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	S12Z
Core Size	16-Bit
Speed	32MHz
Connectivity	CANbus, I²C, LINbus, SCI, SPI
Peripherals	DMA, LCD, POR, PWM, WDT
Number of I/O	73
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 4x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	100-LQFP
Supplier Device Package	100-LQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/s912zvhy32f1vll

5.1.3.3 Low-Power Modes

5.1.3.3.1 Stop Mode

The execution of the CPU STOP instruction leads to stop mode only when all bus masters (CPU, or others, depending on the device) have finished processing. The operation during stop mode depends on the ENBDC and BDCCIS bit settings as summarized in Table 5-3

Table 5-3. BDC STOP Operation Dependencies

ENBDC	BDCCIS	Description Of Operation
0	0	BDC has no effect on STOP mode.
0	1	BDC has no effect on STOP mode.
1	0	Only BDCCLK clock continues
1	1	All clocks continue

A disabled BDC has no influence on stop mode operation. In this case the BDCSI clock is disabled in stop mode thus it is not possible to enable the BDC from within stop mode.

STOP Mode With BDC Enabled And BDCCIS Clear

If the BDC is enabled and BDCCIS is clear, then the BDC prevents the BDCCLK clock (Figure 5-5) from being disabled in stop mode. This allows BDC communication to continue throughout stop mode in order to access the BDCCSR register. All other device level clock signals are disabled on entering stop mode.

NOTE

This is intended for application debugging, not for fast flash programming. Thus the CLKSW bit must be clear to map the BDCSI to BDCCLK.

With the BDC enabled, an internal acknowledge delays stop mode entry and exit by 2 BDCSI clock + 2 bus clock cycles. If no other module delays stop mode entry and exit, then these additional clock cycles represent a difference between the debug and not debug cases. Furthermore if a BDC internal access is being executed when the device is entering stop mode, then the stop mode entry is delayed until the internal access is complete (typically for 1 bus clock cycle).

Accesses to the internal memory map are not possible when the internal device clocks are disabled. Thus attempted accesses to memory mapped resources are suppressed and the NORESP flag is set. Resources can be accessed again by the next command received following exit from Stop mode.

A BACKGROUND command issued whilst in stop mode remains pending internally until the device leaves stop mode. This means that subsequent active BDM commands, issued whilst BACKGROUND is pending, set the ILLCMD flag because the device is not yet in active BDM.

If ACK handshaking is enabled, then the first ACK, following a stop mode entry is long to indicate a stop exception. The BDC indicates a stop mode occurrence by setting the BDCCSR bit STOP. If the host attempts further communication before the ACK pulse generation then the OVRUN bit is set.

Chapter 6
S12Z Debug (S12ZDBGV2) Module

Table 6-1. Revision History Table

Revision Number	Revision Date	Sections Affected	Description Of Changes
2.04	19.APR.2012	Section 6.4.5.2.1	Documented DBGTB read dependency on PROFILE bit
2.05	23.MAY.2012	General	Formatting changes to support DBGV3 from single source
2.06	10.SEP.2012	Section 6.4.5.3	Added NOTE about PC trace buffer entries for Comp D timestamps
2.07	18.OCT.2012	General	Formatting corrections
2.08	16.NOV.2012	Section 6.5.1	Modified step over breakpoint information
2.09	19.DEC.2012	General	Formatting corrections
2.10	28.JUN.2013	General Section 6.3.2.21 Section 6.3.2.1 Section 6.3.2.5	Emphasized need to set TSOURCE for tracing or profiling Corrected DBGCDM write access dependency Corrrected ARM versus PTACT dependency Modified DBGTBH read access dependencies
2.11	15.JUL.2013	Section 6.3.2	Added explicit names to state control register bit fields

6.1 Introduction

The DBG module provides on-chip breakpoints and trace buffer with flexible triggering capability to allow non-intrusive debug of application software. The DBG module is optimized for the S12Z architecture and allows debugging of CPU module operations.

Typically the DBG module is used in conjunction with the BDC module, whereby the user configures the DBG module for a debugging session over the BDC interface. Once configured the DBG module is armed and the device leaves active BDM returning control to the user program, which is then monitored by the DBG module. Alternatively the DBG module can be configured over a serial interface using SWI routines.

6.1.1 Glossary

Table 6-2. Glossary Of Terms

Term	Definition
COF	Change Of Flow. Change in the program flow due to a conditional branch, indexed jump or interrupt
PC	Program Counter

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0102	DBGTCRH	R	reserved	TSOURCE	TRANGE		TRCMOD		TALIGN	
		W								
0x0103	DBGTCRL	R	0	0	0	0	DSTAMP	PDOE	PROFILE	STAMP
		W								
0x0104	DBGTB	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0105	DBGTB	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0106	DBGCNT	R	0	CNT						
		W								
0x0107	DBGSCR1	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0108	DBGSCR2	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x0109	DBGSCR3	R	C3SC1	C3SC0	C2SC1	C2SC0	C1SC1	C1SC0	C0SC1	C0SC0
		W								
0x010A	DBGEFR	R	PTBOVF	TRIGF	0	EEVF	ME3	ME2	ME1	ME0
		W								
0x010B	DBGSR	R	TBF	0	0	PTACT	0	SSF2	SSF1	SSF0
		W								
0x010C-0x010F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0110	DBGACTL	R	0	NDB	INST	0	RW	RWE	reserved	COMPE
		W								
0x0111-0x0114	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0115	DBGAAH	R	DBGAA[23:16]							
		W								
0x0116	DBGAAM	R	DBGAA[15:8]							
		W								
0x0117	DBGAAL	R	DBGAA[7:0]							
		W								
0x0118	DBGAD0	R	Bit 31	30	29	28	27	26	25	Bit 24
		W								
0x0119	DBGAD1	R	Bit 23	22	21	20	19	18	17	Bit 16
		W								
0x011A	DBGAD2	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								

Figure 6-2. Quick Reference to DBG Registers

Table 6-41. Comparator Address Bus Matches

Access	Address	ADDR[n]	ADDR[n+1]	ADDR[n+2]	ADDR[n+3]
8-bit	ADDR[n]	Match	No Match	No Match	No Match

If the comparator INST bit is set, the comparator address register contents are compared with the PC, the data register contents and access type bits are ignored. The comparator address register must be loaded with the address of the first opcode byte.

6.4.2.2 Address and Data Comparator Match

Comparators A and C feature data comparators, for data access comparisons. The comparators do not evaluate if accessed data is valid. Accesses across aligned 32-bit boundaries are split internally into consecutive accesses. The data comparator mapping to accessed addresses for the CPU is shown in Table 6-42, whereby the Address column refers to the lowest 2 bits of the lowest accessed address. This corresponds to the most significant data byte.

Table 6-42. Comparator Data Byte Alignment

Address[1:0]	Data Comparator
00	DBGxD0
01	DBGxD1
10	DBGxD2
11	DBGxD3

The fixed mapping of data comparator bytes to addresses within a 32-bit data field ensures data matches independent of access size. To compare a single data byte within the 32-bit field, the other bytes within that field must be masked using the corresponding data mask registers. This ensures that any access of that byte (32-bit,16-bit or 8-bit) with matching data causes a match. If no bytes are masked then the data comparator always compares all 32-bits and can only generate a match on a 32-bit access with correct 32-bit data value. In this case, 8-bit or 16-bit accesses within the 32-bit field cannot generate a match even if the contents of the addressed bytes match because all 32-bits must match. In Table 6-43 the Access Address column refers to the address bits[1:0] of the lowest accessed address (most significant data byte).

Table 6-43. Data Register Use Dependency On CPU Access Type

			Memory Address[2:0]						
Case	Access Address	Access Size	000	001	010	011	100	101	110
1	00	32-bit	DBGxD0	DBGxD1	DBGxD2	DBGxD3			
2	01	32-bit		DBGxD1	DBGxD2	DBGxD3	DBGxD0		
3	10	32-bit			DBGxD2	DBGxD3	DBGxD0	DBGxD1	
4	11	32-bit				DBGxD3	DBGxD0	DBGxD1	DBGxD2
5	00	16-bit	DBGxD0	DBGxD1					
6	01	16-bit		DBGxD1	DBGxD2				
7	10	16-bit			DBGxD2	DBGxD3			
8	11	16-bit				DBGxD3	DBGxD0		

Address Offset	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000F	CPMU ARMCOP	R	0	0	0	0	0	0	0	0
		W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0010	CPMU HTCTL	R	0	0	VSEL	0	HTE	HTDS	HTIE	HTIF
		W								
0x0011	CPMU LVCTL	R	0	0	0	0	0	LVDS	LVIE	LVIF
		W								
0x0012	CPMU APICTL	R	APICLK	0	0	APIES	APIEA	APIFE	APIE	APIF
		W								
0x0013	CPMUACLKTR	R	ACLKTR5	ACLKTR4	ACLKTR3	ACLKTR2	ACLKTR1	ACLKTR0	0	0
		W								
0x0014	CPMUAPIRH	R	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
		W								
0x0015	CPMUAPIRL	R	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
		W								
0x0016	RESERVED CPMUTEST3	R	0	0	0	0	0	0	0	0
		W								
0x0017	CPMUHTTR	R	HTOE	0	0	0	HTTR3	HTTR2	HTTR1	HTTR0
		W								
0x0018	CPMU IRCTRIMH	R	TCTRIM[4:0]					0	IRCTRIM[9:8]	
		W								
0x0019	CPMU IRCTRIML	R	IRCTRIM[7:0]							
		W								
0x001A	CPMUOSC	R	OSCE	0	Reserved	0	0	0	0	0
		W								
0x001B	CPMUPROT	R	0	0	0	0	0	0	0	PROT
		W								
0x001C	RESERVED CPMUTEST2	R	0	0	0	0	0	0	0	0
		W								
0x001D	CPMU VREGCTL	R	0	0	0	0	0	0	EXTXON	INTXON
		W								
0x001E	CPMUOSC2	R	0	0	0	0	0	0	OMRE	OSCMOD
		W								
0x001F	CPMU RESERVED1F	R	0	0	0	0	0	0	0	0
		W								

= Unimplemented or Reserved

Figure 7-3. CPMU Register Summary

Table 7-29. CPMUOSC Field Descriptions

Field	Description
7 OSCE	**Oscillator Enable Bit** — This bit enables the external oscillator (XOSCLCP). The UPOSC status bit in the CPMIUFLG register indicates when the oscillation is stable and when OSCCLK can be selected as source of the Bus Clock or source of the COP or RTI.**If the oscillator clock monitor reset is enabled (OMRE = 1 in CPMUOSC2 register), then a loss of oscillation will lead to an oscillator clock monitor reset.** 0 External oscillator is disabled. REFCLK for PLL is IRCCLK. 1 External oscillator is enabled. Oscillator clock monitor is enabled. External oscillator is qualified by PLLCLK. REFCLK for PLL is the external oscillator clock divided by REFDIV. **If OSCE bit has been set (write "1") the EXTAL and XTAL pins are exclusively reserved for the oscillator and they can not be used anymore as general purpose I/O until the next system reset.** **Note:** When starting up the external oscillator (either by programming OSCE bit to 1 or on exit from Full Stop Mode with OSCE bit already 1) the software must wait for a minimum time equivalent to the startup-time of the external oscillator t_{UPOSC} before entering Pseudo Stop Mode.
5 Reserved	Do not alter this bit from its reset value. It is for Manufacturer use only and can change the Oscillator behavior.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0021 PAFLG	R	0	0	0	0	0	0	PAOVF	PAIF
	W								
0x0022 PACNTH	R	PACNT15	PACNT14	PACNT13	PACNT12	PACNT11	PACNT10	PACNT9	PACNT8
	W								
0x0023 PACNTL	R	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
	W								
0x0024–0x002B Reserved	R								
	W								
0x002C OCPD	R	OCPD7	OCPD6	OCPD5	OCPD4	OCPD3	OCPD2	OCPD1	OCPD0
	W								
0x002D Reserved	R								
0x002E PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
	W								
0x002F Reserved	R								
	W								

Figure 8-5. TIM16B8CV3 Register Summary (Sheet 2 of 2)

1. The register is available only if corresponding channel exists.

8.3.2.1 Timer Input Capture/Output Compare Select (TIOS)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
Reset	0	0	0	0	0	0	0	0

Figure 8-6. Timer Input Capture/Output Compare Select (TIOS)

Read: Anytime

Write: Anytime

Table 8-2. TIOS Field Descriptions

Note: Writing to unavailable bits has no effect. Reading from unavailable bits return a zero.

Field	Description
7:0 IOS[7:0]	**Input Capture or Output Compare Channel Configuration** 0 The corresponding implemented channel acts as an input capture. 1 The corresponding implemented channel acts as an output compare.

Chapter 9
Pulse-Width Modulator (S12PWM8B8CV2)

Table 9-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
v02.00	Feb. 20, 2009	All	Initial revision of scalable PWM. Started from pwm_8b8c (v01.08).

9.1 Introduction

The Version 2 of S12 PWM module is a channel scalable and optimized implementation of S12 PWM8B8C Version 1. The channel is scalable in pairs from PWM0 to PWM7 and the available channel number is 2, 4, 6 and 8. The shutdown feature has been removed and the flexibility to select one of four clock sources per channel has improved. If the corresponding channels exist and shutdown feature is not used, the Version 2 is fully software compatible to Version 1.

9.1.1 Features

The scalable PWM block includes these distinctive features:

- Up to eight independent PWM channels, scalable in pairs (PWM0 to PWM7)
- Available channel number could be 2, 4, 6, 8 (refer to device specification for exact number)
- Programmable period and duty cycle for each channel
- Dedicated counter for each PWM channel
- Programmable PWM enable/disable for each channel
- Software selection of PWM duty pulse polarity for each channel
- Period and duty cycle are double buffered. Change takes effect when the end of the effective period is reached (PWM counter reaches zero) or when the channel is disabled.
- Programmable center or left aligned outputs on individual channels
- Up to eight 8-bit channel or four 16-bit channel PWM resolution
- Four clock sources (A, B, SA, and SB) provide for a wide range of frequencies
- Programmable clock select logic

9.1.2 Modes of Operation

There is a software programmable option for low power consumption in wait mode that disables the input clock to the prescaler.

10.4.2.6 ADC Conversion Flow Control Register (ADCFLWCTL)

Bit set and bit clear instructions should not be used to access this register.

When the ADC is enabled the bits of ADCFLWCTL register can be modified after a latency time of three Bus Clock cycles.
All bits are cleared if bit ADC_EN is clear or via ADC soft-reset.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	SEQA	TRIG	RSTA	LDOK	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-9. ADC Conversion Flow Control Register (ADCFLWCTL)

Read: Anytime

Write:

- Bits SEQA, TRIG, RSTA, LDOK can only be set if bit ADC_EN is set.
- Writing 1’b0 to any of these bits does not have an effect

Timing considerations (Trigger Event - channel sample start) depending on ADC mode configuration:

- Restart Mode
  When the Restart Event has been processed (initial command of current CSL is loaded) it takes two Bus Clock cycles plus two ADC conversion clock cycles (pump phase) from the Trigger Event (bit TRIG set) until the select channel starts to sample.
  During a conversion sequence (back to back conversions) it takes five Bus Clock cycles plus two ADC conversion clock cycles (pump phase) from current conversion period end until the newly selected channel is sampled in the following conversion period.
- Trigger Mode
  When a Restart Event occurs a Trigger Event is issued simultaneously. The time required to process the Restart Event is mainly defined by the internal read data bus availability and therefore can vary. In this mode the Trigger Event is processed immediately after the Restart Event is finished and both conversion flow control bits are cleared simultaneously. From de-assert of bit TRIG until sampling begins five Bus Clock cycles are required. Hence from occurrence of a Restart Event until channel sampling it takes five Bus Clock cycles plus an uncertainty of a few Bus Clock cycles.

For more details regarding the sample phase please refer to Section 10.5.2.2, “Sample and Hold Machine with Sample Buffer Amplifier.

12.4.6 Receiver

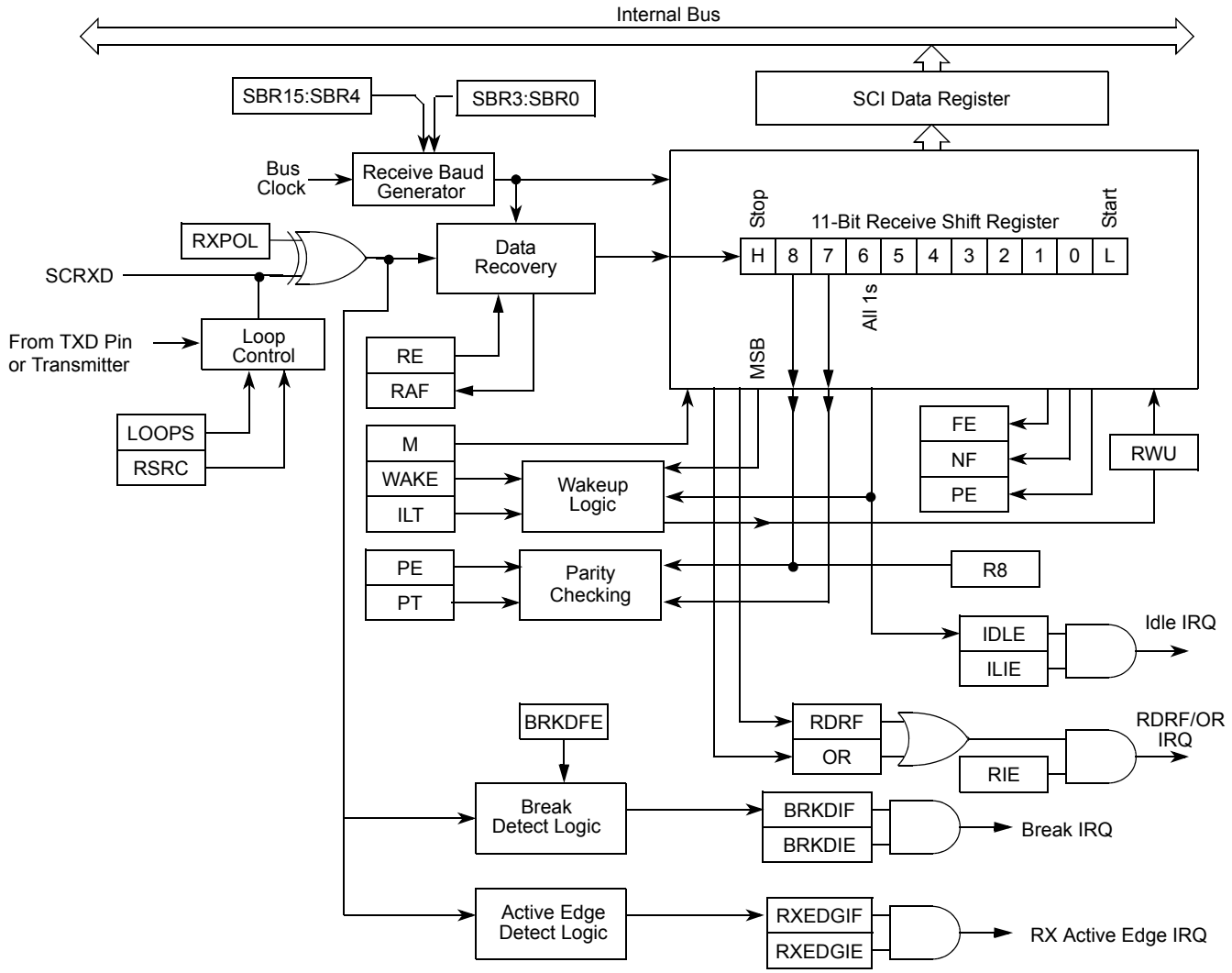


Figure 12-20. SCI Receiver Block Diagram

12.4.6.1 Receiver Character Length

The SCI receiver can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When receiving 9-bit data, bit R8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

12.4.6.2 Character Reception

During an SCI reception, the receive shift register shifts a frame in from the RXD pin. The SCI data register is the read-only buffer between the internal data bus and the receive shift register.

After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the SCI data register. The receive data register full flag, RDRF, in SCI status register 1 (SCISR1) becomes set,

Table 13-9. SPIF Interrupt Flag Clearing Sequence

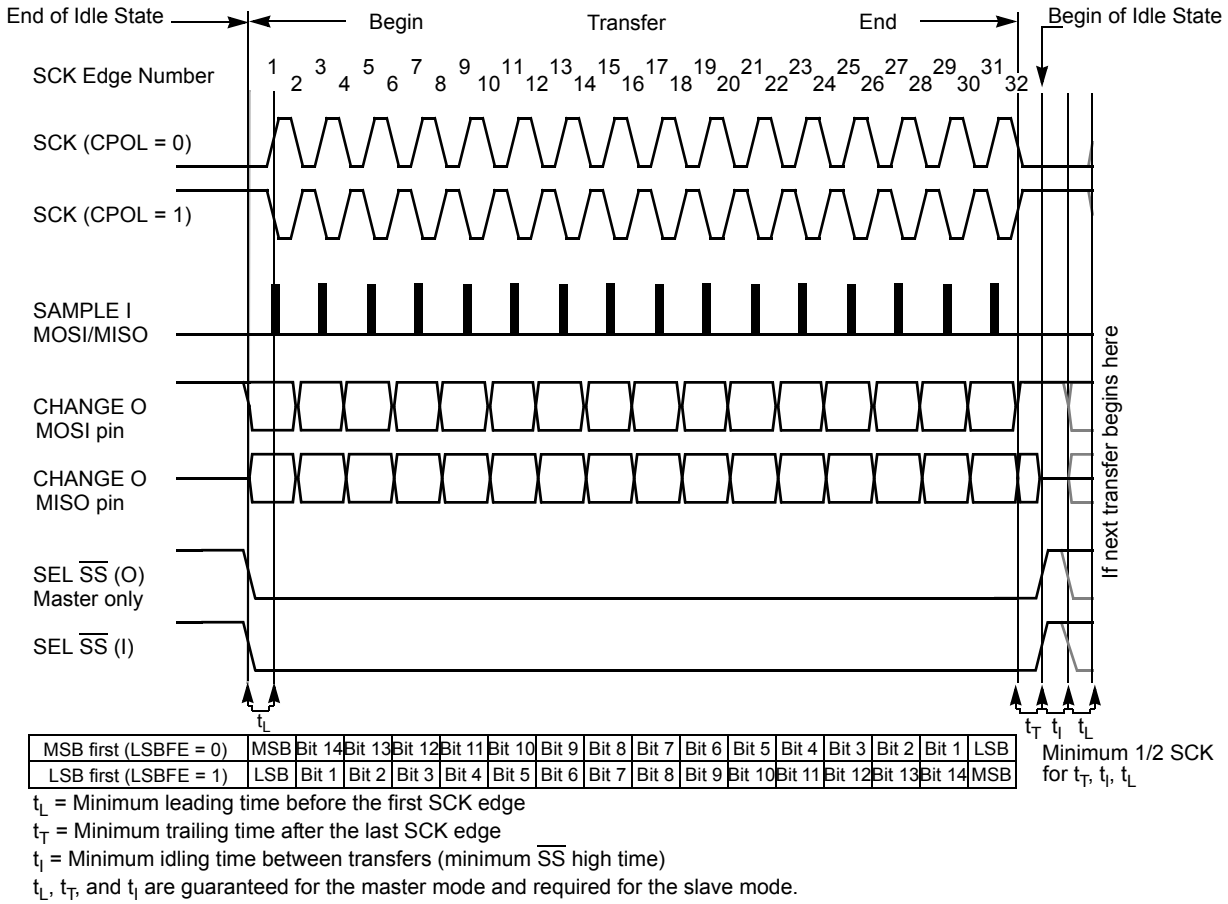
XFRW Bit	SPIF Interrupt Flag Clearing Sequence		
0	Read SPISR with SPIF == 1	**then**	Read SPIDRL
1	Read SPISR with SPIF == 1	**then**	Byte Read SPIDRL (1)
			or
			Byte Read SPIDRH (2) then Byte Read SPIDRL
			or
			Word Read (SPIDRH:SPIDRL)

1. Data in SPIDRH is lost in this case.
2. SPIDRH can be read repeatedly without any effect on SPIF. SPIF Flag is cleared only by the read of SPIDRL after reading SPISR with SPIF == 1.

Table 13-10. SPTEF Interrupt Flag Clearing Sequence

XFRW Bit	SPTEF Interrupt Flag Clearing Sequence		
0	Read SPISR with SPTEF == 1	**then**	Write to SPIDRL (1)
1	Read SPISR with SPTEF == 1	**then**	Byte Write to SPIDRL 1(2)
			or
			Byte Write to SPIDRH 1(3) then Byte Write to SPIDRL 1
			or
			Word Write to (SPIDRH:SPIDRL) 1

1. Any write to SPIDRH or SPIDRL with SPTEF == 0 is effectively ignored.
2. Data in SPIDRH is undefined in this case.
3. SPIDRH can be written repeatedly without any effect on SPTEF. SPTEF Flag is cleared only by writing to SPIDRL after reading SPISR with SPTEF == 1.



t_L = Minimum leading time before the first SCK edge
t_T = Minimum trailing time after the last SCK edge
t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time)
t_L, t_T, and t_I are guaranteed for the master mode and required for the slave mode.

Figure 13-13. SPI Clock Format 0 (CPHA = 0), with 16-Bit Transfer Width selected (XFRW = 1)

In slave mode, if the $\overline{SS}$ line is not deasserted between the successive transmissions then the content of the SPI data register is not transmitted; instead the last received data is transmitted. If the $\overline{SS}$ line is deasserted for at least minimum idle time (half SCK cycle) between successive transmissions, then the content of the SPI data register is transmitted.

In master mode, with slave select output enabled the $\overline{SS}$ line is always deasserted and reasserted between successive transfers for at least minimum idle time.

13.4.3.3 CPHA = 1 Transfer Format

Some peripherals require the first SCK edge before the first data bit becomes available at the data out pin, the second edge clocks data into the system. In this format, the first SCK edge is issued by setting the CPHA bit at the beginning of the n[1]-cycle transfer operation.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its first data bit to the serial data input pin of the master.

A half SCK cycle later, the second edge appears on the SCK pin. This is the latching edge for both the master and slave.

1. n depends on the selected transfer width, please refer to Section 13.3.2.2, “SPI Control Register 2 (SPICR2)

Table 15-8. LCD Clock and Frame Frequency

Source clock Frequency in Hz	LCD Clock Prescaler		Divider	LCD Clock Frequency [Hz]	Frame Frequency [Hz]			
	LCLK1	LCLK0			1/1 Duty	1/2 Duty	1/3 Duty	1/4 Duty
RTCCLK = 64000	0 0 1 1	0 1 0 1	64 128 256 512	1000 500 250 125	1000 500250 125	500 250 125 63	333 167 83 42	250 125 63 31

For other combinations of RTCCLK and divider not shown in Table 15-8, the following formula may be used to calculate the LCD frame frequency for each multiplex mode:

$$\text{LCD Frame Frequency (Hz)} = \left[\frac{(\text{RTCCLK (Hz)})}{\text{Divider}}\right] \cdot \text{Duty}$$

The possible divider values are shown in Table 15-8.

15.4.1.3 LCD RAM

For a segment on the LCD to be displayed, data must be written to the LCD RAM which is shown in Section 15.3, “Memory Map and Register Definition”. The 160 bits in the LCD RAM correspond to the 160 segments that are driven by the frontplane and backplane drivers. Writing a 1 to a given location will result in the corresponding display segment being driven with a differential RMS voltage necessary to turn the segment ON when the LCDEN bit is set and the corresponding FP[39:0]EN bit is set. Writing a 0 to a given location will result in the corresponding display segment being driven with a differential RMS voltage necessary to turn the segment OFF. The LCD RAM is a dual port RAM that interfaces with the internal address and data buses of the MCU. It is possible to read from LCD RAM locations for scrolling purposes. When LCDEN = 0, the LCD RAM can be used as on-chip RAM. Writing or reading of the LCDEN bit does not change the contents of the LCD RAM. After a reset, the LCD RAM contents will be indeterminate.

15.4.1.4 LCD Driver System Enable and Frontplane Enable Sequencing

If LCDEN = 0 (LCD40F4BV3 driver system disabled) and the frontplane enable bit, FP[39:0]EN, is set, the frontplane driver waveform will not appear on the output until LCDEN is set. If LCDEN = 1 (LCD40F4BV3 driver system enabled), the frontplane driver waveform will appear on the output as soon as the corresponding frontplane enable bit, FP[39:0]EN, in the registers LCDFPENR0–LCDFPENR4 is set.

15.4.1.5 LCD Bias and Modes of Operation

The LCD40F4BV3 driver has five modes of operation:

- 1/1 duty (1 backplane), 1/1 bias (2 voltage levels)
- 1/2 duty (2 backplanes), 1/2 bias (3 voltage levels)
- 1/2 duty (2 backplanes), 1/3 bias (4 voltage levels)

- 16-bit modulus down counter with interrupt

17.1.3 Block Diagram

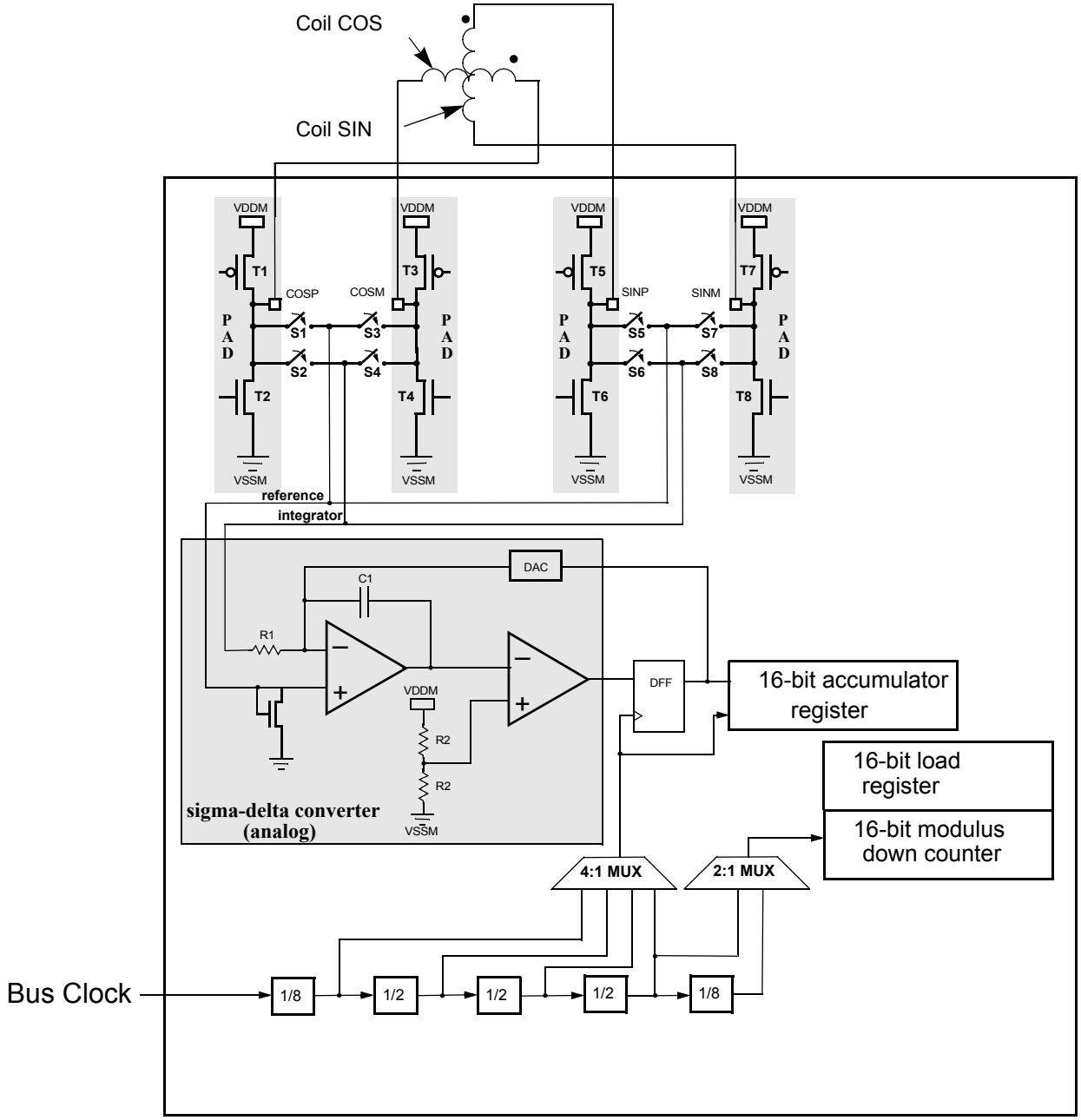


Figure 17-1. SSD Block Diagram

- In linear attack operation SSGAMPB (SSGAMP's buffer) will be increased by SSGAA every SSGSSGDUR + 1 tone cycle. In linear decay operation, SSGAMPB will be decreased by SSGAA every SSGDUR + 1 tone cycle. See below linear attack/decay formula.
- In gong attack operation SSGAMPB will be increased by SSGAMPB/32 every SSGDUR + 1 tone cycle. In gong decay operation SSGAMPB will be decreased by SSGAMPB/32 every SSGDUR + 1 tone cycle. See below gong attack/decay formula.
- In exponential attack operation SSGAMPB will multiply with 2 then add 1 every SSGDUR + 1 tone cycle. In exponential decay operation, SSGAMPB will be divided by 2 every SSGDUR + 1 tone cycle. See below exponential attack/decay formula.

Linear attack operation:

```
SSGAMPB = SSGAMP;
do{
    SSGAMPB = SSGAMPB + SSGAA_buf;
} While (SSGAMPB < AT_buf)
```

Where : AT_buf is the internal buffer of amplitude threshold register SSGAT.
SSGAA_buf is the internal buffer of SSGAA.

Linear decay operation:

```
SSGAMPB = SSGAMP;
do{
    SSGAMPB = SSGAMPB - SSGAA_buf;
} While (SSGAMPB > AT_buf)
```

Where : AT_buf is the internal buffer of amplitude threshold register SSGAT.
SSGAA_buf is the internal buffer of SSGAA.

Gong attack operation:

```
SSGAMPB = SSGAMP;
AMP_int = {SSGAMPB, 5'b00000};

do{
    AMP_int = AMP_int + (AMP_int >> 5);
    SSGAMPB = AMP_int >> 5;
  } While (SSGAMPB < AT_buf)
```

where : AMP_int is a 16 bit internal register.

AT_buf is the internal buffer of amplitude threshold register SSGAT.

Gong decay operation:

```
SSGAMPB = SSGAMP;
AMP_int = {SSGAMPB, 5'b11111};

do{
    AMP_int = AMP_int - (AMP_int >> 5);
    SSGAMPB = AMP_int >> 5;
  } While (SSGAMPB > AT_buf)
```

where : AMP_int is a 16 bit internal register.

AT_buf is the internal buffer of amplitude threshold register SSGAT.

Upon clearing CCIF to launch the Erase EEPROM Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase EEPROM Sector operation has completed.

Table 21-67. Erase EEPROM Sector 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 21-29)
		Set if an invalid global address [23:0] is suppliedsee Table 21-3
		Set if a misaligned word address is supplied (global address [0] != 0)
	FPVIOL	Set if the selected area of the EEPROM memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation

21.4.7.17 Protection Override Command

The Protection Override command allows the user to temporarily override the protection limits, either decreasing, increasing or disabling protection limits, on P-Flash and/or EEPROM, if the comparison key provided as a parameter loaded on FCCOB matches the value of the key previously programmed on the Flash Configuration Field (see Table 21-4). The value of the Protection Override Comparison Key must not be 16'hFFFF, that is considered invalid and if used as argument will cause the Protection Override feature to be disabled. Any valid key value that does not match the value programmed in the Flash Configuration Field will cause the Protection Override feature to be disabled. Current status of the Protection Override feature can be observed on FPSTAT FPOVRD bit (see Section 21.3.2.4, "Flash Protection Status Register (FPSTAT)).

Table 21-68. Protection Override Command FCCOB Requirements

Register	FCCOB Parameters	
FCCOB0	0x13	Protection Update Selection [1:0] See Table 21-69
FCCOB1	Comparison Key	
FCCOB2	reserved	New FPROT value
FCCOB3	reserved	New DFPROT value

Table 21-69. Protection Override selection description

Protection Update Selection code [1:0]	Protection register selection
bit 0	Update P-Flash protection 0 - keep unchanged (do not update) 1 - update P-Flash protection with new FPROT value loaded on FCCOB

When opening the resistors path to ground by changing BSESE, BSEAE or BSUSE, BSUAE then for a time T_{EN_UNC} + two bus cycles the measured value is invalid. This is to let internal nodes be charged to correct value. BVHIE, BVLIE might be cleared for this time period to avoid false interrupts.

23.5 Application Information

23.5.1 Module Initialization

The following steps should be used to configure the module before starting the transmission:

1. Set the slew rate in the LPSLRM register to the desired transmission baud rate.
2. When using the LIN Physical Layer for other purposes than LIN transmission, de-activate the dominant timeout feature in the LPSLRM register if needed.
3. In most cases, the internal pullup should be enabled in the LPCR register.
4. Route the desired source in the PIM module to the LIN Physical Layer.
5. Select the transmit mode (Receive only mode or Normal mode) in the LPCR register.
6. If the SCI is selected as source, activate the wake-up feature in the LPCR register if needed for the application (SCI active edge interrupt must also be enabled).
7. Enable the LIN Physical Layer in the LPCR register.
8. Wait for a minimum of a transmit bit.
9. Begin transmission if needed.

NOTE

It is not allowed to try to clear LPOCIF or LPDTIF if they are already cleared. Before trying to clear an error flag, always make sure that it is already set.

23.5.2 Interrupt handling in Interrupt Service Routine (ISR)

Both interrupts (TxD-dominant timeout and overcurrent) represent a failure in transmission. To avoid more disturbances on the transmission line, the transmitter is de-activated in both cases. The interrupt subroutine must take care of clearing the error condition and starting the routine that re-enables the transmission. For that purpose, the following steps are recommended:

1. First, the cause of the interrupt must be cleared:
   - The overcurrent will be gone after the transmitter has been disabled.
   - The TxD-dominant timeout condition will be gone once the selected source for LPTxD has turned recessive.
2. Clear the corresponding enable bit (LPDTIE or LPOCIE) to avoid entering the ISR again until the flags are cleared.
3. Notify the application of the error condition (LIN Error handler) and leave the ISR.

In the LIN Error handler, the following sequence is recommended:

1. Disable the LIN Physical Layer (LPCR) while re-configuring the transmission.
   - If the receiver must remain enabled, set the LIN Physical Layer into receive only mode instead.
2. Do all required configurations (SCI, etc.) to re-enable the transmission.
3. Wait for a transmit bit (this is needed to successfully re-enable the transmitter).

0x0A40–0x0A7F Motor Control (MC)

0x0A67	MCDC3L	R	D7	D6	D5	D4	D3	D2	D1	D0
		W								
0x0A68–0x0A7F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0A80–0x0A87 Stepper Stall Detector (SSD0)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0A80	RTZ0CTL	R	ITG	DCOIL	RCIR	POL	0	0	STEP	
		W								
0x0A81	MDC0CTL	R	MCZIE	MODMC	RDMCL	PRE	0	MCEN	0	AOVIE
		W					FLMC			
0x0A82	SSD0CTL	R	RTZE	SDCPU	SSDWAI	FTST	0	0	ACLKS	
		W								
0x0A83	SSD0FLG	R	MCZIF	0	0	0	0	0	0	AOVIF
		W								
0x0A84	MDC0CNTH	R	MDCCNT[15:8]							
		W								
0x0A85	MDC0CNTL	R	MDCCNT[7:0]							
		W								
0x0A86	ITG0ACCH	R	ITGACC[15:8]							
		W								
0x0A87	ITG0ACCL	R	ITGACC[7:0]							
		W								

0x0A88–0x0A8F Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0A88-0x0A8F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0A90–0x0A97 Stepper Stall Detector (SSD1)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0A90	RTZ1CTL	R	ITG	DCOIL	RCIR	POL	0	0	STEP	
		W								
0x0A91	MDC1CTL	R	MCZIE	MODMC	RDMCL	PRE	0	MCEN	0	AOVIE
		W					FLMC			