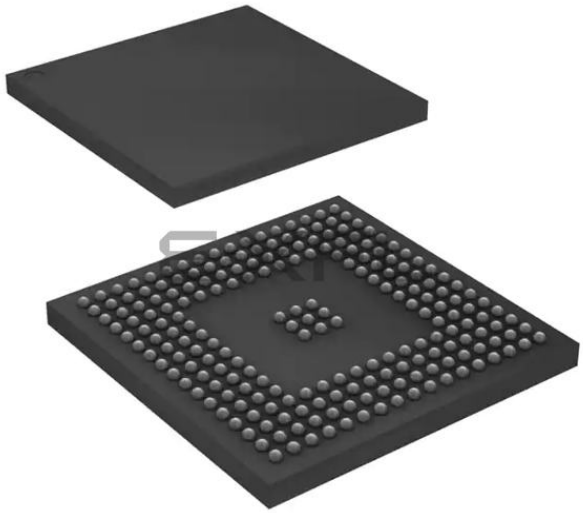


Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Active
Core Processor	ARM926EJ-S
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	400MHz
Co-Processors/DSP	-
RAM Controllers	LPDDR, LPSDR, DDR2, SDR, SRAM
Graphics Acceleration	No
Display & Interface Controllers	LCD, Touchscreen
Ethernet	-
SATA	-
USB	USB 2.0 (3)
Voltage - I/O	1.8V, 3.3V
Operating Temperature	-40°C ~ 85°C (TA)
Security Features	-
Package / Case	217-LFBGA
Supplier Device Package	217-LFBGA (15x15)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at91sam9g15-cu

8.6.3 Translation Table Walk Hardware

The translation table walk hardware is a logic that traverses the translation tables located in physical memory, gets the physical address and access permissions and updates the TLB.

The number of stages in the hardware table walking is one or two depending whether the address is marked as a section-mapped access or a page-mapped access.

There are three sizes of page-mapped accesses and one size of section-mapped access. Page-mapped accesses are for large pages, small pages and tiny pages. The translation process always begins with a level one fetch. A section-mapped access requires only a level one fetch, but a page-mapped access requires an additional level two fetch. For further details on the MMU, please refer to chapter 3 in ARM926EJ-S Technical Reference Manual.

8.6.4 MMU Faults

The MMU generates an abort on the following types of faults:

- Alignment faults (for data accesses only)
- Translation faults
- Domain faults
- Permission faults

The access control mechanism of the MMU detects the conditions that produce these faults. If the fault is a result of memory access, the MMU aborts the access and signals the fault to the CPU core.The MMU retains status and address information about faults generated by the data accesses in the data fault status register and fault address register. It also retains the status of faults generated by instruction fetches in the instruction fault status register.

The fault status register (register 5 in CP15) indicates the cause of a data or prefetch abort, and the domain number of the aborted access when it happens. The fault address register (register 6 in CP15) holds the MVA associated with the access that caused the Data Abort. For further details on MMU faults, please refer to chapter 3 in ARM926EJ-S Technical Reference Manual.

8.7 Caches and Write Buffer

The ARM926EJ-S contains a 16KB Instruction Cache (ICache), a 16KB Data Cache (DCache), and a write buffer. Although the ICache and DCache share common features, each still has some specific mechanisms.

The caches (ICache and DCache) are four-way set associative, addressed, indexed and tagged using the Modified Virtual Address (MVA), with a cache line length of eight words with two dirty bits for the DCache. The ICache and DCache provide mechanisms for cache lockdown, cache pollution control, and line replacement.

A new feature is now supported by ARM926EJ-S caches called allocate on read-miss commonly known as wrapping. This feature enables the caches to perform critical word first cache refilling. This means that when a request for a word causes a read-miss, the cache performs an AHB access. Instead of loading the whole line (eight words), the cache loads the critical word first, so the processor can reach it quickly, and then the remaining words, no matter where the word is located in the line.

The caches and the write buffer are controlled by the CP15 register 1 (Control), CP15 register 7 (cache operations) and CP15 register 9 (cache lockdown).

8.7.1 Instruction Cache (ICache)

The ICache caches fetched instructions to be executed by the processor. The ICache can be enabled by writing 1 to I bit of the CP15 Register 1 and disabled by writing 0 to this same bit.

When the MMU is enabled, all instruction fetches are subject to translation and permission checks. If the MMU is disabled, all instructions fetches are cachable, no protection checks are made and the physical address is flat-mapped to the modified virtual address. With the MVA use disabled, context switching incurs ICache cleaning and/or invalidating.

17.7.2 Shutdown Mode Register

Name: SHDW_MR

Address: 0xFFFFFE14

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	RTCWKEN	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
CPTWK0				–	–	WKMODE0	

- **WKMODE0: Wake-up Mode 0**

Value	Name	Description
0	NO_DETECTION	No detection is performed on the wake-up input
1	RISING_EDGE	Low to high transition triggers the detection process
2	FALLING_EDGE	High to low level transition triggers the detection process
3	ANY_EDGE	Any edge on the wake-up input triggers the detection process

- **CPTWK0: Debounce Counter on Wake-up 0**

Defines the minimum duration of the WKUP1 pin after the occurence of the selected triggering edge (WKMODE0).

The SHDN pin is released if the WKUP0 holds the selected level for (CPTWK $\times$ 16 + 1) consecutive Slow Clock cycles after the occurence of the selected triggering edge on WKUP0.

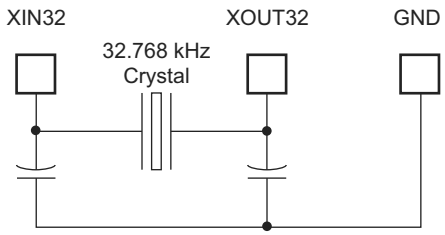
- **RTCWKEN: Real-time Clock Wake-up Enable**

0: The RTC Alarm signal has no effect on the Shutdown Controller.

1: The RTC Alarm signal forces the de-assertion of the SHDN pin.

Note that the user is not obliged to use the 32.768 kHz crystal oscillator and can use the 32 kHz (typical) RC oscillator instead.

Figure 20-2. Typical 32.768 kHz Crystal Oscillator Connection



The 32.768 kHz crystal oscillator provides a more accurate frequency than the 32 kHz (typical) RC oscillator.

To select the 32.768 kHz crystal oscillator as the source of the slow clock, the bit SCKC_CR.OSCSEL must be set. This results in a sequence which enables the 32.768 kHz crystal oscillator. The switch of the slow clock source is glitch-free.

The user can also set the 32.768 kHz crystal oscillator in Bypass mode instead of connecting a crystal. In this case, the user must provide the external clock signal on XIN32. The input characteristics of the XIN32 pin are given in the section “Electrical Characteristics”. To enter Bypass mode, the SCKC_CR.OSC32BYP must be set prior to setting SCKC_CR.OSCSEL.

21.17.10 PMC Clock Generator PLLA Register

Name: CKGR_PLLAR

Address: 0xFFFFFC28

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ONE	–	–	–	–	–

23	22	21	20	19	18	17	16
MULA							

15	14	13	12	11	10	9	8
OUTA		PLLACOUNT					

7	6	5	4	3	2	1	0
DIVA							

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Possible limitations on PLL input frequencies and multiplier factors should be checked before using the PMC.

- **DIVA: Divider A**

Value	Name	Description
0	0	Divider output is 0
1	BYPASS	Divider is bypassed
2–255	–	Divider output is the selected clock divided by DIVA.

- **PLLACOUNT: PLLA Counter**

Specifies the number of slow clock cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

- **OUTA: PLLA Clock Frequency Range**

To optimize clock performance, this field must be programmed as specified in “PLL Characteristics” in the Electrical Characteristics section of the product datasheet.

- **MULA: PLLA Multiplier**

0: The PLLA is deactivated.

1–254: The PLLA Clock frequency is the PLLA input frequency multiplied by MULA + 1.

- **ONE: Must Be Set to 1**

Bit 29 must always be set to 1 when programming the CKGR_PLLAR.

22.6.47 PIO Schmitt Trigger Register

Name: PIO_SCHMITT

Address: 0xFFFFF500 (PIOA), 0xFFFFF700 (PIOB), 0xFFFFF900 (PIOC), 0xFFFFFB00 (PIOD)

Access: Read/Write

31	30	29	28	27	26	25	24
SCHMITT31	SCHMITT30	SCHMITT29	SCHMITT28	SCHMITT27	SCHMITT26	SCHMITT25	SCHMITT24
23	22	21	20	19	18	17	16
SCHMITT23	SCHMITT22	SCHMITT21	SCHMITT20	SCHMITT19	SCHMITT18	SCHMITT17	SCHMITT16
15	14	13	12	11	10	9	8
SCHMITT15	SCHMITT14	SCHMITT13	SCHMITT12	SCHMITT11	SCHMITT10	SCHMITT9	SCHMITT8
7	6	5	4	3	2	1	0
SCHMITT7	SCHMITT6	SCHMITT5	SCHMITT4	SCHMITT3	SCHMITT2	SCHMITT1	SCHMITT0

- **SCHMITTx [x=0..31]: Schmitt Trigger Control**

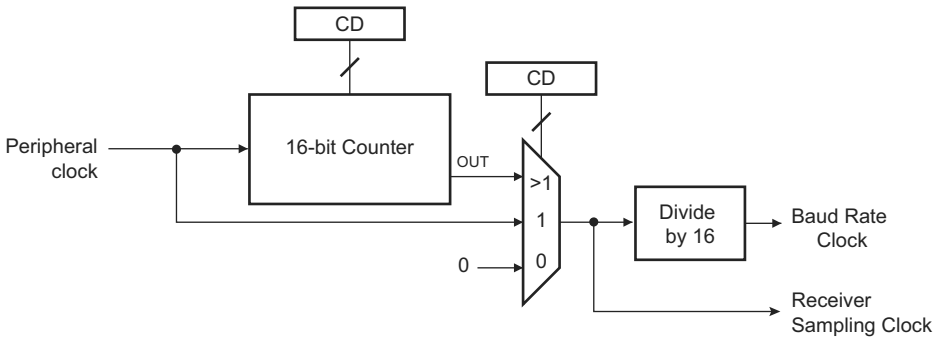
0: Schmitt trigger is enabled.

1: Schmitt trigger is disabled.

The baud rate clock is the peripheral clock divided by 16 times the value (CD) written in the Debug Unit Baud Rate Generator register (DBGU_BRGR). If DBGU_BRGR is set to 0, the baud rate clock is disabled and the Debug Unit's UART remains inactive. The maximum allowable baud rate is peripheral clock divided by 16. The minimum allowable baud rate is peripheral clock divided by (16 x 65536).

$$\text{Baud Rate} = \frac{f_{\text{peripheral clock}}}{16 \times \text{CD}}$$

Figure 23-3. Baud Rate Generator



23.5.2 Receiver

23.5.2.1 Receiver Reset, Enable and Disable

After device reset, the Debug Unit receiver is disabled and must be enabled before being used. The receiver can be enabled by writing one to the RXEN bit in the Debug Unit Control register (DBGU_CR). At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by writing a one to the RXDIS bit in the DBGU_CR. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The programmer can also put the receiver in its reset state by writing a one to the RSTRX bit in the DBGU_CR. In doing so, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost.

23.5.2.2 Start Detection and Data Sampling

The Debug Unit only supports asynchronous operations, and this affects only its receiver. The Debug Unit receiver detects the start of a received character by sampling the DRXD signal until it detects a valid start bit. A low level (space) on DRXD is interpreted as a valid start bit if it is detected for more than 7 cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the DRXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after the falling edge of the start bit was detected.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

Figure 23-4. Start Bit Detection

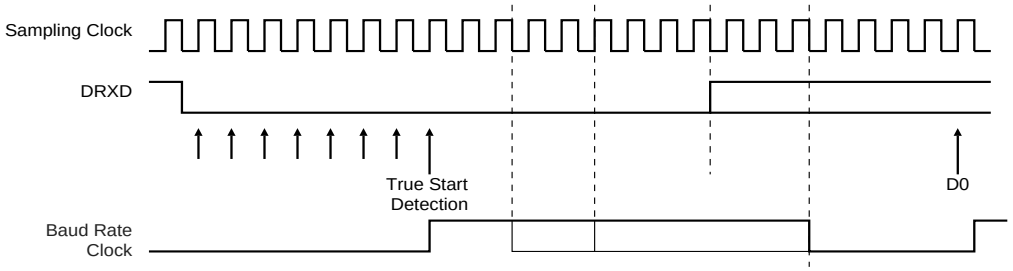
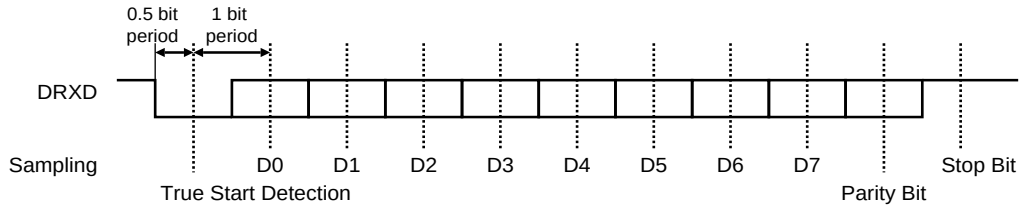


Figure 23-5. Character Reception

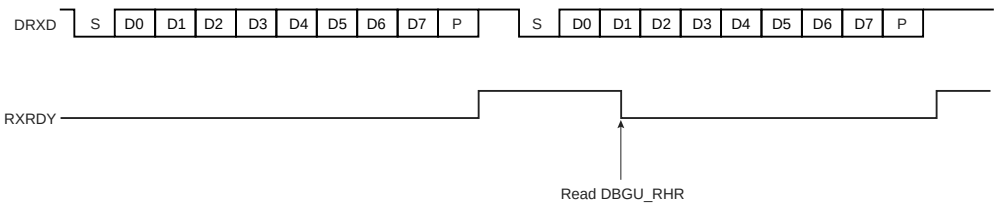
Example: 8-bit, parity enabled 1 stop



23.5.2.3 Receiver Ready

When a complete character is received, it is transferred to the Debug Unit Receive Holding register (DBGU_RHR) and the RXRDY status bit in the Debug Unit Status register (DBGU_SR) is set. The bit RXRDY is automatically cleared when the receive holding register DBGU_RHR is read.

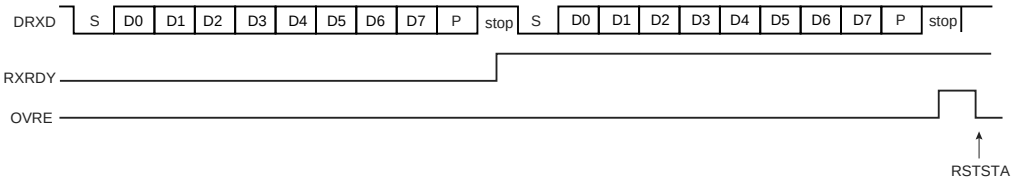
Figure 23-6. Receiver Ready



23.5.2.4 Receiver Overrun

If DBGU_RHR has not been read by the software (or the Peripheral Data Controller or DMA Controller) since the last transfer, the RXRDY bit is still set and a new character is received, the OVRE status bit in DBGU_SR is set. OVRE is cleared when the software writes a one to the bit RSTSTA (Reset Status) in the DBGU_CR.

Figure 23-7. Receiver Overrun



23.5.2.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Debug Unit Mode register (DBGU_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in DBGU_SR is set at the same time as the RXRDY is set. The parity bit is

24.7.2 Bus Matrix Slave Configuration Registers

Name: MATRIX_SCFG0...MATRIX_SCFG9

Address: 0xFFFFDE40 [0], 0xFFFFDE44 [1], 0xFFFFDE48 [2], 0xFFFFDE4C [3], 0xFFFFDE50 [4], 0xFFFFDE54 [5], 0xFFFFDE58 [6], 0xFFFFDE5C [7], 0xFFFFDE60 [8], 0xFFFFDE64 [9]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	FIXED_DEFMSTR				DEFMSTR_TYPE	

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SLOT_CYCLE

7	6	5	4	3	2	1	0
SLOT_CYCLE							

This register can only be written if the WPEN bit is cleared in the “Write Protection Mode Register” .

- **SLOT_CYCLE: Maximum Bus Grant Duration for Masters**

When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place so as to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.

If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access or in the particular case of a master performing back-to-back undefined length bursts indefinitely freezing the arbitration.

This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.

In most cases this feature is not needed and should be disabled for power saving. See Section 24.5.1.2 on page 305.

- **DEFMSTR_TYPE: Default Master Type**

0: No Default Master

At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters.

This results in a one-clock cycle latency for the first access of a burst transfer or for a single access.

1: Last Default Master

At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it.

This results in not having a one-clock cycle latency when the last master tries to access the slave again.

2: Fixed Default Master

At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field.

This results in not having a one-clock cycle latency when the fixed master tries to access the slave again.

- **FIXED_DEFMSTR: Fixed Default Master**

This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Figure 28-3. Memory Connection for an 8-bit Data Bus

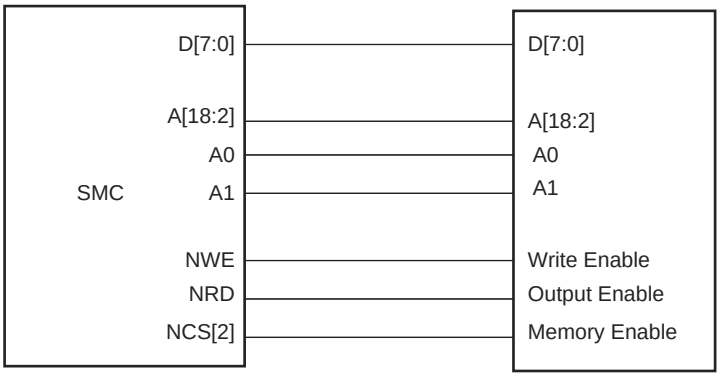


Figure 28-4. Memory Connection for a 16-bit Data Bus

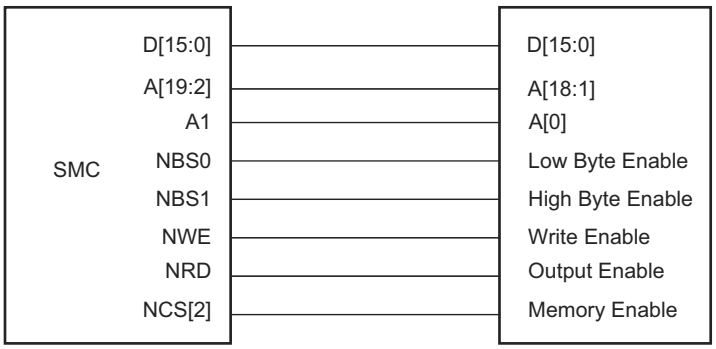
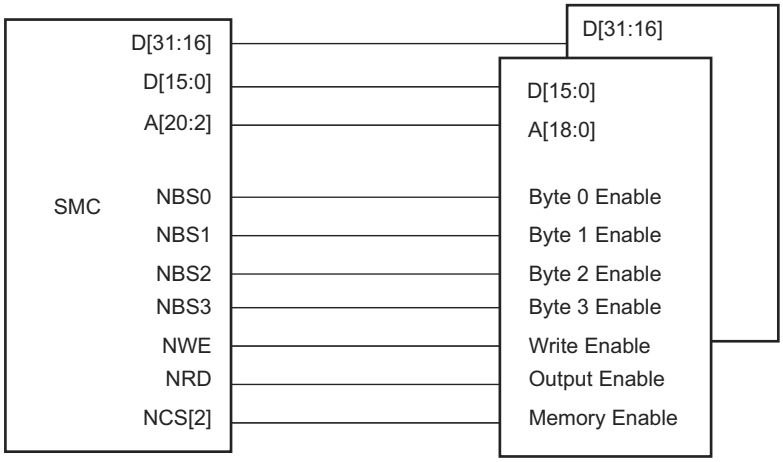


Figure 28-5. Memory Connection for a 32-bit Data Bus



28.8.2.1 Byte Write Access

Byte write access supports one byte write signal per byte of the data bus and a single read signal.

Note that the SMC does not allow boot in Byte Write Access mode.

- For 16-bit devices: the SMC provides NWR0 and NWR1 write signals for respectively byte0 (lower byte) and byte1 (upper byte) of a 16-bit bus. One single read signal (NRD) is provided.

  Byte Write Access is used to connect 2 x 8-bit devices as a 16-bit memory.

- For 32-bit devices: NWR0, NWR1, NWR2 and NWR3, are the write signals of byte0 (lower byte), byte1, byte2 and byte 3 (upper byte) respectively. One single read signal (NRD) is provided.

  Byte Write Access is used to connect 4 x 8-bit devices as a 32-bit memory.

  Byte Write option is illustrated on Figure 28-6.

28.8.2.2 Byte Select Access

In this mode, read/write operations can be enabled/disabled at a byte level. One byte-select line per byte of the data bus is provided. One NRD and one NWE signal control read and write.

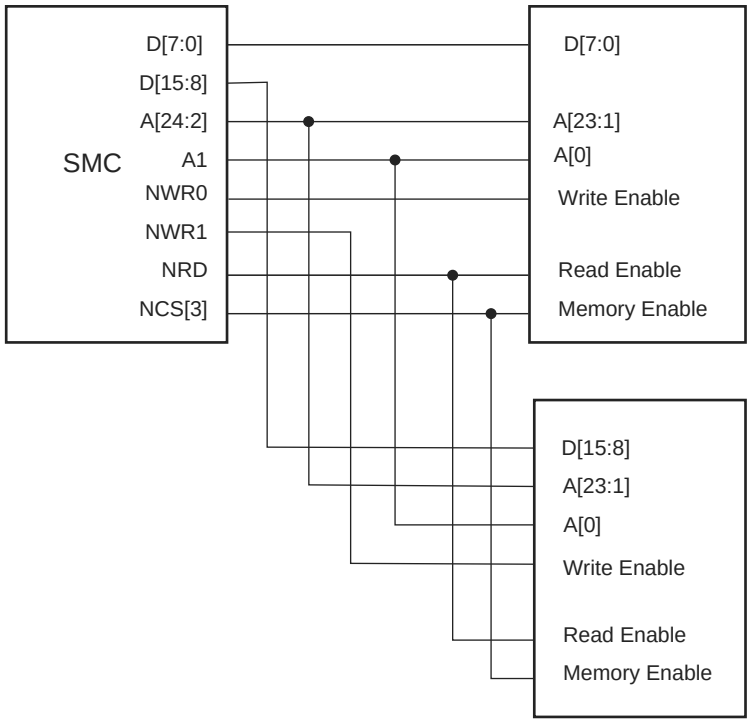
- For 16-bit devices: the SMC provides NBS0 and NBS1 selection signals for respectively byte0 (lower byte) and byte1 (upper byte) of a 16-bit bus.

  Byte Select Access is used to connect one 16-bit device.

- For 32-bit devices: NBS0, NBS1, NBS2 and NBS3, are the selection signals of byte0 (lower byte), byte1, byte2 and byte 3 (upper byte) respectively. Byte Select Access is used to connect two 16-bit devices.

Figure 28-7 shows how to connect two 16-bit devices on a 32-bit data bus in Byte Select Access mode, on NCS3 (BAT = Byte Select Access).

Figure 28-6. Connection of 2 x 8-bit Devices on a 16-bit Bus: Byte Write Option

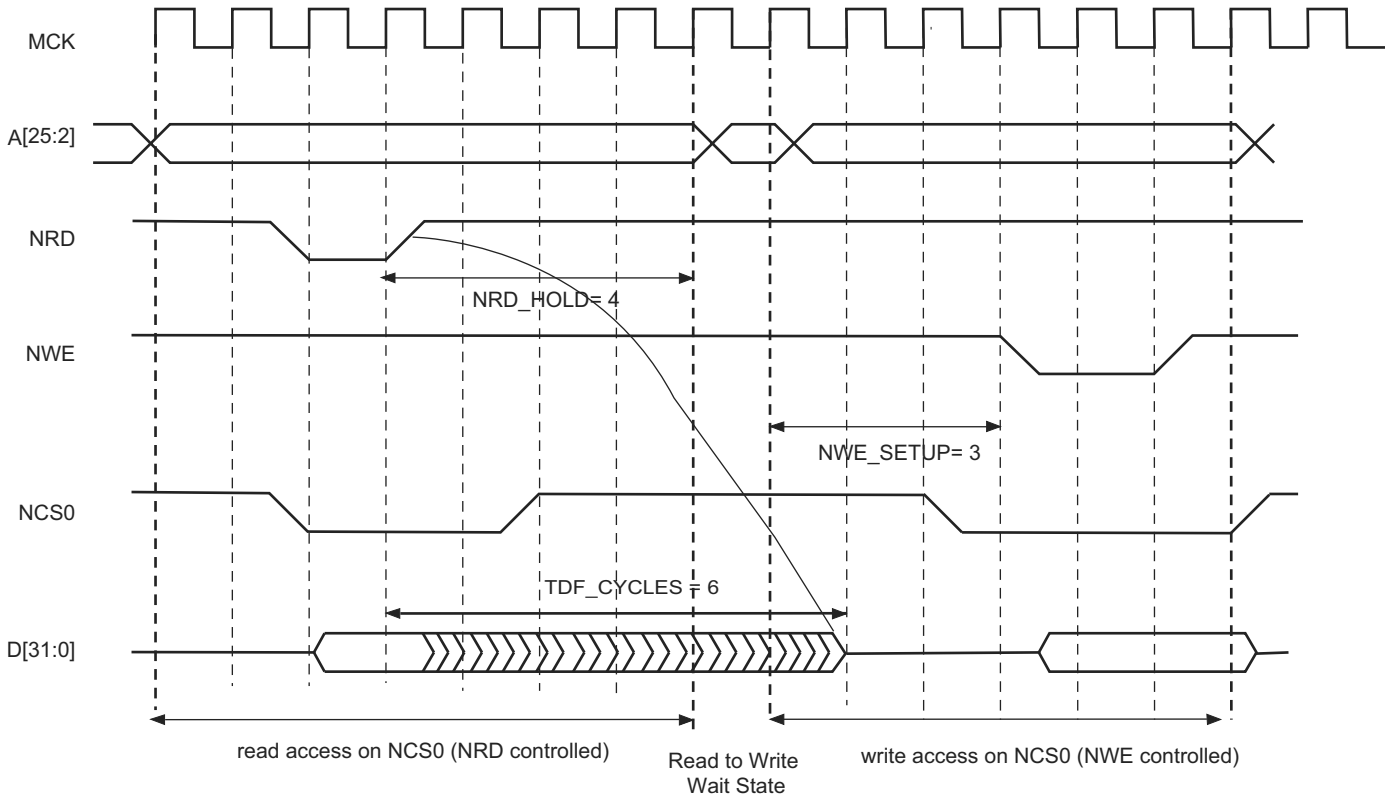


28.8.2.3 Signal Multiplexing

Depending on the byte access type (BAT), only the write signals or the byte select signals are used. To save IOs at the external bus interface, control signals at the SMC interface are multiplexed. Table 28-4 shows signal multiplexing depending on the data bus width and the byte access type.

For 32-bit devices, bits A0 and A1 are unused. For 16-bit devices, bit A0 of address is unused. When the Byte Select option is selected, NWR1 to NWR3 are unused. When teh Byte Write option is selected, NBS0 to NBS3 are unused.

Figure 28-22. TDF Optimization: No TDF wait states are inserted if the TDF period is over when the next access begins



28.11.3 TDF Optimization Disabled (TDF_MODE = 0)

When optimization is disabled, tdf wait states are inserted at the end of the read transfer, so that the data float period is ended when the second access begins. If the hold period of the read1 controlling signal overlaps the data float period, no additional tdf wait states will be inserted.

Figure 28-23, Figure 28-24 and Figure 28-25 illustrate the cases:

- read access followed by a read access on another chip select,
- read access followed by a write access on another chip select,
- read access followed by a write access on the same chip select,

with no TDF optimization.

29. DDR SDR SDRAM Controller (DDRSDRC)

29.1 Description

The DDR SDR SDRAM Controller (DDRSDRC) is a multiport memory controller. It comprises four slave AHB interfaces. All simultaneous accesses (four independent AHB ports) are interleaved to maximize memory bandwidth and minimize transaction latency due to SDRAM protocol.

The DDRSDRC extends the memory capabilities of a chip by providing the interface to an external 16-bit or 32-bit SDR-SDRAM device and external 16-bit DDR-SDRAM device. The page size supports ranges from 2048 to 16384 rows and from 256 to 4096 columns. It supports byte (8-bit), half-word (16-bit) and word (32-bit) accesses.

The DDRSDRC supports a read or write burst length (BL) of eight locations. This enables the command and address bus to anticipate the next command, thus reducing latency imposed by the SDRAM protocol and improving the SDRAM bandwidth. Moreover, the DDRSDRC keeps track of the active row in each bank, thus maximizing SDRAM performance, e.g., the application may be placed in one bank and data in the other banks. To optimize performance, it is advisable to avoid accessing different rows in the same bank. The DDRSDRC supports a CAS latency of 2 or 3 and optimizes the read access depending on the frequency.

The features of Self refresh, Power-down, and Deep Power-down modes minimize the consumption of the SDRAM device.

The DDRSDRC user interface is compliant with ARM Advanced Peripheral Bus (APB rev2).

Note: The term “SDRAM device” regroups SDR-SDRAM, Low-power SDR-SDRAM, Low-power DDR1-SDRAM and DDR2-SDRAM devices.

30.8.16 DMAC Channel x [x = 0..7] Control A Register

Name: DMAC_CTRLAx [x = 0..7]

Address: 0xFFFFEC48 (0)[0], 0xFFFFEC70 (0)[1], 0xFFFFEC98 (0)[2], 0xFFFFECC0 (0)[3], 0xFFFFECE8 (0)[4], 0xFFFFED10 (0)[5], 0xFFFFED38 (0)[6], 0xFFFFED60 (0)[7], 0xFFFFEE48 (1)[0], 0xFFFFEE70 (1)[1], 0xFFFFEE98 (1)[2], 0xFFFFEEC0 (1)[3], 0xFFFFEEE8 (1)[4], 0xFFFFEF10 (1)[5], 0xFFFFEF38 (1)[6], 0xFFFFEF60 (1)[7]

Access: Read/Write

31	30	29	28	27	26	25	24
DONE	–	DST_WIDTH		–	–	SRC_WIDTH	

23	22	21	20	19	18	17	16
–	DCSIZE			–	SCSIZE		

15	14	13	12	11	10	9	8
BTSIZE							

7	6	5	4	3	2	1	0
BTSIZE							

This register can only be written if the WPEN bit is cleared in “DMAC Write Protection Mode Register” on page 526.

- **BTSIZE: Buffer Transfer Size**

The transfer size relates to the number of transfers to be performed, that is, for writes it refers to the number of source width transfers to perform when DMAC is flow controller. For reads, BTSIZE refers to the number of transfers completed on the Source Interface. When this field is cleared, the DMAC module is automatically disabled when the relevant channel is enabled.

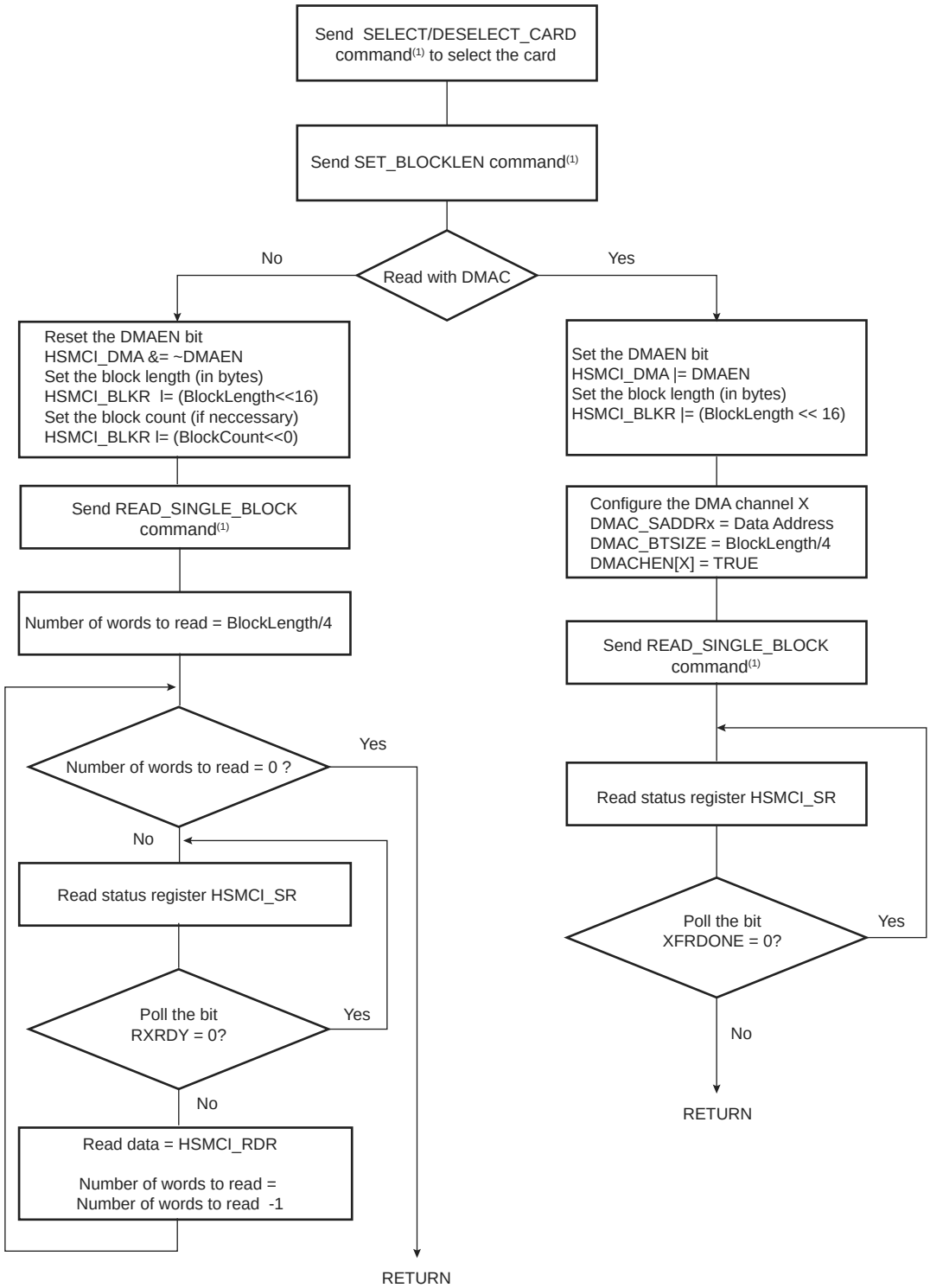
- **SCSIZE: Source Chunk Transfer Size**

Value	Name	Description
000	CHK_1	1 data transferred
001	CHK_4	4 data transferred
010	CHK_8	8 data transferred
011	CHK_16	16 data transferred

- **DCSIZE: Destination Chunk Transfer Size**

Value	Name	Description
000	CHK_1	1 data transferred
001	CHK_4	4 data transferred
010	CHK_8	8 data transferred
011	CHK_16	16 data transferred

Figure 33-8. Read Functional Flow Diagram



Notes: 1. It is assumed that this command has been correctly sent (see Figure 33-7).

33.8.4 Write Operation

In write operation, the HSMCI Mode Register (HSMCI_MR) is used to define the padding value when writing non-multiple block size. If the bit PADV is 0, then 0x00 value is used when padding data, otherwise 0xFF is used.

37.8.5 TWI Clock Waveform Generator Register

Name: TWI_CWGR

Address: 0xF8010010 (0), 0xF8014010 (1), 0xF8018010 (2)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	CKDIV		

15	14	13	12	11	10	9	8
CHDIV							

7	6	5	4	3	2	1	0
CLDIV							

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

TWI_CWGR is only used in Master mode.

- **CLDIV: Clock Low Divider**

The TWCK low period is defined as follows: $t_{low} = ((CLDIV \times 2^{CKDIV}) + 4 \times t_{peripheral\ clock}$

- **CHDIV: Clock High Divider**

The TWCK high period is defined as follows: $t_{high} = ((CHDIV \times 2^{CKDIV}) + 4 \times t_{peripheral\ clock}$

- **CKDIV: Clock Divider**

The CKDIV field is used to increase both TWCK high and low periods.

43.6.3.5 Window Attributes Software Operation

1. When required, write the overlay attributes configuration registers.
2. Set UPDATEEN field of the CHXCHER register.
3. Poll UPDATESR field in the CHXCHSR, the update applies when that field is reset.

43.6.4 RGB Frame Buffer Memory Bitmap

43.6.4.1 1 bpp Through Color Lookup Table

Table 43-10. 1 bpp memory mapping, little endian organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 1 bpp	p31	p30	p29	p28	p27	p26	p25	p24	p23	p22	p21	p20	p19	p18	p17	p16	p15	p14	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0

43.6.4.2 2 bpp Through Color Lookup Table

Table 43-11. 2 bpp memory mapping, little endian organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 2 bpp	p15		p14		p13		p12		p11		p10		p9		p8		p7		p6		p5		p4		p3		p2		p1		p0	

43.6.4.3 4 bpp Through Color Lookup Table

Table 43-12. 4 bpp memory mapping, little endian organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 4 bpp	p7				p6				p5				p4				p3				p2				p1				p0			

43.6.4.4 8 bpp Through Color Lookup Table

Table 43-13. 8 bpp memory mapping, little endian organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 8 bpp	p3								p2								p1								p0							

43.6.4.5 12 bpp Memory Mapping, RGB 4:4:4

Table 43-14. 12 bpp memory mapping, little endian organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	–				R1[3:0]				G1[3:0]				B1[3:0]				–				R0[3:0]				G0[3:0]				B0[3:0]			

43.6.4.6 16 bpp Memory Mapping with Alpha Channel, ARGB 4:4:4:4

Table 43-15. 16 bpp memory mapping, little endian organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	A1[3:0]				R1[3:0]				G1[3:0]				B1[3:0]				A0[3:0]				R0[3:0]				G0[3:0]				B0[3:0]			

43.7.94 Hardware Cursor Layer Interrupt Status Register

Name: LCDC_HCRISR

Address: 0xF8038358

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

When set to one this flag indicates that an End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

When set to one this flag indicates that a descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

When set to one this flag indicates that the descriptor pointed to by the head register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

When set to one this flag indicates that an End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

When set to one this flag indicates that an Overflow has occurred. This flag is reset after a read operation.

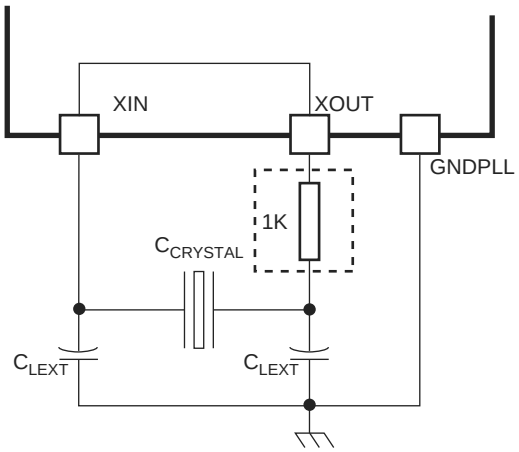
44.5 Main Oscillator Characteristics

Table 44-7. Main Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$1/(t_{CPMAIN})$	Crystal Oscillator Frequency		12		16	MHz
$C_{CRYSTAL}$ (1)	Crystal Load Capacitance		15		20	pF
C_{LEXT}	External Load Capacitance	$C_{CRYSTAL}$ = 15 pF (1)		27		pF
		$C_{CRYSTAL}$ = 20 pF (1)		32		pF
	Duty Cycle		40		60	%
t_{START}	Startup Time				2	ms
I_{DDST}	Standby Current Consumption	Standby mode			1	µA
P_{ON}	Drive Level				150	µW
$I_{DD\ ON}$	Current Dissipation	@ 12 MHz		0.52	0.55	mA
		@ 16 MHz		0.7	1.1	mA

Note: 1. The $C_{CRYSTAL}$ value is specified by the crystal manufacturer. In our case, $C_{CRYSTAL}$ must be between 15 pF and 20 pF. All parasitic capacitance, package and board, **must be calculated** in order to reach 15 pF (minimum targeted load for the oscillator) by taking into account the internal load C_{INT}. So, to target the minimum oscillator load of 15 pF, external capacitance must be 15 pF - 4 pF = 11 pF which means that 22 pF is the target value (22 pF from XIN to GND and 22 pF from XOUT to GND). If 20 pF load is targeted, the sum of pad, package, board and external capacitances must be 20 pF - 4 pF = 16 pF which means 32 pF (32 pF from XIN to GND and 32 pF from XOUT to GND).

Figure 44-2. Main Oscillator Schematics



Note: A 1K resistor must be added on XOUT pin for crystals with frequencies lower than 8 MHz.

Doc. Rev. 11052G	Comments
31-Aug-15	Section 14. “Real-time Clock (RTC)” (cont’d) Section 14.6.5 “RTC Time Alarm Register”: added sentence on register write protection; added recommendation for changing SEC, MIN, and HOUR fields Section 14.6.6 “RTC Calendar Alarm Register”: added sentence on register write protection; added recommendation for changing DATE and MONTH fields
	Section 15. “Periodic Interval Timer (PIT)” Section 15.2 “Embedded Characteristics”: removed “Real Time OS or Linux/WinCE compliant tick generator” and “AMBA-compliant Interface”
	Section 16. “Watchdog Timer (WDT)” Section 16.1 “Description”: updated slow clock frequency Section 16.2 “Embedded Characteristics”: added bullet “Watchdog Clock is independent from Processor Clock” Section 16.4 “Functional Description”: below third paragraph, added sentence “When the WDDIS bit is set, the fields WDV and WDD must not be modified.” Figure 16-2 “Watchdog Behavior”: “WDT_CR = WDRSTT” corrected to “WDT_CR.WDRSTT=1” Section 16.5.1 “Watchdog Timer Control Register”: added note below bitmap; updated descriptions of WDRSTT bit and KEY field Section 16.5.2 “Watchdog Timer Mode Register”: added two notes below bitmap; added note to WDDIS bit description Section 16.5.3 “Watchdog Timer Status Register”: updated WDUNF and WDERR bit descriptions
	Section 17. “Shutdown Controller (SHDWC)” Section 17.6 “Functional Description”: inserted heading Section 17.6.1 “Wake-up Inputs” Section 17.7.1 “Shutdown Control Register”: updated KEY field description Section 17.7.2 “Shutdown Mode Register”: updated description of fields WKMODE0 and CPTWK0
	Section 18. “General Purpose Backup Registers (GPBR)” Updated Section 18.1 “Description” Updated Section 18.2 “Embedded Characteristics” Table 18-1 “Register Mapping”: added reset value 0x00000000 for all registers SYS_GPBRx Section 18.3.1 “General Purpose Backup Register x”: inserted sentence “These registers are reset at first power-up and on each loss of VDDBU”
	Section 19. “Slow Clock Controller (SCKC)” Updated Section 19.1 “Description” Updated Figure 19-1 “Block Diagram” Inserted heading Section 19.4 “Functional Description” and updated content Section 19.5.1 “Slow Clock Controller Configuration Register”: updated bit descriptions; removed reset value (redundant with reset value in Table 19-1 “Register Mapping”)
	Section 20. “Clock Generator” Section 20.2 “Embedded Characteristics”: updated description of low-power RC oscillator Replaced section “Slow Clock Selection” with new Section 20.4 “Slow Clock” Revised Section 20.5 “Main Clock” Removed section “Main Clock Selection” (refer to Section 20.5.3 “Main Clock Source Selection”) Updated Section 20.7 “UTMI Phase Lock Loop Programming”