Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	HCS12
Core Size	16-Bit
Speed	25MHz
Connectivity	EBI/EMI, Ethernet, I²C, SCI, SPI
Peripherals	POR, PWM, WDT
Number of I/O	38
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2.375V ~ 3.465V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP Exposed Pad
Supplier Device Package	80-TQFP-EP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc9s12ne64vtue

$001F - $001F Interrupt Module (INT) Map 2 of 2

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$001F	HPRIO	Read:	PSEL7	PSEL6	PSEL5	PSEL4	PSEL3	PSEL2	PSEL1	0
		Write:								

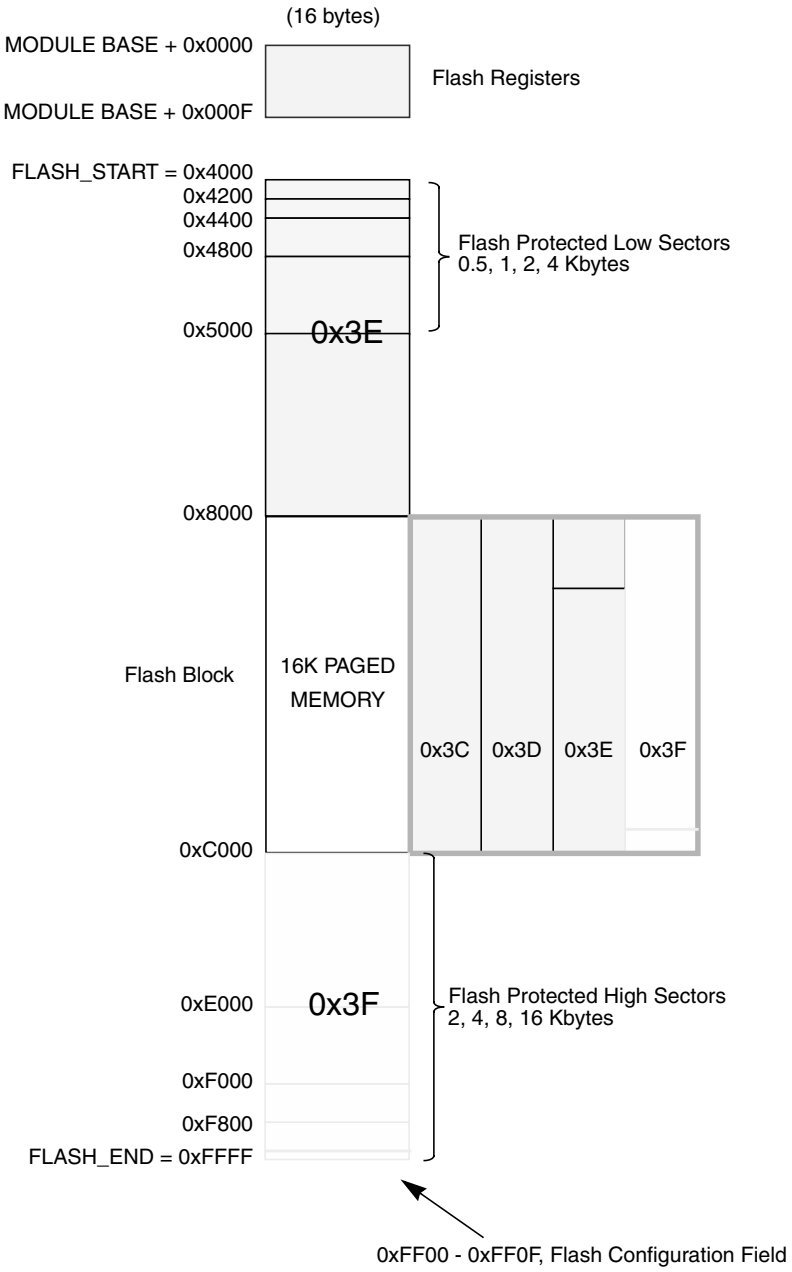
$0020 - $002F Debug Module (DBG) Including BKP Map 1 of 1

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0020	DBGC1	Read:	DBGEN	ARM	TRGSEL	BEGIN	DBGBRK	0	CAPMOD	
		Write:								
$0021	DBGSC	Read:	AF	BF	CF	0	TRG			
		Write:								
$0022	DBGTBH	Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Write:								
$0023	DBGTBL	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
$0024	DBGCNT	Read:	TBF	0	CNT					
		Write:								
$0025	DBGCCX	Read:	PAGSEL		EXTCMP					
		Write:								
$0026	DBGCCH	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$0027	DBGCCL	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$0028	DBGC2 (BKPCT0)[1]	Read:	BKABEN	FULL	BDM	TAGAB	BKCEN	TAGC	RWCEN	RWC
		Write:								
$0029	DBGC3 (BKPCT1)[1]	Read:	BKAMBH	BKAMBL	BKBMBH	BKBMBL	RWAEN	RWA	RWBEN	RWB
		Write:								
$002A	DBGCAX (BKP0X)[1]	Read:	PAGSEL		EXTCMP					
		Write:								
$002B	DBGCAH (BKP0H)[1]	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$002C	DBGCAL (BKP0L)[1]	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
$002D	DBGCBX (BKP1X)[1]	Read:	PAGSEL		EXTCMP					
		Write:								
$002E	DBGCBH (BKP1H)[1]	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
$002F	DBGCBL (BKP1L)[1]	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

[1]Legacy HCS12 MCUs used this name for this register.

$0140 - $016F Ethernet Media Access Controller (EMAC) (Continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$0156	MCMST	Read:	0	0	BUSY	NOPRE	MDCSEL			
		Write:	OP							
$0157	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
$0158	BUFCFG [15:8]	Read:	0	BUFMAP			0	MAXFL[10:8]		
		Write:								
$0159	BUFCFG [7:0]	Read:	MAXFL[7:0]							
		Write:								
$015A	RXAEFP [15:8]	Read:	0	0	0	0	0	RXAEFP[10:8]		
		Write:								
$015B	RXAEFP [7:0]	Read:	RXAEFP[7:0]							
		Write:								
$015C	RXBEFP [15:8]	Read:	0	0	0	0	0	RXBEFP[10:8]		
		Write:								
$015D	RXBEFP [7:0]	Read:	RXBEFP[7:0]							
		Write:								
$015E	TXEFP [15:8]	Read:	0	0	0	0	0	TXEFP[10:8]		
		Write:								
$015F	TXEFP	Read:	TXEFP[7:0]							
		Write:								
$0160	MCHASH	Read:	MCHASH[63:56]							
		Write:								
$0161	MCHASH	Read:	MCHASH[55:48]							
		Write:								
$0162	MCHASH	Read:	MCHASH[47:40]							
		Write:								
$0163	MCHASH	Read:	MCHASH[39:32]							
		Write:								
$0164	MCHASH	Read:	MCHASH[31:24]							
		Write:								
$0165	MCHASH	Read:	MCHASH[23:16]							
		Write:								
$0166	MCHASH	Read:	MCHASH[15:8]							
		Write:								
$0167	MCHASH	Read:	MCHASH[7:0]							
		Write:								
$0168	MACAD	Read:	MACAD[47:40]							
		Write:								
$0169	MACAD	Read:	MACAD[39:32]							
		Write:								
$016A	MACAD	Read:	MACAD[31:24]							
		Write:								
$016B	MACAD	Read:	MACAD[23:16]							
		Write:								
$016C	MACAD	Read:	MACAD[15:8]							
		Write:								



Note: 0x3C-0x3F correspond to the PPAGE register content

Figure 2-2. Flash Memory Map

2.4.1.1 Writing the FCLKDIV Register

Prior to issuing any program, erase, erase verify, or data compress command, it is first necessary to write the FCLKDIV register to divide the oscillator clock down to within the 150 kHz to 200 kHz range. Because the program and erase timings are also a function of the bus clock, the FCLKDIV determination must take this information into account.

If we define:

- FCLK as the clock of the Flash timing control block,
- Tbus as the period of the bus clock, and
- INT(x) as taking the integer part of x (e.g. INT(4.323)=4).

Then, FCLKDIV register bits PRDIV8 and FDIV[5:0] are to be set as described in Figure 2-21.

For example, if the oscillator clock frequency is 950 kHz and the bus clock frequency is 10 MHz, FCLKDIV bits FDIV[5:0] must be set to 4 (000100) and bit PRDIV8 set to 0. The resulting FCLK frequency is then 190 kHz. As a result, the Flash program and erase algorithm timings are increased over the optimum target by:

$$(200 - 190) / 200 \times 100 \; = \; 5\%$$

CAUTION

Program and erase command execution time will increase proportionally with the period of FCLK. Because of the impact of clock synchronization on the accuracy of the functional timings, programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Programming or erasing the Flash memory with FCLK < 150 kHz must be avoided. Setting FCLKDIV to a value such that FCLK < 150 kHz can destroy the Flash memory due to overstress. Setting FCLKDIV to a value such that (1/FCLK + Tbus) < 5µs can result in incomplete programming or erasure of the Flash memory cells.

If the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. Flash commands will not be executed if the FCLKDIV register has not been written to.

2.6.2 Unsecuring the Flash Module in Special Single-Chip Mode using BDM

The MCU can be unsecured in special single-chip mode by erasing the Flash module by the following method :

- Reset the MCU into special single-chip mode, delay while the erase test is performed by the BDM secure ROM, send BDM commands to disable protection in the Flash module, and execute a mass erase command write sequence to erase the Flash memory.

After the CCIF flag sets to indicate that the mass operation has completed, reset the MCU into special single-chip mode. The BDM secure ROM will verify that the Flash memory is erased and will assert the UNSEC bit in the BDM status register. This BDM action will cause the MCU to override the Flash security state and the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a word program sequence to program the Flash security byte to the unsecured state and reset the MCU.

2.7 Resets

2.7.1 Flash Reset Sequence

On each reset, the Flash module executes a reset sequence to hold CPU activity while loading the following registers from the Flash memory according to Table 2-1:

- FPROT — Flash Protection Register (see Section 2.3.2.5).
- FCTL — Flash Control Register (see Section 2.3.2.9).
- FSEC — Flash Security Register (see Section 2.3.2.2).

2.7.2 Reset While Flash Command Active

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector / block being erased is not guaranteed.

2.8 Interrupts

The Flash module can generate an interrupt when all Flash command operations have completed, when the Flash address, data, and command buffers are empty.

Table 2-19. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash address, data and command buffers empty	CBEIF (FSTAT register)	CBEIE (FCNFG register)	I Bit
All Flash commands completed	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit

3.3.2.2.3 Data Direction Register (DDRS)

Module Base + $A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
Reset:	0	0	0	0	0	0	0	0

Figure 3-10. Port S Data Direction Register (DDRS)

Read:Anytime.

Write:Anytime.

This register configures each port S pin as either input or output.

If the SPI is enabled, the SPI controls the SPI related pins (SPI_$\overline{SS}$, SPI_SCK, SPI_MOSI, SPI_MISO) I/O direction, and the corresponding DDRS[7:4] bits have no effect on the SPI pins I/O direction. Refer to the SPI block description chapter for details.

When the SCI0 or SCI1 transmitters are enabled, the corresponding transmit pins, SCI0_TxD and SCI0_TxD, I/O direction is controlled by the SCI0 and SCI1 respectively, and the corresponding DDRS3 and DDRS1 bits have no effect on their I/O direction. When the SCI0 or SCI1 receivers are enabled, the corresponding receive pins, SCI0_RXD and SCI1_RXD, I/O direction is controlled by the SCI0 and SCI1 respectively, and the DDRS2 and DDRS0 bits have no effect on their I/O direction. Refer to the SCI block description chapter for further details.

The DDRS[7:0] bits revert to controlling the I/O direction of the pins when the associated SPI or SCI function is disabled.

DDRS[7:0] — Data Direction Port S

1 = Associated pin is configured as output.
0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTS or PTIS registers, when changing the DDRS register.

3.3.2.2.4 Reduced Drive Register (RDRS)

Module Base + $B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
Reset:	0	0	0	0	0	0	0	0

Figure 3-11. Port S Reduced Drive Register (RDRS)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port S output pin as either full or reduced. If the port is used as input this bit is ignored.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
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

= Unimplemented or Reserved

Figure 4-3. CRG Register Summary (continued)

4.3.2.1 CRG Synthesizer Register (SYNR)

The SYNR register controls the multiplication factor of the PLL. If the PLL is on, the count in the loop divider (SYNR) register effectively multiplies up the PLL clock (PLLCLK) from the reference frequency by 2 x (SYNR+1). PLLCLK will not be below the minimum VCO frequency (f_{SCM}).

$$PLLCLK = 2 \times OSCCLK \times \frac{(SYNR + 1)}{(REFDV + 1)}$$

NOTE

If PLL is selected (PLLSEL=1), Bus Clock = PLLCLK / 2
Bus Clock must not exceed the maximum operating system frequency.

	7	6	5	4	3	2	1	0
R	0	0	SYN5	SYNR	SYN3	SYN2	SYN1	SYN0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-4. CRG Synthesizer Register (SYNR)

Read: anytime

Write: anytime except if PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit and the track detector bit.

Table 4-4. CLKSEL Field Descriptions (continued)

Field	Description
1 RTIWAI	**RTI Stops in Wait Mode Bit** — Write: anytime 0 RTI keeps running in wait mode. 1 RTI stops and initializes the RTI dividers whenever the part goes into wait mode.
0 COPWAI	**COP Stops in Wait Mode Bit** — Normal modes: Write once —Special modes: Write anytime 0 COP keeps running in wait mode. 1 COP stops and initializes the COP dividers whenever the part goes into wait mode.

4.3.2.7 CRG PLL Control Register (PLLCTL)

This register controls the PLL functionality.

	7	6	5	4	3	2	1	0
R	CME	PLLON	AUTO	ACQ	0	PRE	PCE	SCME
W								
Reset	1	1	1	1	0	0	0	1

= Unimplemented or Reserved

Figure 4-10. CRG PLL Control Register (PLLCTL)

Read: anytime

Write: refer to each bit for individual write conditions

Table 4-5. PLLCTL Field Descriptions

Field	Description
7 CME	**Clock Monitor Enable Bit** — CME enables the clock monitor. Write anytime except when SCM = 1. 0 Clock monitor is disabled. 1 Clock monitor is enabled. Slow or stopped clocks will cause a clock monitor reset sequence or self-clock mode. **Note:** Operating with CME = 0 will not detect any loss of clock. In case of poor clock quality this could cause unpredictable operation of the MCU. **Note:** In Stop Mode (PSTP = 0) the clock monitor is disabled independently of the CME bit setting and any loss of clock will not be detected.
6 PLLON	**Phase Lock Loop On Bit** — PLLON turns on the PLL circuitry. In self-clock mode, the PLL is turned on, but the PLLON bit reads the last latched value. Write anytime except when PLLSEL = 1. 0 PLL is turned off. 1 PLL is turned on. If AUTO bit is set, the PLL will lock automatically.
5 AUTO	**Automatic Bandwidth Control Bit** — AUTO selects either the high bandwidth (acquisition) mode or the low bandwidth (tracking) mode depending on how close to the desired frequency the VCO is running. Write anytime except when PLLWAI=1, because PLLWAI sets the AUTO bit to 1. 0 Automatic mode control is disabled and the PLL is under software control, using ACQ bit. 1 Automatic mode control is enabled and ACQ bit has no effect.
4 ACQ	**Acquisition Bit** — Write anytime. If AUTO=1 this bit has no effect. 0 Low bandwidth filter is selected. 1 High bandwidth filter is selected.

6.4 Functional Description

This section provides a complete functional description of the timer TIM16B4CV1 block. Please refer to the detailed timer block diagram in Figure 6-26 as necessary.

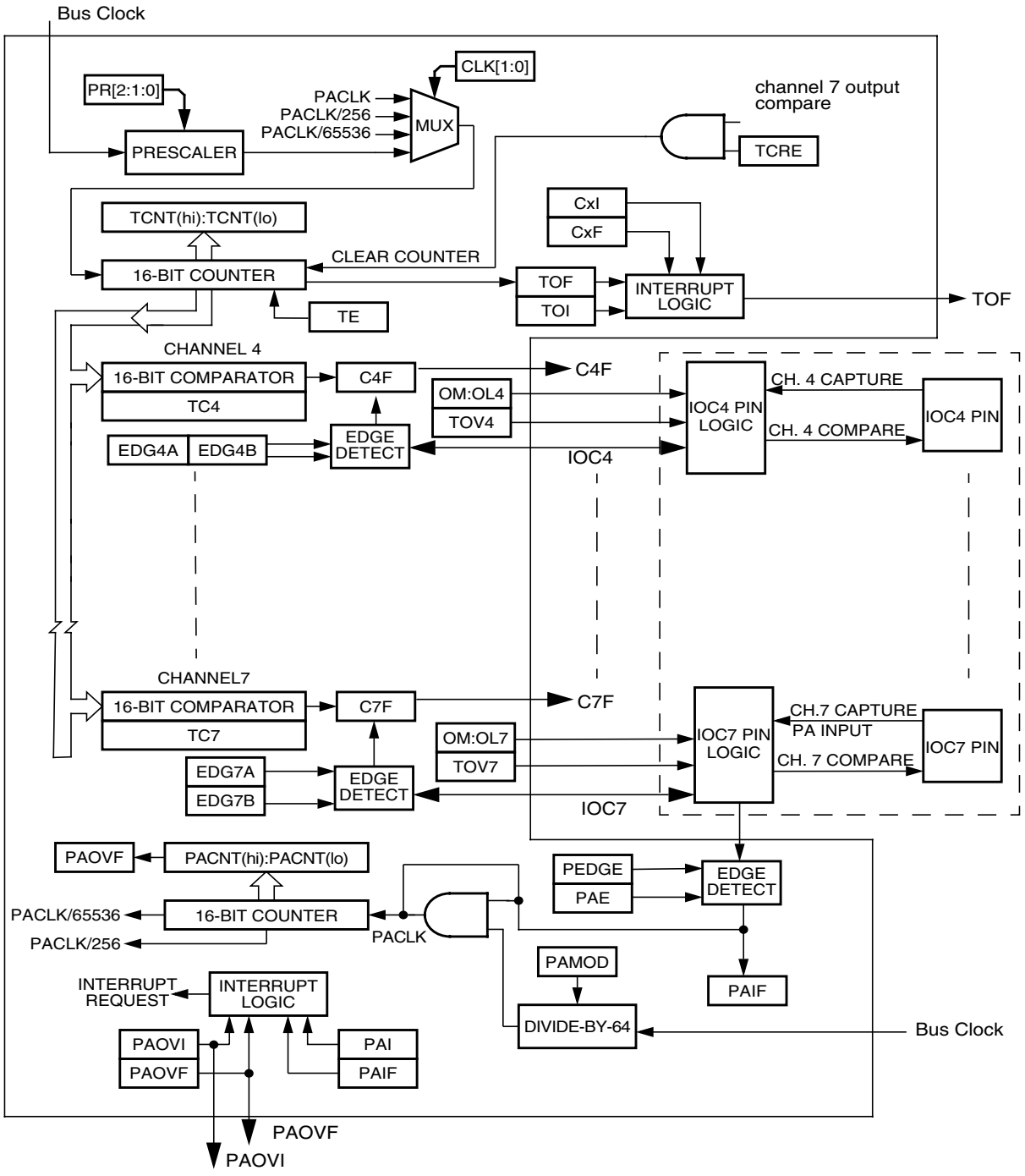


Figure 6-26. Detailed Timer Block Diagram

9.2.2 MISO — Master In/Slave Out Pin

This pin is used to transmit data out of the SPI module when it is configured as a slave and receive data when it is configured as master.

9.2.3 $\overline{SS}$ — Slave Select Pin

This pin is used to output the select signal from the SPI module to another peripheral with which a data transfer is to take place when its configured as a master and its used as an input to receive the slave select signal when the SPI is configured as slave.

9.2.4 SCK — Serial Clock Pin

This pin is used to output the clock with respect to which the SPI transfers data or receive clock in case of slave.

9.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the SPI.

The memory map for the SPIV3 is given below in Table 9-1. The address listed for each register is the sum of a base address and an address offset. The base address is defined at the SoC level and the address offset is defined at the module level. Reads from the reserved bits return zeros and writes to the reserved bits have no effect.

9.3.1 Module Memory Map

Table 9-1. SPIV3 Memory Map

Address	Use	Access
0x0000	SPI Control Register 1 (SPICR1)	R/W
0x0001	SPI Control Register 2 (SPICR2)	R/W[1]
0x0002	SPI Baud Rate Register (SPIBR)	R/W[1]
0x0003	SPI Status Register (SPISR)	R[2]
0x0004	Reserved	—[2,3]
0x0005	SPI Data Register (SPIDR)	R/W
0x0006	Reserved	—[2,3]
0x0007	Reserved	—[2,3]

1 Certain bits are non-writable.
2 Writes to this register are ignored.
3 Reading from this register returns all zeros.

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.

9.4.3.3 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 8-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.

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the SPI shift register, depending on LSBFE bit. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pin on the slave.

This process continues for a total of 16 edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

Data reception is double buffered, data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI Data Register after the last bit is shifted in.

After the 16th SCK edge:

- Data that was previously in the SPI Data Register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

Figure 9-10 shows two clocking variations for CPHA = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The $\overline{SS}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

The $\overline{SS}$ line can remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave that drive the MISO data line.

- Back-to-back transfers in master mode

In master mode, if a transmission has completed and a new data byte is available in the SPI Data Register, this byte is send out immediately without a trailing and minimum idle time.

The SPI interrupt request flag (SPIF) is common to both the master and slave modes. SPIF gets set one half SCK cycle after the last SCK edge.

11.2.1 MII_TXCLK — MII Transmit Clock

The PHY provides this input clock, which is used as a timing reference for MII_TXD, MII_TXEN, and MII_TXER. It operates at 25% of the transmit data rate (25 MHz for 100 Mbps or 2.5 MHz for 10 Mbps). The EMAC bus clock frequency must be greater-than or equal-to MII_TXCLK.

11.2.2 MII_TXD[3:0] — MII Transmit Data

MII_TXD[3:0] is a transmit nibble of data to be transferred from the EMAC to the PHY. The nibble is synchronized to the rising edge of MII_TXCLK. When MII_TXEN is asserted, the PHY accepts MII_TXD[3:0], and at all other times, MII_TXD[3:0] is ignored. MII_TXD[0] is the least significant bit.

Table 11-1 summarizes the permissible encoding of MII_TXD[3:0], MII_TXEN, and MII_TXER.

Table 11-1. Permissible Encoding of MII_TXD, MII_TXEN, and MII_TXER

MII_TXEN	MII_TXER	MII_TXD[3:0]	Indication
0	0	0000 through 1111	Normal interframe
0	1	0000 through 1111	Reserved
1	0	0000 through 1111	Normal data transmission
1	1	0000 through 1111	Transmit error propagation

11.2.3 MII_TXEN — MII Transmit Enable

Assertion of this output signal indicates that there are valid nibbles being presented on the MII and the transmission can start. This signal is asserted with the first nibble of the preamble, remains asserted until all nibbles to be transmitted have been presented to the PHY, and is negated following the final nibble of the frame.

11.2.4 MII_TXER — MII Transmit Coding Error

Assertion of this output signal for one or more clock cycles while MII_TXEN is asserted causes the PHY to transmit one or more illegal symbols. MII_TXER is asserted if the ABORT command is issued during a transmit. This signal transitions synchronously with respect to MII_TXCLK.

11.2.5 MII_RXCLK — MII Receive Clock

The PHY provides this input clock, which is used as a timing reference for MII_RXD, MII_RXDV, and MII_RXER. It operates at 25% of the receive data rate (25 MHz for 100 Mbps or 2.5 MHz for 10 Mbps). The EMAC bus clock frequency must be greater-than or equal-to MII_RXCLK.

ANDIS — Auto Negotiation Disable

This bit can be written anytime, but the value is latched in the ANE bit of the MII PHY control register (MII address 0.12) only when the EPHYEN bit transitions from 0 to 1.

1 = Auto negotiation is disabled after start-up. A 0 is latched in the ANE bit of the MII PHY control register (MII address 0.12), and upon completion of the start-up delay ($t_{Start-up}$), the EPHY will bypass auto-negotiation. The mode of operation will be determined by the manual setting of MII registers.

0 = Auto negotiation is enabled after start-up. A 1 is latched in the ANE bit of the MII PHY control register (MII address 0.12), and upon completion of the start-up delay ($t_{Start-up}$), the EPHY will enter auto-negotiation. The mode of operation will be automatically determined.

DIS100 — Disable 100 BASE-TX PLL

This bit can be written anytime. Allows user to power down the clock generation PLL for 100BASE-TX clocks.

1 = Disables 100BASE-TX PLL

0 = 100BASE-TX PLL state determined by EPHY operation mode

DIS10 — Disable 10BASE-T PLL

This bit can be written anytime. Allows user to power down the clock generation PLL for 10BASE-T clocks.

1 = Disables 10BASE-T PLL

0 = 10 BASE-T PLL state determined by EPHY operation mode

LEDEN — LED Drive Enable

This bit can be written anytime.

1 = Enables the EPHY to drive LED signals.

0 = Disables the EPHY to drive LED signals.

EPHYWAI — EPHY Module Stops While in Wait

This bit can be written anytime.

1 = Disables the EPHY module while the MCU is in wait mode. EPHY interrupts cannot be used to bring the MCU out of wait.

0 = Allows the EPHY module to continue running during wait.

EPHYIEN — EPHY Interrupt Enable

This bit can be written anytime.

1 = Enables EPHY module interrupts

0 = Disables EPHY module interrupts

12.3.2.2 Ethernet Physical Transceiver Control Register 1 (EPHYCTL1)

Module Base + $1

	7	6	5	4	3	2	1	0
R	0	0	0	PHYADD4	PHYADD3	PHYADD2	PHYADD1	PHYADD0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-4. Ethernet Physical Transceiver Control Register 1 (EPHYCTL1)

13.4.3 POR - Power-On Reset

This functional block monitors output VDD. If V_{DD} is below V_{PORD}, signal POR is high, if it exceeds V_{PORD}, the signal goes low. The transition to low forces the CPU in the power-on sequence.

Due to its role during chip power-up this module must be active in all operating modes of VREG_PHY.

13.4.4 LVR - Low Voltage Reset

Block LVR monitors the primary output voltage V_{DD}. If it drops below the assertion level (V_{LVRA}) signal LVR asserts and when rising above the deassertion level (V_{LVRD}) signal LVR negates again. The LVR function is available only in Full Performance Mode.

13.4.5 CTRL - Regulator Control

This part contains the register block of VREG_PHY and further digital functionality needed to control the operating modes. CTRL also represents the interface to the digital core logic.

13.5 Resets

13.5.1 General

This section describes how VREG_PHY controls the reset of the MCU.The reset values of registers and signals are provided in Section 13.3, “Memory Map and Registers.” Possible reset sources are listed in Table 13-2.

Table 13-2. VREG_PHY - Reset Sources

Reset Source	Local Enable
Power-on Reset	always active
Low Voltage Reset	available only in Full Performance Mode

13.5.2 Description of Reset Operation

13.5.2.1 Power-On Reset

During chip power-up the digital core may not work if its supply voltage V_{DD} is below the POR deassertion level (V_{PORD}). Therefore signal POR which forces the other blocks of the device into reset is kept high until V_{DD} exceeds V_{PORD}. Then POR becomes low and the reset generator of the device continues the start-up sequence. The power-on reset is active in all operation modes of VREG_PHY.

Table 14-4. HPRIO Field Descriptions

Field	Description
7:1 PSEL[7:1]	**Highest Priority I Interrupt Select Bits** — The state of these bits determines which I-bit maskable interrupt will be promoted to highest priority (of the I-bit maskable interrupts). To promote an interrupt, the user writes the least significant byte of the associated interrupt vector address to this register. If an unimplemented vector address or a non I-bit masked vector address (value higher than 0x00F2) is written, IRQ (0xFFF2) will be the default highest priority interrupt.

14.4 Functional Description

The interrupt sub-block processes all exception requests made by the CPU. These exceptions include interrupt vector requests and reset vector requests. Each of these exception types and their overall priority level is discussed in the subsections below.

14.4.1 Low-Power Modes

The INT does not contain any user-controlled options for reducing power consumption. The operation of the INT in low-power modes is discussed in the following subsections.

14.4.1.1 Operation in Run Mode

The INT does not contain any options for reducing power in run mode.

14.4.1.2 Operation in Wait Mode

Clocks to the INT can be shut off during system wait mode and the asynchronous interrupt path will be used to generate the wake-up signal upon recognition of a valid interrupt or any $\overline{\text{XIRQ}}$ request.

14.4.1.3 Operation in Stop Mode

Clocks to the INT can be shut off during system stop mode and the asynchronous interrupt path will be used to generate the wake-up signal upon recognition of a valid interrupt or any $\overline{\text{XIRQ}}$ request.

14.5 Resets

The INT supports three system reset exception request types: normal system reset or power-on-reset request, crystal monitor reset request, and COP watchdog reset request. The type of reset exception request must be decoded by the system and the proper request made to the core. The INT will then provide the service routine address for the type of reset requested.

14.6 Interrupts

As shown in the block diagram in Figure 14-1, the INT contains a register block to provide interrupt status and control, an optional highest priority I interrupt (HPRIO) block, and a priority decoder to evaluate whether pending interrupts are valid and assess their priority.

14.6.1 Interrupt Registers

The INT registers are accessible only in special modes of operation and function as described in Section 14.3.2.1, “Interrupt Test Control Register,” and Section 14.3.2.2, “Interrupt Test Registers,” previously.

14.6.2 Highest Priority I-Bit Maskable Interrupt

When the optional HPRIO block is implemented, the user is allowed to promote a single I-bit maskable interrupt to be the highest priority I interrupt. The HPRIO evaluates all interrupt exception requests and passes the HPRIO vector to the priority decoder if the highest priority I interrupt is active. RTI replaces the promoted interrupt source.

14.6.3 Interrupt Priority Decoder

The priority decoder evaluates all interrupts pending and determines their validity and priority. When the CPU requests an interrupt vector, the decoder will provide the vector for the highest priority interrupt request. Because the vector is not supplied until the CPU requests it, it is possible that a higher priority interrupt request could override the original exception that 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 instead of the original request.

NOTE

Care must be taken to ensure that all exception requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not be processed.

If for any reason the interrupt source is unknown (e.g., an interrupt request becomes inactive after the interrupt has been recognized but prior to the vector request), the vector address will default to that of the last valid interrupt that existed during the particular interrupt sequence. If the CPU requests an interrupt vector when there has never been a pending interrupt request, the INT will provide the software interrupt (SWI) vector address.

14.7 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the INT upon request by the CPU is shown in Table 14-5.

Table 14-5. Exception Vector Map and Priority

Vector Address	Source
0xFFFE–0xFFFF	System reset
0xFFFC–0xFFFD	Crystal monitor reset
0xFFFA–0xFFFB	COP reset
0xFFF8–0xFFF9	Unimplemented opcode trap
0xFFF6–0xFFF7	Software interrupt instruction (SWI) or BDM vector request
0xFFF4–0xFFF5	$\overline{\text{XIRQ}}$ signal

Table 16-17. External/Internal Page Window Access

pag_sw1:pag_sw0	Partitioning	PIX5:0 Value	Page Window Access
00	876K off-Chip, 128K on-Chip	0x0000–0x0037	External
		0x0038–0x003F	Internal
01	768K off-chip, 256K on-chip	0x0000–0x002F	External
		0x0030–0x003F	Internal
10	512K off-chip, 512K on-chip	0x0000–0x001F	External
		0x0020–0x003F	Internal
11	0K off-chip, 1M on-chip	N/A	External
		0x0000–0x003F	Internal

NOTE

The partitioning as defined in Table 16-17 applies only to the allocated memory space and the actual on-chip memory sizes implemented in the system may differ. Please refer to the device overview chapter for actual sizes.

The PPAGE register holds the page select value for the program page window. The value of the PPAGE register can be manipulated by normal read and write (some devices don't allow writes in some modes) instructions as well as the CALL and RTC instructions.

Control registers, vector spaces, and a portion of on-chip memory are located in unpaged portions of the 64K byte physical address space. The stack and I/O addresses should also be in unpaged memory to make them accessible from any page.

The starting address of a service routine must be located in unpaged memory because the 16-bit exception vectors cannot point to addresses in paged memory. However, a service routine can call other routines that are in paged memory. The upper 16K byte block of memory space (0xC000–0xFFFF) is unpaged. It is recommended that all reset and interrupt vectors point to locations in this area.

16.4.3.1 CALL and Return from Call Instructions

CALL and RTC are uninterruptable instructions that automate page switching in the program expansion window. CALL is similar to a JSR instruction, but the subroutine that is called can be located anywhere in the normal 64K byte address space or on any page of program expansion memory. CALL calculates and stacks a return address, stacks the current PPAGE value, and writes a new instruction-supplied value to PPAGE. The PPAGE value controls which of the 64 possible pages is visible through the 16K byte expansion window in the 64K byte memory map. Execution then begins at the address of the called subroutine.

During the execution of a CALL instruction, the CPU:

- Writes the old PPAGE value into an internal temporary register and writes the new instruction-supplied PPAGE value into the PPAGE register.

Table 18-14. DBGC2 Field Descriptions (continued)

Field	Description
1 RWCEN	**Read/Write Comparator C Enable Bit** — The RWCEN bit controls whether read or write comparison is enabled for comparator C. RWCEN is not useful for tagged breakpoints. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
0 RWC	**Read/Write Comparator C Value Bit** — The RWC bit controls whether read or write is used in compare for comparator C. The RWC bit is not used if RWCEN = 0. 0 Write cycle will be matched 1 Read cycle will be matched

18.3.2.8 Debug Control Register 3 (DBGC3)

	7	6	5	4	3	2	1	0
R W	BKAMBH[1]	BKAMBL[1]	BKBMBH[2]	BKBMBL[2]	RWAEN	RWA	RWBEN	RWB
Reset	0	0	0	0	0	0	0	0

1 In DBG mode, BKAMBH:BKAMBL has no meaning and are forced to 0's.

2 In DBG mode, BKBMBH:BKBMBL are used in full mode to qualify data.

Figure 18-14. Debug Control Register 3 (DBGC3)

Table 18-15. DBGC3 Field Descriptions

Field	Description
7:6 BKAMB[H:L]	**Breakpoint Mask High Byte for First Address** — In dual or full mode, these bits may be used to mask (disable) the comparison of the high and/or low bytes of the first address breakpoint. The functionality is as given in Table 18-16. The x:0 case is for a full address compare. When a program page is selected, the full address compare will be based on bits for a 20-bit compare. The registers used for the compare are {DBGCAX[5:0], DBGCAH[5:0], DBGCAL[7:0]}, where DBGAX[5:0] corresponds to PPAGE[5:0] or extended address bits [19:14] and CPU address [13:0]. When a program page is not selected, the full address compare will be based on bits for a 16-bit compare. The registers used for the compare are {DBGCAH[7:0], DBGCAL[7:0]} which corresponds to CPU address [15:0]. **Note:** This extended address compare scheme causes an aliasing problem in BKP mode in which several physical addresses may match with a single logical address. This problem may be avoided by using DBG mode to generate breakpoints. The 1:0 case is not sensible because it would ignore the high order address and compare the low order and expansion addresses. Logic forces this case to compare all address lines (effectively ignoring the BKAMBH control bit). The 1:1 case is useful for triggering a breakpoint on any access to a particular expansion page. This only makes sense if a program page is being accessed so that the breakpoint trigger will occur only if DBGCAX compares.

A.17 Voltage Regulator Operating Characteristics

This section describes the characteristics of the on-chip voltage regulator (VREG_PHY).

Table A-30. VREG_PHY - Operating Conditions

Num	C	Characteristic	Symbol	Min	Typical	Max	Unit
1	P	Input Voltages	$V_{VDDR,A,X1,X2}$	3.135	—	3.465	V
2	P	Regulator Current Reduced Power Mode Shutdown Mode	I_{REG}	 — —	 20 12	 50 40	 µA µA
3	P	Output Voltage Core Full Performance Mode Reduced Power Mode Shutdown Mode	V_{DD}	 2.375 1.6 —	 2.5 2.5 [1] —	 2.625 2.75 —	 V V V
4	P	Output Voltage PLL Full Performance Mode Reduced Power Mode [2] Shutdown Mode	V_{DDPLL}	 2.375 1.6 —	 2.5 2.5 [3] —	 2.625 2.75 —	 V V V
5	P	Low Voltage Reset [4] Assert Level Deassert Level	 V_{LVRA} V_{LVRD}	 2.25 —	 — —	 — 2.55	 V V
7	C	Power-on Reset [5] Assert Level Deassert Level	 V_{PORA} V_{PORD}	 0.97 —	 --- ---	 — 2.05	 V V

1 High Impedance Output

2 Current I_{DDPLL} = 3 mA (Pierce Oscillator)

3 High Impedance Output

4 Monitors V_{DD}, active only in Full Performance Mode. V_{LVRA} and V_{PORD} must overlap

5 Monitors V_{DD}. Active in all modes.

The electrical characteristics given in this section are preliminary and should be used as a guide only. Values in this section cannot be guaranteed by Freescale Semiconductor and are subject to change without notice.

A.17.1 MCU Power-Up and LVR Graphical Explanation

Voltage regulator sub modules POR (power-on reset) and LVR (low-voltage reset) handle chip power-up or drops of the supply voltage. Their function is described in Figure A-15.