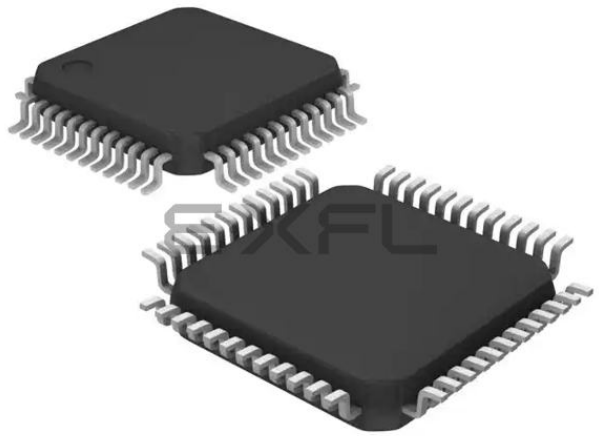

Welcome to E-XFL.COM

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, IrDA, LINbus, SCI, SPI, UART/USART
Peripherals	LVD, POR, PWM, WDT
Number of I/O	34
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	5.5V ~ 18V
Data Converters	A/D 10x10b; D/A 1x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 150°C (TA)
Mounting Type	Surface Mount
Package / Case	48-LQFP
Supplier Device Package	48-LQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/s912zvla12f0wlf

- Amplifier output connected to ADC
- Amplifier signal reference voltage selectable from DAC, VDDA/2 or input pin

1.4.16 Supply Voltage Sensor (BATS)

- Monitoring of supply (VSUP) voltage
- Internal ADC interface from an internal resistive divider
- Generation of low or high voltage interrupts

1.4.17 On-Chip Voltage Regulator system (VREG)

- Voltage regulator
  - Linear voltage regulator directly supplied by VSUP
  - Supports 3.3V or 5V VDDX
  - Low-voltage detect on VSUP
  - Power-on reset (POR)
  - Low-voltage reset (LVR) for VDDX domain
  - External ballast device support to extend current capability and reduce internal power dissipation
  - Capable of supplying both the MCU internally plus external components
  - Over-temperature interrupt
- Internal voltage regulator
  - Linear voltage regulator with bandgap reference
  - Low-voltage detect on VDDA
  - Power-on reset (POR) circuit
  - Low-voltage reset for VDD domain

Similarly the STEP1 command issued from a WAI instruction cannot be completed by the CPU until the CPU leaves wait mode due to an interrupt. The first STEP1 into wait mode sets the BDCCSR WAIT bit.

If the part is still in Wait mode and a further STEP1 is carried out then the NORESP and ILLCMD bits are set because the device is no longer in active BDM for the duration of WAI execution.

5.1.4 Block Diagram

A block diagram of the BDC is shown in Figure 5-1.

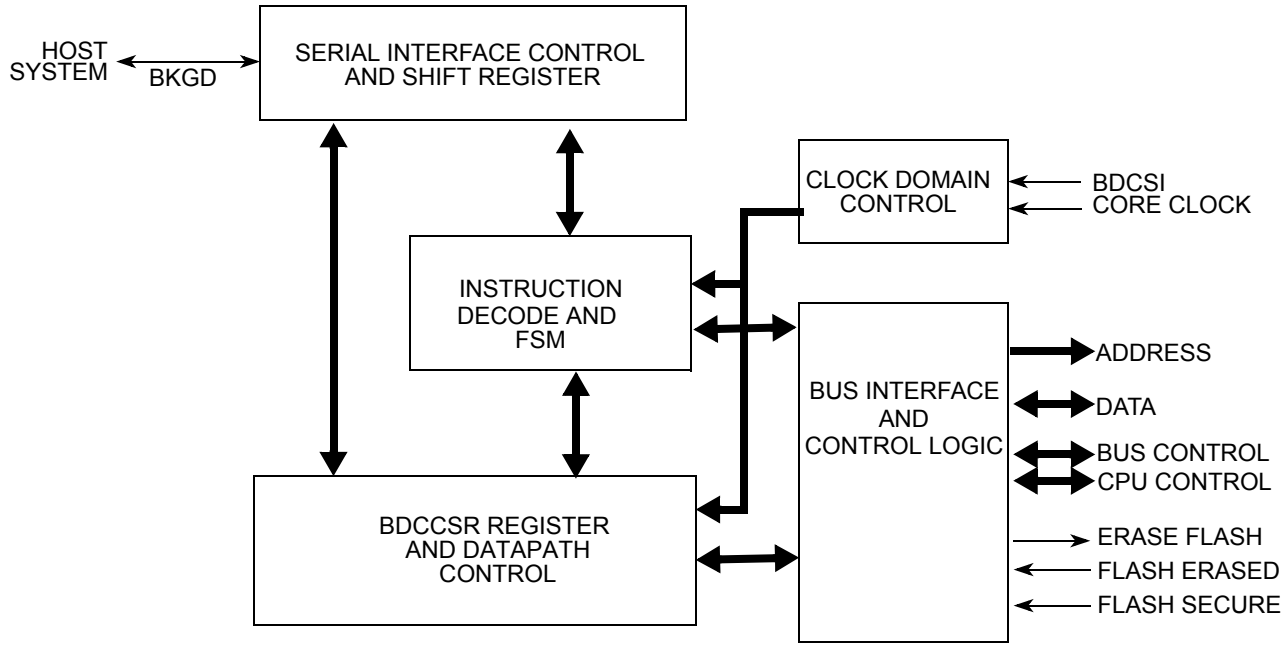

Figure 5-1. BDC Block Diagram

5.2 External Signal Description

A single-wire interface pin (BKGD) is used to communicate with the BDC system. During reset, this pin is a device mode select input. After reset, this pin becomes the dedicated serial interface pin for the BDC.

BKGD is a pseudo-open-drain pin with an on-chip pull-up. Unlike typical open-drain pins, the external RC time constant on this pin due to external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speed-up pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 5.4.6, “BDC Serial Interface” for more details.

5.3 Memory Map and Register Definition

5.3.1 Module Memory Map

Table 5-4 shows the BDC memory map.

Table 5-4. BDC Memory Map

Global Address	Module	Size (Bytes)
Not Applicable	BDC registers	2

5.3.2 Register Descriptions

The BDC registers are shown in Figure 5-2. Registers are accessed only by host-driven communications to the BDC hardware using READ_BDCCSR and WRITE_BDCCSR commands. They are not accessible in the device memory map.

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
Not Applicable	BDCCSRH	R	ENBDC	BDMACT	BDCCIS	0	STEAL	CLKSW	UNSEC	ERASE
		W								
Not Applicable	BDCCSRL	R	WAIT	STOP	RAMWF	OVRUN	NORESP	RDINV	ILLACC	ILLCMD
		W								

= Unimplemented, Reserved 0 = Always read zero

Figure 5-2. BDC Register Summary

5.3.2.1 BDC Control Status Register High (BDCCSRH)

Register Address: This register is not in the device memory map. It is accessible using BDC inherent addressing commands

	7	6	5	4	3	2	1	0
R	ENBDC	BDMACT	BDCCIS	0	STEAL	CLKSW	UNSEC	ERASE
W								
Reset								
Secure AND SSC-Mode	1	1	0	0	0	0	0	0
Unsecure AND SSC-Mode	1	1	0	0	0	0	1	0
Secure AND NSC-Mode	0	0	0	0	0	0	0	0
Unsecure AND NSC-Mode	0	0	0	0	0	0	1	0

= Unimplemented, Reserved

0 = Always read zero

Figure 5-3. BDC Control Status Register High (BDCCSRH)

Table 5-6. BDCCSRL Field Descriptions (continued)

Field	Description
0 ILLCMD	**Illegal Command Flag** — Indicates an illegal BDC command. This bit is set in the following cases: When an unimplemented BDC command opcode is received. When a DUMP_MEM{_WS}, FILL_MEM{_WS} or READ_SAME{_WS} is attempted in an illegal sequence. When an active BDM command is received whilst BDM is not active When a non Always-available command is received whilst the BDC is disabled or a flash mass erase is ongoing. When a non Always-available command is received whilst the device is secure Read commands return a value of 0xEE for each data byte Writing a “1” to this bit, clears the bit. 0 No illegal command detected. 1 Illegal BDC command detected.

5.4 Functional Description

5.4.1 Security

If the device resets with the system secured, the device clears the BDCCSR UNSEC bit. In the secure state BDC access is restricted to the BDCCSR register. A mass erase can be requested using the ERASE_FLASH command. If the mass erase is completed successfully, the device programs the security bits to the unsecure state and sets the BDC UNSEC bit. If the mass erase is unsuccessful, the device remains secure and the UNSEC bit is not set.

For more information regarding security, please refer to device specific security information.

5.4.2 Enabling BDC And Entering Active BDM

BDM can be activated only after being enabled. BDC is enabled by setting the ENBDC bit in the BDCCSR register, via the single-wire interface, using the command WRITE_BDCCSR.

After being enabled, BDM is activated by one of the following[1]:

- The BDC BACKGROUND command
- A CPU BGND instruction
- The DBG Breakpoint mechanism

Alternatively BDM can be activated directly from reset when resetting into Special Single Chip Mode.

The BDC is ready for receiving the first command 10 core clock cycles after the deassertion of the internal reset signal. This is delayed relative to the external pin reset as specified in the device reset documentation. On S12Z devices an NVM initialization phase follows reset. During this phase the BDC commands classified as always available are carried out immediately, whereas other BDC commands are subject to delayed response due to the NVM initialization phase.

NOTE

After resetting into SSC mode, the initial PC address must be supplied by the host using the WRITE_Rn command before issuing the GO command.

1. BDM active immediately out of special single-chip reset.

which cannot be disabled, are always handled by the CPU and have a fixed priority levels. A priority level of 0 effectively disables the associated I-bit maskable interrupt request.

If more than one interrupt request is configured to the same interrupt priority level the interrupt request with the higher vector address wins the prioritization.

The following conditions must be met for an I-bit maskable interrupt request to be processed.

1. The local interrupt enabled bit in the peripheral module must be set.
2. The setup in the configuration register associated with the interrupt request channel must meet the following conditions:
   a) The priority level must be set to non zero.
   b) The priority level must be greater than the current interrupt processing level in the condition code register (CCW) of the CPU (PRIOLVL[2:0] > IPL[2:0]).
3. The I-bit in the condition code register (CCW) of the CPU must be cleared.
4. There is no access violation interrupt request pending.
5. There is no SYS, SWI, SPARE, TRAP, Machine Exception or $\overline{\text{XIRQ}}$ request pending.

NOTE

All non I-bit maskable interrupt requests always have higher priority than I-bit maskable interrupt requests. If an I-bit maskable interrupt request is interrupted by a non I-bit maskable interrupt request, the currently active interrupt processing level (IPL) remains unaffected. It is possible to nest non I-bit maskable interrupt requests, e.g., by nesting SWI, SYS or TRAP calls.

6.4.2.1 Interrupt Priority Stack

The current interrupt processing level (IPL) is stored in the condition code register (CCW) of the CPU. This way the current IPL is automatically pushed to the stack by the standard interrupt stacking procedure. The new IPL is copied to the CCW from the priority level of the highest priority active interrupt request channel which is configured to be handled by the CPU. The copying takes place when the interrupt vector is fetched. The previous IPL is automatically restored from the stack by executing the RTI instruction.

6.4.3 Priority Decoder

The S12ZINTV0 module contains a priority decoder to determine the relative priority for all interrupt requests pending for the CPU.

A CPU interrupt vector is not supplied until the CPU requests it. Therefore, it is possible that a higher priority interrupt request could override the original exception which caused the CPU to request the vector. In this case, the CPU will receive the highest priority vector and the system will process this exception first instead of the original request.

If the interrupt source is unknown (for example, in the case where an interrupt request becomes inactive after the interrupt has been recognized, but prior to the vector request), the vector address supplied to the CPU defaults to that of the spurious interrupt vector.

Chapter 9
S12 Clock, Reset and Power Management Unit (S12CPMU_UHV)

Table 9-1. Revision History

Rev. No. (Item No)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V09.00	8 Sept.2014		• initial version for ZVL128, copied from ZVL32 • CPMUPROT register: added CPMUVREGCTL2 to list of protected registers • added CPMUVREGCTL2 register containing 5V/3V option Bit and respective trim values
V09.01	21 Oct. 2014		• Improved Figure: Start up of clock system after Reset • Improved Figure: Full stop mode using Oscillator • Improved Figure: Enabling the external oscillator • Improved Table: Trimming effect of ACLKTR • Improved Table: Trimming effect of HTTR • Register Description for CPMUHTCTL: Added note on how to compute V_{HT} • Functional Description PBE Mode: Added Note that the clock system might stall if osc monitor reset disabled (OMRE=0) • Signal Descriptions: changed recommended resistor for BCTL pin to 1KΩ
V09.02	11 Nov. 2014		• Moved VREG5VEN bit to CPMUVREGCTL register
V09.03	14 Nov. 2014		• Corrected typos

9.1 Introduction

This specification describes the function of the Clock, Reset and Power Management Unit (S12CPMU_UHV).

- The Pierce oscillator (XOSCLCP) provides a robust, low-noise and low-power external clock source. It is designed for optimal start-up margin with typical crystal oscillators.
- The Voltage regulator (VREGAUTO) operates from the range 6V to 18V. It provides all the required chip internal voltages and voltage monitors.
- The Phase Locked Loop (PLL) provides a highly accurate frequency multiplier with internal filter.
- The Internal Reference Clock (IRC1M) provides a 1MHz internal clock.

If PLL has locked (LOCK=1) $f_{VCO} = 2 \times f_{REF} \times (SYNDIV + 1)$

NOTE

f_{VCO} must be within the specified VCO frequency lock range. Bus frequency f_{bus} must not exceed the specified maximum.

The VCOFRQ[1:0] bits are used to configure the VCO gain for optimal stability and lock time. For correct PLL operation the VCOFRQ[1:0] bits have to be selected according to the actual target VCOCLK frequency as shown in Table 9-3. Setting the VCOFRQ[1:0] bits incorrectly can result in a non functional PLL (no locking and/or insufficient stability).

Table 9-3. VCO Clock Frequency Selection

VCOCLK Frequency Ranges	VCOFRQ[1:0]
32MHz <= f_{VCO} <= 48MHz	00
48MHz < f_{VCO} <= 64MHz	01
Reserved	10
Reserved	11

9.3.2.5 S12CPMU_UHV Reference Divider Register (CPMUREFDIV)

The CPMUREFDIV register provides a finer granularity for the PLL multiplier steps when using the external oscillator as reference.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	REFFRQ[1:0]		0	0	REFDIV[3:0]			
W								
Reset	0	0	0	0	1	1	1	1

Figure 9-8. S12CPMU_UHV Reference Divider Register (CPMUREFDIV)

Read: Anytime

Write: If PROT=0 (CPMUPROT register) and PLLSEL=1 (CPMUCLKS register), then write anytime. Else write has no effect.

NOTE

Write to this register clears the LOCK and UPOSC status bits.

9.3.2.11 S12CPMU_UHV RTI Control Register (CPMURTI)

This register selects the time-out period for the Real Time Interrupt.

The clock source for the RTI is either IRCCLK or OSCCLK depending on the setting of the RTIOSCSEL bit. In Stop Mode with PSTP=1 (Pseudo Stop Mode) and RTIOSCSEL=1 the RTI continues to run, else the RTI counter halts in Stop Mode.

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
Reset	0	0	0	0	0	0	0	0

Figure 9-14. S12CPMU_UHV RTI Control Register (CPMURTI)

Read: Anytime

Write: Anytime

NOTE

A write to this register starts the RTI time-out period. A change of the RTIOSCSEL bit (writing a different value or loosing UPOSC status) re-starts the RTI time-out period.

Table 9-11. CPMURTI Field Descriptions

Field	Description
7 RTDEC	**Decimal or Binary Divider Select Bit** — RTDEC selects decimal or binary based prescaler values. 0 Binary based divider value. See Table 9-12 1 Decimal based divider value. See Table 9-13
6–4 RTR[6:4]	**Real Time Interrupt Prescale Rate Select Bits** — These bits select the prescale rate for the RTI.See Table 9-12 and Table 9-13.
3–0 RTR[3:0]	**Real Time Interrupt Modulus Counter Select Bits** — These bits select the modulus counter target value to provide additional granularity.Table 9-12 and Table 9-13 show all possible divide values selectable by the CPMURTI register.

9.3.2.15 S12CPMU_UHV COP Timer Arm/Reset Register (CPMUARMCOP)

This register is used to restart the COP time-out period.

Module Base + 0x000F

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

Figure 9-18. S12CPMU_UHV CPMUARMCOP Register

Read: Always reads $00

Write: Anytime

When the COP is disabled (CR[2:0] = “000”) writing to this register has no effect.

When the COP is enabled by setting CR[2:0] nonzero, the following applies:

Writing any value other than $55 or $AA causes a COP reset. To restart the COP time-out period write $55 followed by a write of $AA. These writes do not need to occur back-to-back, but the sequence ($55, $AA) must be completed prior to COP end of time-out period to avoid a COP reset. Sequences of $55 writes are allowed. When the WCOP bit is set, $55 and $AA writes must be done in the last 25% of the selected time-out period; writing any value in the first 75% of the selected period will cause a COP reset.

9.3.2.16 High Temperature Control Register (CPMUHTCTL)

The CPMUHTCTL register configures the temperature sense features.

Module Base + 0x0010

	7	6	5	4	3	2	1	0
R	0	0	VSEL	0	HTE	HTDS	HTIE	HTIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-19. High Temperature Control Register (CPMUHTCTL)

Read: Anytime

Write: VSEL, HTE, HTIE and HTIF are write anytime, HTDS is read only

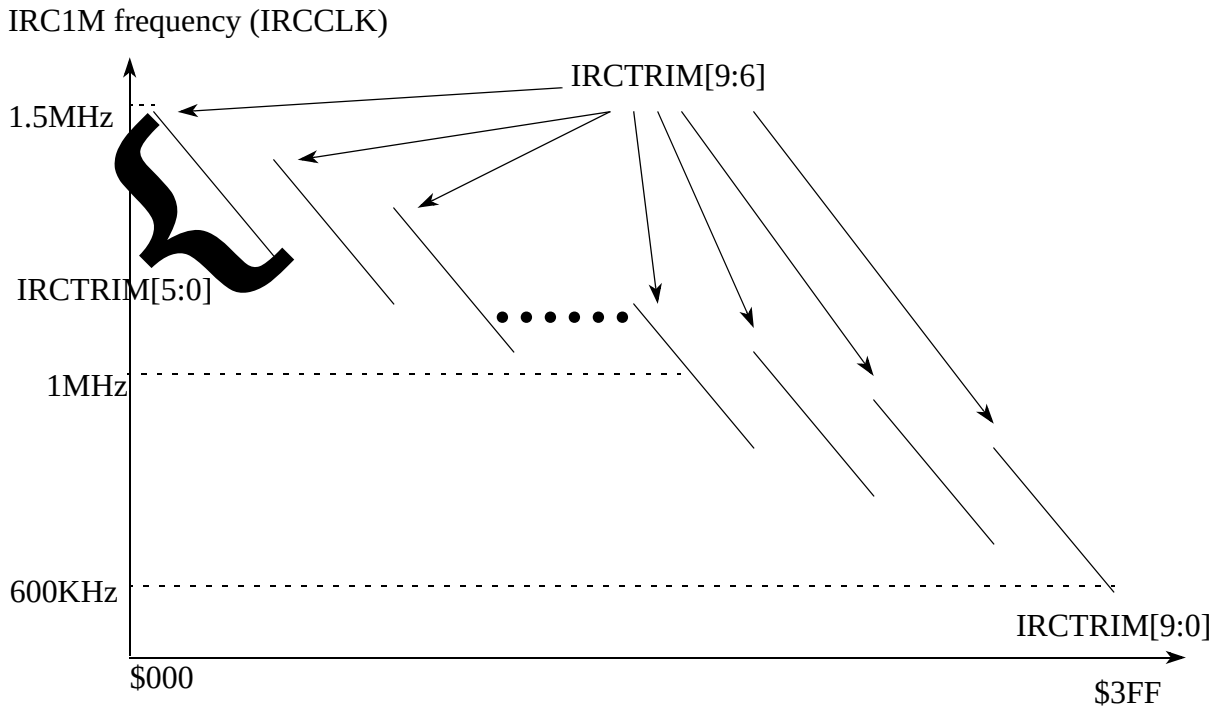

Figure 9-31. IRC1M Frequency Trimming Diagram

```
/* Procedure proposed by to setup PLL and Oscillator */
/* example for OSC = 4 MHz and Bus Clock = 25MHz, That is VCOCLK = 50MHz */

/* Initialize */
/* PLL Clock = 50 MHz, divide by one */
CPMUPOSTDIV = 0x00;

/* Generally: Whenever changing PLL reference clock (REFCLK) frequency to a higher value */
/* it is recommended to write CPMUSYNR = 0x00 in order to stay within specified */
/* maximum frequency of the MCU */
CPMUSYNR = 0x00;

/* configure PLL reference clock (REFCLK) for usage with Oscillator */
/* OSC=4MHz divide by 4 (3+1) = 1MHz, REFCLK range 1MHz to 2 MHz (REFFRQ[1:0] = 00) */
CPMUREFDV = 0x03;

/* enable external Oscillator, switch PLL reference clock (REFCLK) to OSC */
CPMUOSC = 0x80;

/* multiply REFCLK = 1MHz by 2*(24+1)*1MHz = 50MHz */
/* VCO range 48 to 80 MHz (VCOFRQ[1:0] = 01) */
CPMUSYNR = 0x58;

/* clear all flags, especially LOCKIF and OSCIF */
CPMUIFLG = 0xFF;

/* put your code to loop and wait for the LOCKIF and OSCIF or */
/* poll CPMUIFLG register until both UPOSC and LOCK status are “1” */
/* that is CPMIFLG == 0x1B */

/*..............continue to your main code execution here..............*/

/* in case later in your code you want to disable the Oscillator and use the */
/* 1MHz IRCCLK as PLL reference clock */

/* Generally: Whenever changing PLL reference clock (REFCLK) frequency to a higher value */
/* it is recommended to write CPMUSYNR = 0x00 in order to stay within specified */
/* maximum frequency of the MCU */
CPMUSYNR = 0x00;

/* disable OSC and switch PLL reference clock to IRC */
CPMUOSC = 0x00;

/* multiply REFCLK = 1MHz by 2*(24+1)*1MHz = 50MHz */
/* VCO range 48 to 80 MHz (VCOFRQ[1:0] = 01) */
CPMUSYNR = 0x58;

/* clear all flags, especially LOCKIF and OSCIF */
CPMUIFLG = 0xFF;

/* put your code to loop and wait for the LOCKIF or */
/* poll CPMUIFLG register until both LOCK status is “1” */
/* that is CPMIFLG == 0x18 */

/*..............continue to your main code execution here..............*/
```

11.4.2.1 Control Register (DACCTL)

Module Base + 0x0000 Access: User read/write[1]

	7	6	5	4	3	2	1	0
R	FVR	DRIVE	0	0	0	DACM[2:0]		
W								
Reset	1	0	0	0	0	0	0	0

= Unimplemented

Figure 11-3. Control Register (DACCTL)

[1] Read: Anytime
Write: Anytime

Table 11-3. DACCTL Field Description

Field	Description
7 FVR	**Full Voltage Range** — This bit defines the voltage range of the DAC. 0 DAC resistor network operates with the reduced voltage range 1 DAC resistor network operates with the full voltage range **Note:** For more details see Section 11.5.8, “Analog output voltage calculation”.
6 DRIVE	**Drive Select** — This bit selects the output drive capability of the operational amplifier, see electrical Spec. for more details. 0 Low output drive for high resistive loads 1 High output drive for low resistive loads
2:0 DACM[2:0]	**Mode Select** — These bits define the mode of the DAC. A write access with an unsupported mode will be ignored. 000 Off 001 Operational Amplifier 010 Internal DAC only 100 Unbuffered DAC 101 Unbuffered DAC with Operational Amplifier 111 Buffered DAC other Reserved

The MSCAN facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

13.4.2.1 Message Transmit Background

Modern application layer software is built upon two fundamental assumptions:

- Any CAN node is able to send out a stream of scheduled messages without releasing the CAN bus between the two messages. Such nodes arbitrate for the CAN bus immediately after sending the previous message and only release the CAN bus in case of lost arbitration.
- The internal message queue within any CAN node is organized such that the highest priority message is sent out first, if more than one message is ready to be sent.

The behavior described in the bullets above cannot be achieved with a single transmit buffer. That buffer must be reloaded immediately after the previous message is sent. This loading process lasts a finite amount of time and must be completed within the inter-frame sequence (IFS) to be able to send an uninterrupted stream of messages. Even if this is feasible for limited CAN bus speeds, it requires that the CPU reacts with short latencies to the transmit interrupt.

A double buffer scheme de-couples the reloading of the transmit buffer from the actual message sending and, therefore, reduces the reactiveness requirements of the CPU. Problems can arise if the sending of a message is finished while the CPU re-loads the second buffer. No buffer would then be ready for transmission, and the CAN bus would be released.

At least three transmit buffers are required to meet the first of the above requirements under all circumstances. The MSCAN has three transmit buffers.

The second requirement calls for some sort of internal prioritization which the MSCAN implements with the “local priority” concept described in Section 13.4.2.2, “Transmit Structures.”

13.4.2.2 Transmit Structures

The MSCAN triple transmit buffer scheme optimizes real-time performance by allowing multiple messages to be set up in advance. The three buffers are arranged as shown in Figure 13-38.

All three buffers have a 13-byte data structure similar to the outline of the receive buffers (see Section 13.3.3, “Programmer’s Model of Message Storage”). An additional Transmit Buffer Priority Register (TBPR) contains an 8-bit local priority field (PRIO) (see Section 13.3.3.4, “Transmit Buffer Priority Register (TBPR)”). The remaining two bytes are used for time stamping of a message, if required (see Section 13.3.3.5, “Time Stamp Register (TSRH–TSRL)”).

To transmit a message, the CPU must identify an available transmit buffer, which is indicated by a set transmitter buffer empty (TXEx) flag (see Section 13.3.2.7, “MSCAN Transmitter Flag Register (CANTFLG)”). If a transmit buffer is available, the CPU must set a pointer to this buffer by writing to the CANTBSEL register (see Section 13.3.2.11, “MSCAN Transmit Buffer Selection Register (CANTBSEL)”). This makes the respective buffer accessible within the CANTXFG address space (see Section 13.3.3, “Programmer’s Model of Message Storage”). The algorithmic feature associated with the CANTBSEL register simplifies the transmit buffer selection. In addition, this scheme makes the handler

The main element of the SPI system is the SPI data register. The n-bit[1] data register in the master and the n-bit[1] data register in the slave are linked by the MOSI and MISO pins to form a distributed 2n-bit[1] register. When a data transfer operation is performed, this 2n-bit[1] register is serially shifted n[1] bit positions by the S-clock from the master, so data is exchanged between the master and the slave. Data written to the master SPI data register becomes the output data for the slave, and data read from the master SPI data register after a transfer operation is the input data from the slave.

A read of SPISR with SPTEF = 1 followed by a write to SPIDR puts data into the transmit data register. When a transfer is complete and SPIF is cleared, received data is moved into the receive data register. This data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A common SPI data register address is shared for reading data from the read data buffer and for writing data to the transmit data register.

The clock phase control bit (CPHA) and a clock polarity control bit (CPOL) in the SPI control register 1 (SPICR1) select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. The CPHA bit is used to accommodate two fundamentally different protocols by sampling data on odd numbered SCK edges or on even numbered SCK edges (see Section 19.4.3, “Transmission Formats”).

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI control register1 is set, master mode is selected, when the MSTR bit is clear, slave mode is selected.

NOTE

A change of CPOL or MSTR bit while there is a received byte pending in the receive shift register will destroy the received byte and must be avoided.

19.4.1 Master Mode

The SPI operates in master mode when the MSTR bit is set. Only a master SPI module can initiate transmissions. A transmission begins by writing to the master SPI data register. If the shift register is empty, data immediately transfers to the shift register. Data begins shifting out on the MOSI pin under the control of the serial clock.

- Serial clock

  The SPR2, SPR1, and SPR0 baud rate selection bits, in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI baud rate register, control the baud rate generator and determine the speed of the transmission. The SCK pin is the SPI clock output. Through the SCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.
- MOSI, MISO pin

  In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.
- $\overline{SS}$ pin

  If MODFEN and SSOE are set, the $\overline{SS}$ pin is configured as slave select output. The $\overline{SS}$ output becomes low during each transmission and is high when the SPI is in idle state.

1. n depends on the selected transfer width, please refer to Section 19.3.2.2, “SPI Control Register 2 (SPICR2)

20.3.1.1 IIC Address Register (IBAD)

Module Base +0x0000

	7	6	5	4	3	2	1	0
R	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-3. IIC Bus Address Register (IBAD)

Read and write anytime

This register contains the address the IIC bus will respond to when addressed as a slave; note that it is not the address sent on the bus during the address transfer.

Table 20-2. IBAD Field Descriptions

Field	Description
7:1 ADR[7:1]	**Slave Address** — Bit 1 to bit 7 contain the specific slave address to be used by the IIC bus module.The default mode of IIC bus is slave mode for an address match on the bus.
0 Reserved	Reserved — Bit 0 of the IBAD is reserved for future compatibility. This bit will always read 0.

20.3.1.2 IIC Frequency Divider Register (IBFD)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-4. IIC Bus Frequency Divider Register (IBFD)

Read and write anytime

Table 20-3. IBFD Field Descriptions

Field	Description
7:0 IBC[7:0]	**I Bus Clock Rate 7:0** — This field is used to prescale the clock for bit rate selection. The bit clock generator is implemented as a prescale divider — IBC7:6, prescaled shift register — IBC5:3 select the prescaler divider and IBC2-0 select the shift register tap point. The IBC bits are decoded to give the tap and prescale values as shown in Table 20-4.

Table 20-7. IIC Divider and Hold Values (Sheet 2 of 6)

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
00	20/22	7	6	11
01	22/24	7	7	12
02	24/26	8	8	13
03	26/28	8	9	14
04	28/30	9	10	15
05	30/32	9	11	16
06	34/36	10	13	18
07	40/42	10	16	21
08	28/32	7	10	15
09	32/36	7	12	17
0A	36/40	9	14	19
0B	40/44	9	16	21
0C	44/48	11	18	23
0D	48/52	11	20	25
0E	56/60	13	24	29
0F	68/72	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289

F.1 Dynamic Electrical Characteristics

Table F-2. Dynamic Electrical Characteristics - Supply Voltage Sense - (BATS)

Characteristics noted under conditions $5.5V \leq V_{SUP} \leq 18V$, unless otherwise noted. Typical values noted reflect the approximate parameter mean at $T_A = 25°C$[1] under nominal conditions unless otherwise noted.						
Num	**Ratings**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	Enable Stabilization Time	T_{EN_UNC}	–	1	–	µs
2	Voltage Warning Low Pass Filter	f_{VWLP_filter}	–	0.5	–	Mhz

1 T_A: Ambient Temperature

Appendix N
Ordering Information

Customers can choose either the mask-specific partnumber or the generic, mask-independent partnumber. Ordering a mask-specific partnumber enables the customer to specify which particular maskset they receive whereas ordering the generic partnumber means that the currently preferred maskset (which may change over time) is shipped. In either case, the marking on the device always shows the generic, mask-independent partnumber and the mask set number. The below figure illustrates the structure of a typical mask-specific ordering number.

O.19 0x0780-0x0787 SPI0

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0780	SPI0CR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		W								
0x0781	SPI0CR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		W								
0x0782	SPI0BR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		W								
0x0783	SPI0SR	R	SPIF	0	SPTEF	MODF	0	0	0	0
		W								
0x0784	SPI0DRH	R	R15	R14	R13	R12	R11	R10	R9	R8
		W	T15	T14	T13	T12	T11	T10	T9	T8
0x0785	SPI0DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0
0x0786	Reserved	R								
		W								
0x0787	Reserved	R								
		W								