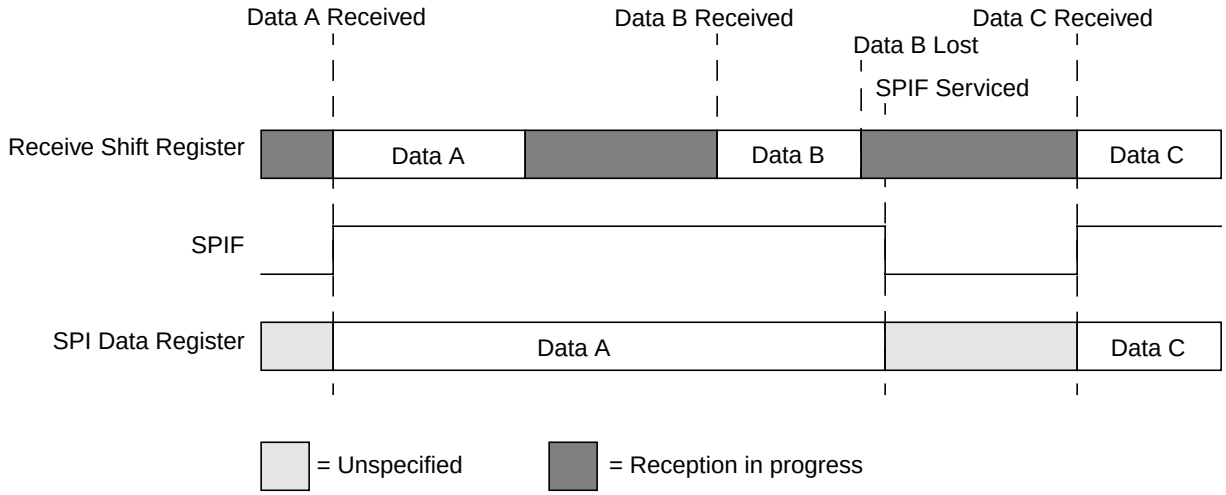


Figure 15-10. Reception with SPIF serviced too late

15.4 Functional Description

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or SPI operation can be interrupt driven.

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI control register 1. While SPE is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select ($\overline{SS}$)
- Serial clock (SCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed.

Table 18-32. Erase Verify All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read

18.4.2.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or D-Flash block has been erased. The FCCOB upper global address bits determine which block must be verified.

Table 18-33. Erase Verify Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x02	Global address [22:16] of the Flash block to be verified.

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or D-Flash block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.

Table 18-34. Erase Verify Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if an invalid global address [22:16] is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read

18.4.2.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases. The section to be verified cannot cross a 256 Kbyte boundary in the P-Flash memory space.

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 P-Flash block will return invalid data.

Table 18-38. Read Once 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 18-28)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read

18.4.2.5 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

CAUTION

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

Table 18-39. Program P-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x06	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed[1]	
010	Word 0 program value	
011	Word 1 program value	
100	Word 2 program value	
101	Word 3 program value	

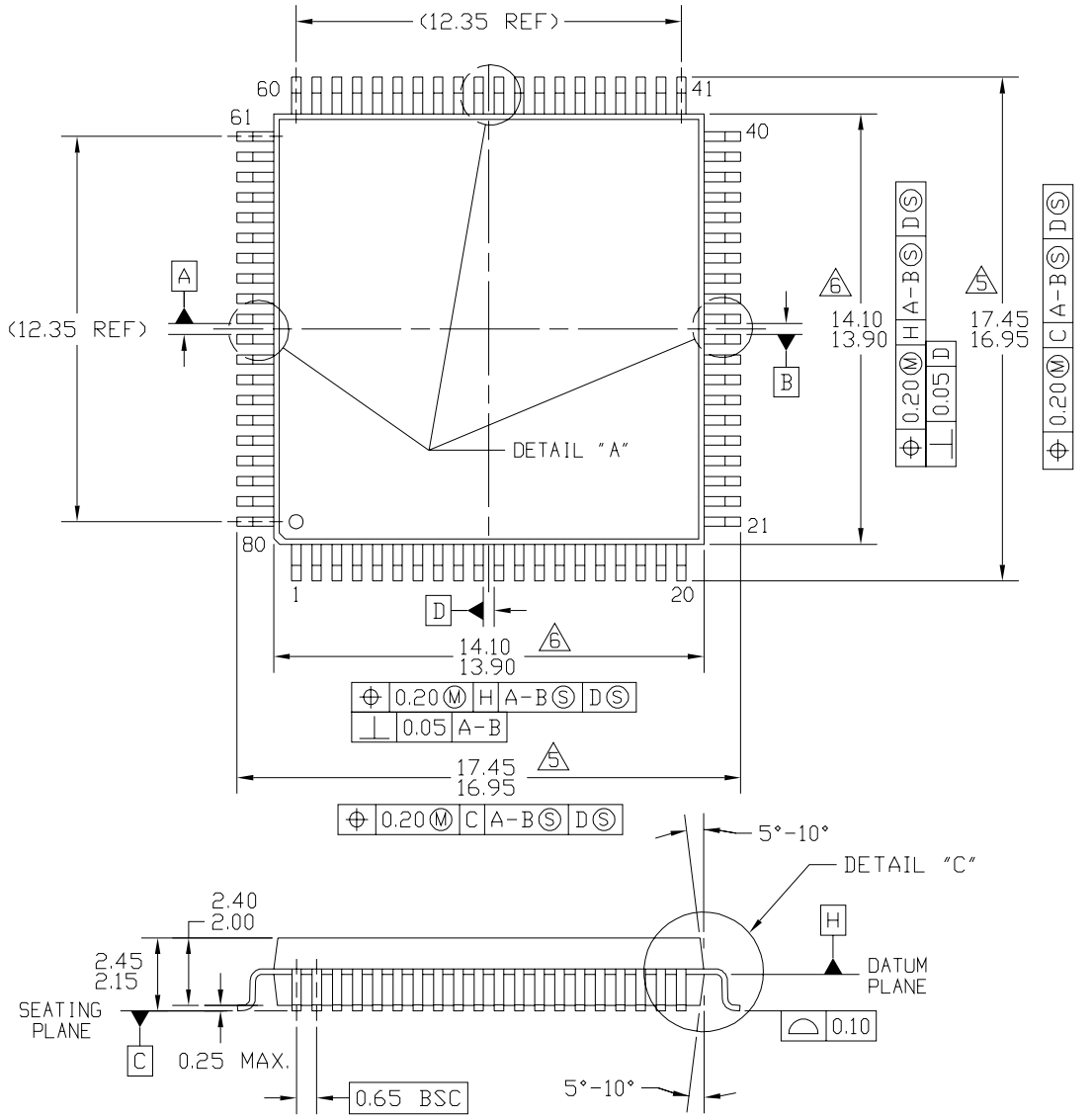
1 Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

A.6.1.5 Pseudo Stop and Wait Recovery

The recovery from pseudo stop and wait is essentially the same since the oscillator is not stopped in both modes. The controller can be woken up by internal or external interrupts. After t_{wrs} the CPU starts fetching the interrupt vector.

B.2 80-Pin QFP Mechanical Dimensions



© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE
TITLE: QUAD FLAT PACKAGE, 80 LEAD, 14 X 14 X 2.2 PKG, 0.65 LEAD PITCH	DOCUMENT NO: 98ASB42846B	REV: C
	CASE NUMBER: 841B-02	20 MAY 2005
	STANDARD: NON-JEDEC	

Figure B-4. 80-pin QFP (case no. 841B) - page 1

C.1.1 112-Pin LQFP Recommended PCB Layout

Figure C-1. 112-Pin LQFP Recommended PCB Layout (Loop Controlled Pierce Oscillator)

C.1.3 64-Pin LQFP Recommended PCB Layout

TBD

Figure C-3. 64-Pin LQFP Recommended PCB Layout (Loop Controlled Pierce Oscillator)