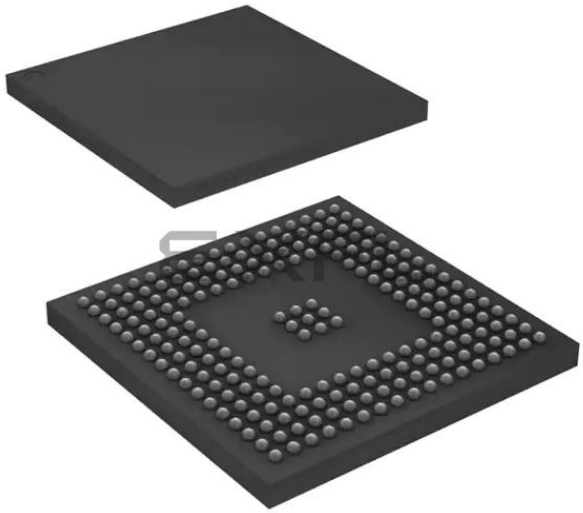


Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

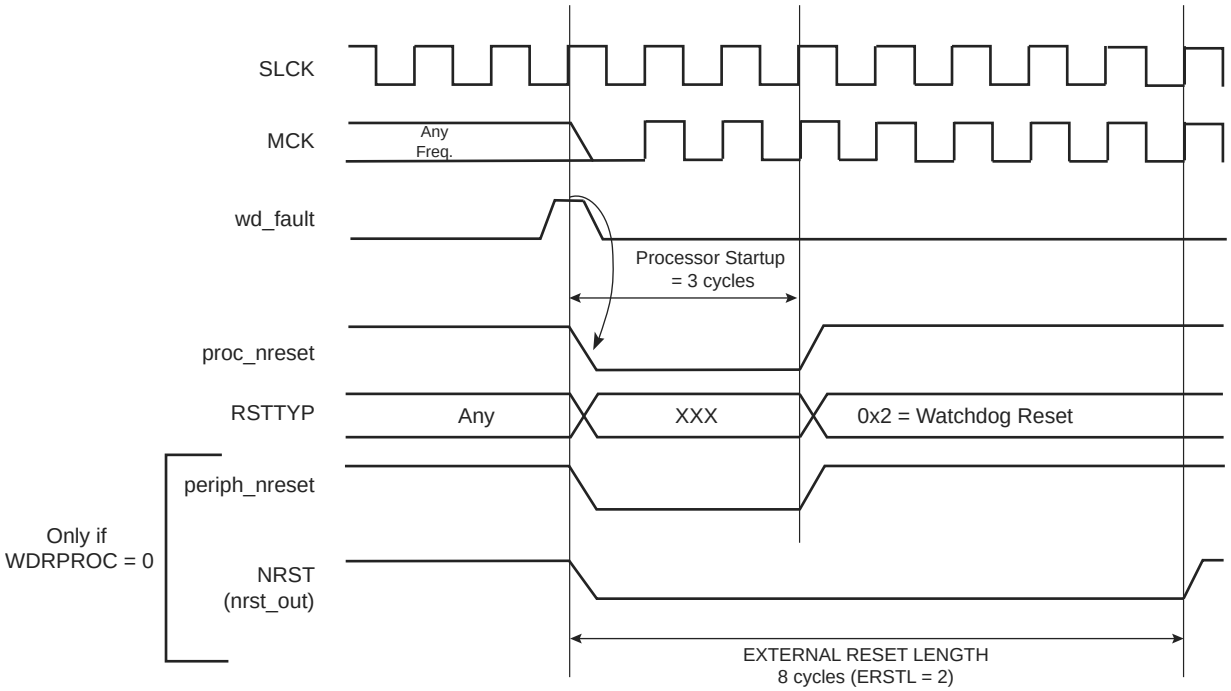
Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, 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	Obsolete
Core Processor	ARM926EJ-S
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	400MHz
Co-Processors/DSP	-
RAM Controllers	LPDDR, LPDDR2, DDR2, SDR, SRAM
Graphics Acceleration	No
Display & Interface Controllers	LCD, Touchscreen
Ethernet	-
SATA	-
USB	USB 2.0 (2)
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/at91sam9cn12-cu

Figure 13-8: Watchdog Reset



13.4.5 Reset State Priorities

The Reset State Manager manages the following priorities between the different reset sources, given in descending order:

- Backup Reset
- Wake-up Reset
- User Reset
- Watchdog Reset
- Software Reset

Particular cases are listed below:

- When in User Reset:
  - A watchdog event is impossible because the Watchdog Timer is being reset by the proc_nreset signal.
  - A software reset is impossible, since the processor reset is being activated.
- When in Software Reset:
  - A watchdog event has priority over the current state.
  - The NRST has no effect.
- When in Watchdog Reset:
  - The processor reset is active and so a Software Reset cannot be programmed.
  - A User Reset cannot be entered.

23.3 Block Diagram

Figure 23-1: Debug Unit Functional Block Diagram

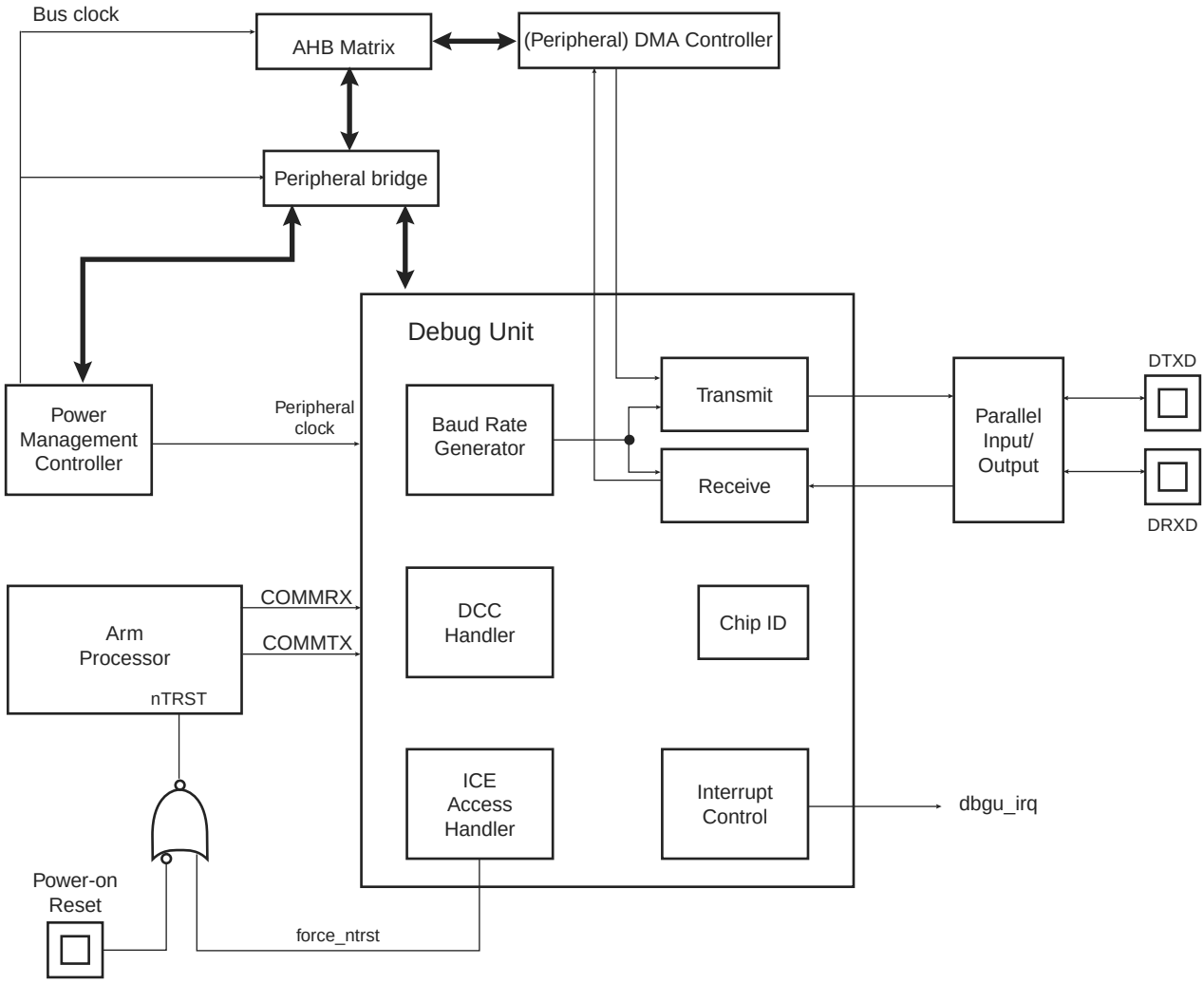


Table 23-1: Debug Unit Pin Description

Pin Name	Description	Type
DRXD	Debug Receive Data	Input
DTXD	Debug Transmit Data	Output

25. Bus Matrix (MATRIX)

25.1 Description

The Bus Matrix implements a multi-layer AHB, based on the AHB-Lite protocol, that enables parallel access paths between multiple AHB masters and slaves in a system, thus increasing the overall bandwidth. The Bus Matrix interconnects up to 16 AHB masters to up to 16 AHB slaves. The normal latency to connect a master to a slave is one cycle except for the default master of the accessed slave which is connected directly (zero cycle latency).

The Bus Matrix user interface is compliant with Arm Advanced Peripheral Bus and provides a Chip Configuration User Interface with Registers that allow the Bus Matrix to support application specific features.

25.2 Embedded Characteristics

- 6-layer Matrix, handling requests from 6 masters
- Programmable Arbitration strategy
  - Fixed-priority Arbitration
  - Round-Robin Arbitration, either with no default master, last accessed default master or fixed default master
- Burst Management
  - Breaking with Slot Cycle Limit Support
  - Undefined Burst Length Support
- One Address Decoder provided per Master
  - Three different slaves may be assigned to each decoded memory area: one for internal ROM boot, one after remap
- Boot Mode Select
  - Non-volatile Boot Memory can be internal ROM or external memory on EBI_NCS0
- Remap Command
  - Allows Remapping of an Internal SRAM in Place of the Boot Non-Volatile Memory (ROM or External Flash)
  - Allows Handling of Dynamic Exception Vectors

25.3 Matrix Masters

The Bus Matrix of the SAM9N12/SAM9CN11/SAM9CN12 product manages six masters, which means that each master can perform an access concurrently with others, to an available slave.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

Table 25-1: List of Bus Matrix Masters

Master 0	Arm926 Instruction
Master 1	Arm926 Data
Master 2&3	DMA Controller
Master 4	USB Host DMA
Master 5	LCD DMA

25.4 Matrix Slaves

The Bus Matrix of the SAM9N12/SAM9CN11/SAM9CN12 product manages five slaves. Each slave has its own arbiter, allowing a different arbitration per slave.

26.5 Application Example

26.5.1 Hardware Interface

Table 26-3 details the connections to be applied between the EBI pins and the external devices for each memory controller.

Table 26-3: EBI Pins and External Static Device Connections

	Pins of the SMC Interfaced Device					
Signals: EBI_	**8-bit Static Device**	**2 x 8-bit Static Devices**	**16-bit Static Device**	**4 x 8-bit Static Devices**	**2 x 16-bit Static Devices**	**32-bit Static Device**
D0–D7	D0–D7	D0–D7	D0–D7	D0–D7	D0–D7	D0–D7
D8–D15	–	D8–D15	D8–D15	D8–D15	D8–15	D8–15
D16–D24	–	–	–	D16–D23	D16–D23	D16–D23
D25–D31(5))	–	–	–	D24–D31	D24–D31	D24–D31
A0/NBS0	A0	–	NLB	–	NLB(3)	BE0
A1/NWR2/NBS2/DQM2	A1	A0	A0	WE(2)	NLB(4)	BE2
A2–A22(5)	A[2:22]	A[1:21]	A[1:21]	A[0:20]	A[0:20]	A[0:20]
A23–A25(5)	A[23:25]	A[22:24]	A[22:24]	A[21:23]	A[21:23]	A[21:23]
NCS0	CS	CS	CS	CS	CS	CS
NCS1/DDRSDCS	CS	CS	CS	CS	CS	CS
NCS2(5)	CS	CS	CS	CS	CS	CS
NCS3/NANDCS	CS	CS	CS	CS	CS	CS
NCS4(5)	CS	CS	CS	CS	CS	CS
NCS5(5)	CS	CS	CS	CS	CS	CS
NRD	OE	OE	OE	OE	OE	OE
NWR0/NWE	WE	WE(1)	WE	WE(2)	WE	WE
NWR1/NBS1	–	WE(1)	NUB	WE(2)	NUB(3)	BE1
NWR3/NBS3/DQM3	–	–	–	WE(2)	NUB(4)	BE3

Note 1: NWR1 enables upper byte writes. NWR0 enables lower byte writes.

2: NWRx enables corresponding byte x writes. (x = 0, 1, 2 or 3).

3: NBS0 and NBS1 enable respectively lower and upper bytes of the lower 16-bit word.

4: NBS2 and NBS3 enable respectively lower and upper bytes of the upper 16-bit word.

5: D25–D31 and A20, A23-A25, NCS2, NCS4, NCS5 are multiplexed on PD15–PD31.

32.7 USB Device Port (UDP) User Interface

WARNING: The UDP peripheral clock in the PMC must be enabled before any read/write operations to the UDP registers, including the UDP_TXVC register.

Table 32-6: Register Mapping

Offset	Register	Name	Access	Reset
0x000	Frame Number Register	UDP_FRM_NUM	Read-only	0x0000_0000
0x004	Global State Register	UDP_GLB_STAT	Read/Write	0x0000_0010
0x008	Function Address Register	UDP_FADDR	Read/Write	0x0000_0100
0x00C	Reserved	–	–	–
0x010	Interrupt Enable Register	UDP_IER	Write-only	
0x014	Interrupt Disable Register	UDP_IDR	Write-only	
0x018	Interrupt Mask Register	UDP_IMR	Read-only	0x0000_1200
0x01C	Interrupt Status Register	UDP_ISR	Read-only	–(1)
0x020	Interrupt Clear Register	UDP_ICR	Write-only	
0x024	Reserved	–	–	–
0x028	Reset Endpoint Register	UDP_RST_EP	Read/Write	0x0000_0000
0x02C	Reserved	–	–	–
0x030	Endpoint Control and Status Register 0	UDP_CSR0	Read/Write	0x0000_0000
...	...	...	...	...
0x030 + 0x4 * 5	Endpoint Control and Status Register 5	UDP_CSR5	Read/Write	0x0000_0000
0x050	Endpoint FIFO Data Register 0	UDP_FDR0	Read/Write	–(1)
...	...	...	...	...
0x050 + 0x4 * 5	Endpoint FIFO Data Register 5	UDP_FDR5	Read/Write	–(1)
0x070	Reserved	–	–	–
0x074	Transceiver Control Register	UDP_TXVC(2)	Read/Write	0x0000_0100
0x078–0xFC	Reserved	–	–	–

Note 1: Reset values are not defined for UDP_ISR or UDP_FDRx. UDP_FDRs reflect Dual Port RAM memory locations which are not affected by any reset signals.

2: See Warning above Table 32-6.

32.7.9 UDP Reset Endpoint Register

Name:UDP_RST_EP

Address:0xF803C028

Access: Read/Write

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
		EP5	EP4	EP3	EP2	EP1	EP0

EP0: Reset Endpoint 0

EP1: Reset Endpoint 1

EP2: Reset Endpoint 2

EP3: Reset Endpoint 3

EP4: Reset Endpoint 4

EP5: Reset Endpoint 5

This flag is used to reset the FIFO associated with the endpoint and the bit RXBYTECOUNT in the UDP_CSRx. It also resets the data toggle to DATA0. It is useful after removing a HALT condition on a BULK endpoint. Refer to Chapter 5.8.5 in the *USB Serial Bus Specification, Rev.2.0*.

Warning: This flag must be cleared at the end of the reset. It does not clear UDP_CSRx flags.

0: No reset

1: Forces the corresponding endpoint FIF0 pointers to 0, therefore RXBYTECNT field is read at 0 in UDP_CSRx

Resetting the endpoint is a two-step operation:

1. Set the corresponding EPx field.
2. Clear the corresponding EPx field.

h) Configure the fields of LLI_W(n).DMAC_CTRLBx as follows:
   – DST_INCR is set to INCR.
   – SRC_INCR is set to INCR.
   – FC field is programmed with peripheral to memory flow control mode.
   – SRC_DSCR is configured to 0 (descriptor fetch is enabled for the SRC).
   – DST_DSCR is set to TRUE (descriptor fetch is disabled for the DST).
   – DIF and SIF are set with their respective layer ID. If SIF is different from DIF, the DMA Controller is able to prefetch data and write HSMCI simultaneously.

i) Configure the fields of LLI_W(n).DMAC_CFGx for Channel x as follows:
   – FIFOCFG defines the watermark of the DMA channel FIFO.
   – DST_REP is set to zero. Address are contiguous.
   – SRC_H2SEL is set to true to enable hardware handshaking on the destination.
   – SRC_PER is programmed with the hardware handshaking ID of the targeted HSMCI Host Controller.

j) Program LLI_W(n).DMAC_DSCRx with the address of LLI_B(n) descriptor. And set the DSCRx_IF to the AHB Layer ID. This operation actually links the Word oriented descriptor on the second byte oriented descriptor. When *block_length[1:0]* is equal to 0 (multiple of 4) LLI_W(n).DMAC_DSCRx points to 0, only LLI_W(n) is relevant.

k) Program the channel registers in the Memory for the second descriptor. This descriptor will be byte oriented. This descriptor is referred to as LLI_B(n), standing for LLI Byte oriented.

l) The LLI_B(n).DMAC_SADDRx field in memory must be set with the starting address of the HSMCI_FIFO address.

m) The LLI_B(n).DMAC_DADDRx is not relevant if previous word aligned descriptor was enabled. If 1, 2 or 3 bytes are transferred, that address is user defined and not word aligned.

n) Configure the fields of LLI_B(n).DMAC_CTRLAx as follows:
   – DST_WIDTH is set to BYTE.
   – SRC_WIDTH is set to BYTE.
   – SCSIZE must be set according to the value of HSMCI_DMA.CHKSIZE.
   – BTSIZE is programmed with *block_length[1:0].* (last 1, 2, or 3 bytes of the buffer).

o) Configure the fields of LLI_B(n).DMAC_CTRLBx as follows:
   – DST_INCR is set to INCR.
   – SRC_INCR is set to INCR.
   – FC field is programmed with peripheral to memory flow control mode.
   – Both SRC_DSCR and DST_DSCR are set (descriptor fetch is disabled) or Next descriptor location points to 0.
   – DIF and SIF are set with their respective layer ID. If SIF is different from DIF, the DMA Controller is able to prefetch data and write HSMCI simultaneously.

p) Configure the LLI_B(n).DMAC_CFGx memory location for Channel x as follows:
   – FIFOCFG defines the watermark of the DMAC channel FIFO.
   – SRC_H2SEL is set to true to enable hardware handshaking on the destination.
   – SRC_PER is programmed with the hardware handshaking ID of the targeted HSMCI Host Controller

q) Program LLI_B(n).DMAC_DSCR with address of descriptor LLI_W(n+1). If LLI_B(n) is the last descriptor, then program LLI_B(n).DMAC_DSCR with 0.

r) Program the DMAC_CTRLBx register for Channel x with 0. Its content is updated with the LLI Fetch operation.

s) Program DMAC_DSCRx with the address of LLI_W(0) if *block_length* is greater than 4 else with address of LLI_B(0).

t) Enable Channel x writing one to DMAC_CHER[x]. The DMAC is ready and waiting for request.

4. Enable DMADONE interrupt in the HSMCI_IER.
5. Poll CBTC[x] bit in the DMAC_EBCISR.
6. If a new list of buffers shall be transferred, repeat step 7. Check and handle HSMCI errors.

PCS: Peripheral Chip Select

This field is only used if fixed peripheral select is active (PS = 0).

If SPI_MR.PCSDEC = 0:

PCS = xxx0NPCS[3:0] = 1110

PCS = xx01NPCS[3:0] = 1101

PCS = x011NPCS[3:0] = 1011

PCS = 0111NPCS[3:0] = 0111

PCS = 1111forbidden (no peripheral is selected)

(x = don't care)

If SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

DLYBCS: Delay Between Chip Selects

This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees non-overlapping chip selects and solves bus contentions in case of peripherals having long data float times.

If DLYBCS is lower than 6, six peripheral clock periods are inserted by default.

Otherwise, the following equation determines the delay:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{f_{\text{peripheral clock}}}$$

36.7.2 TC Channel Mode Register: Capture Mode

Name:TC_CMRx [x=0..2] (CAPTURE_MODE)

Address:0xF8008004 (0)[0], 0xF8008044 (0)[1], 0xF8008084 (0)[2], 0xF800C004 (1)[0], 0xF800C044 (1)[1], 0xF800C084 (1)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	LDRB		LDRA	
15	14	13	12	11	10	9	8
WAVE	CPCTRG	–	–	–	ABETRG	ETRGEDG	
7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

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

TCCLKS: Clock Selection

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal MCK/2 clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal MCK/8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal MCK/32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal MCK/128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal SLCK clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

CLKI: Clock Invert

0: Counter is incremented on rising edge of the clock.

1: Counter is incremented on falling edge of the clock.

BURST: Burst Signal Selection

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	XC0	XC0 is ANDed with the selected clock.
2	XC1	XC1 is ANDed with the selected clock.
3	XC2	XC2 is ANDed with the selected clock.

LDBSTOP: Counter Clock Stopped with RB Loading

0: Counter clock is not stopped when RB loading occurs.

1: Counter clock is stopped when RB loading occurs.

37.7.4 PWM Status Register

Name:PWM_SR

Address:0xF803400C

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
–	–	–	–	CHID3	CHID2	CHID1	CHID0

CHIDx: Channel ID

0: PWM output for channel x is disabled.

1: PWM output for channel x is enabled.

Master Node Configuration

The user can choose between two DMAC modes by the PDCM bit in the US_LINMR:

- PDCM = 1: the LIN configuration is stored in the WRITE buffer and it is written by the DMAC in the Transmit Holding register US_THR (instead of the LIN Mode register US_LINMR). Because the DMAC transfer size is limited to a byte, the transfer is split into two accesses. During the first access the bits, NACT, PARDIS, CHKDIS, CHKTYP, DLM and FSDIS are written. During the second access the 8-bit DLC field is written.
- PDCM = 0: the LIN configuration is not stored in the WRITE buffer and it must be written by the user in US_LINMR.

The WRITE buffer also contains the Identifier and the DATA, if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the DATA if the USART receives the response (NACT = SUBSCRIBE).

Figure 39-51: Master Node with DMAC (PDCM = 1)

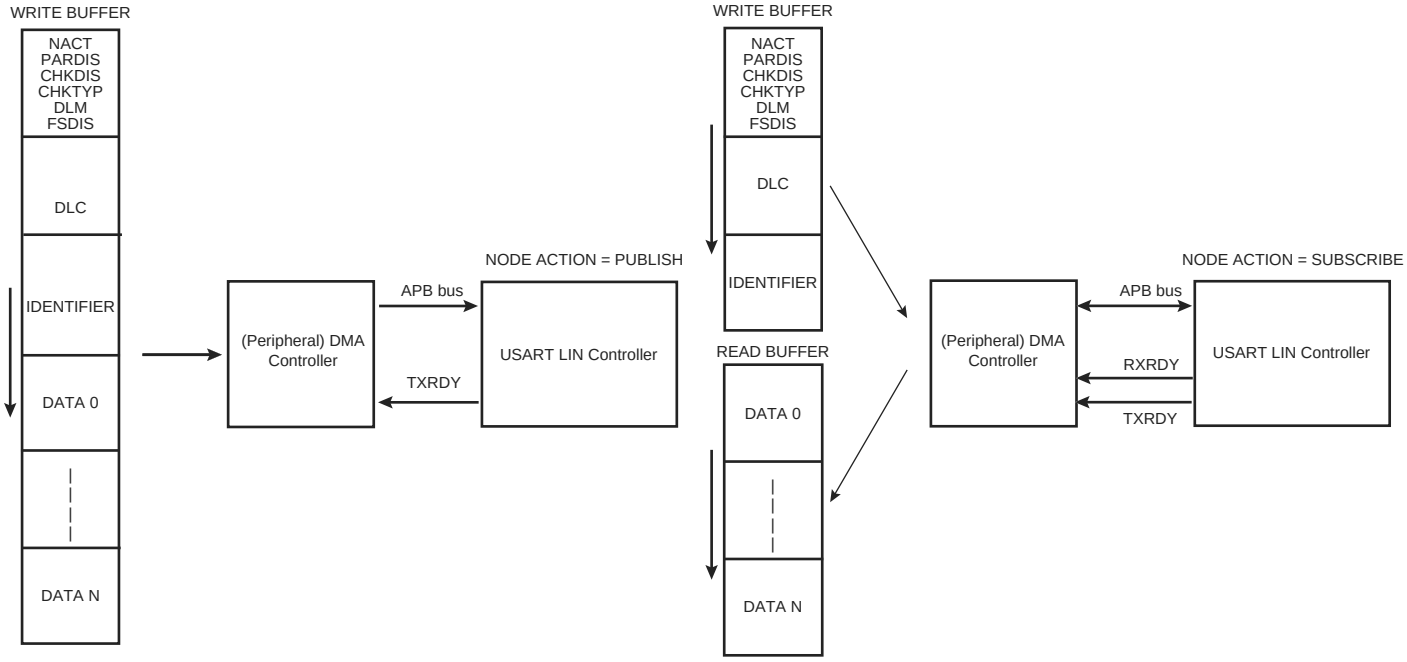
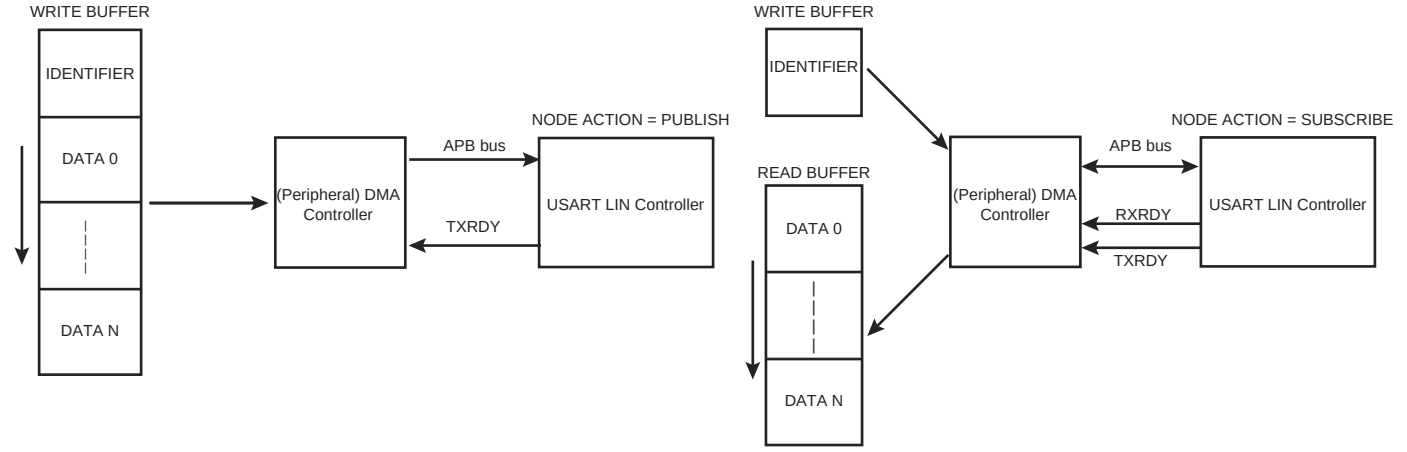


Figure 39-52: Master Node with DMAC (PDCM = 0)



39.7.9 USART Interrupt Disable Register (SPI_MODE)

Name:US_IDR (SPI_MODE)

Address:0xF801C00C (0), 0xF802000C (1), 0xF802400C (2), 0xF802800C (3)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

RXRDY: RXRDY Interrupt Disable

TXRDY: TXRDY Interrupt Disable

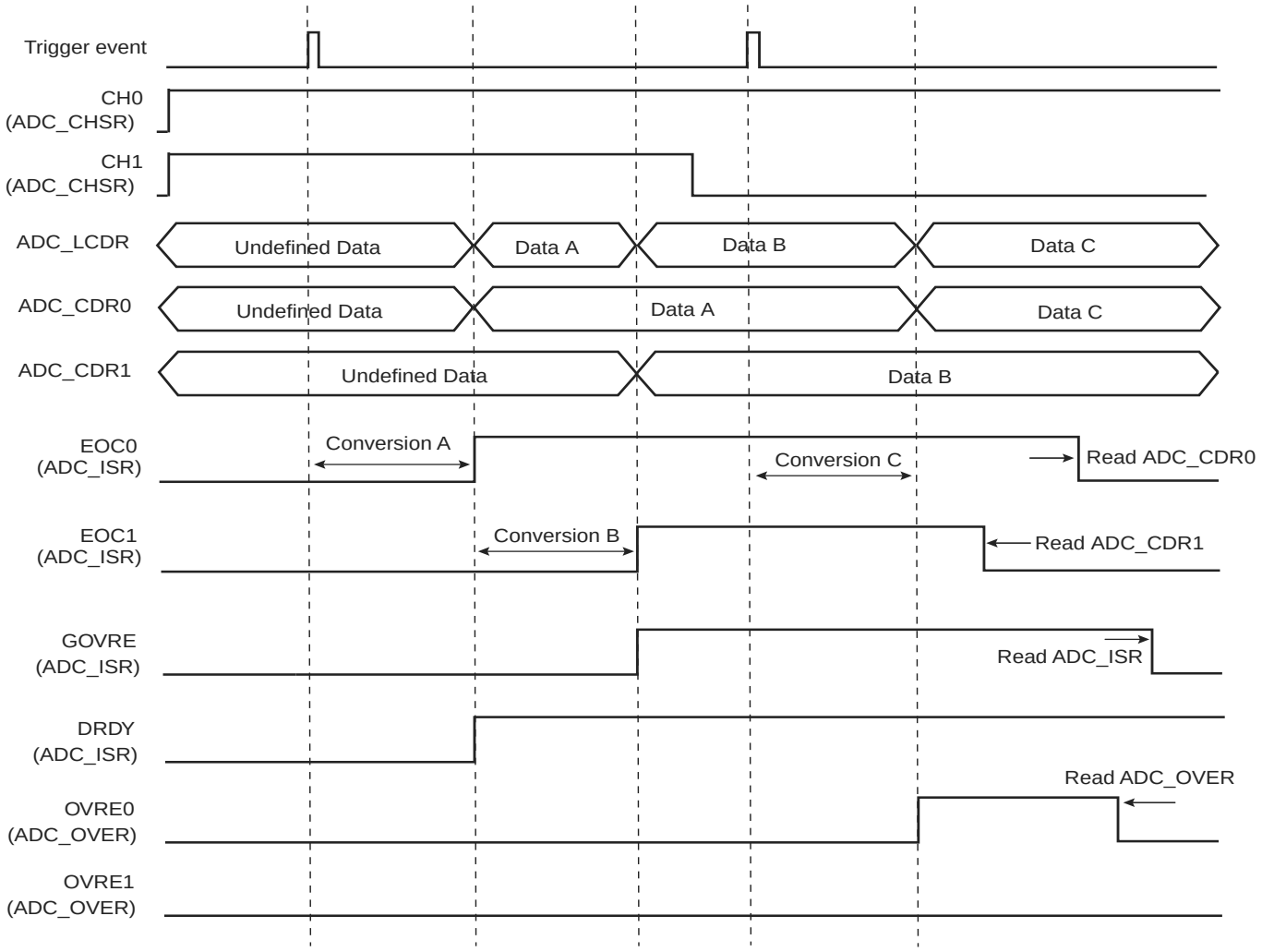
OVRE: Overrun Error Interrupt Disable

TXEMPTY: TXEMPTY Interrupt Disable

UNRE: SPI Underrun Error Interrupt Disable

NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Disable

Figure 41-4: EOCx, OVREx and GOVREx Flag Behavior



Warning: If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable.

41.6.6 Conversion Triggers

Conversions of the active analog channels are started with a software or hardware trigger. The software trigger is provided by writing the Control register (ADC_CR) with the START bit at 1.

The hardware trigger can be one of the TIOA outputs of the Timer Counter channels or the external trigger input of the ADC (ADTRG).

The TRGMOD field in the ADC Trigger Register (ADC_TRGR) selects the hardware trigger from the following:

- any edge, either rising or falling or both, detected on the external trigger pin ADTRG
- the Pen Detect, depending on how the PENDET bit is set in the ADC Touchscreen Mode Register (ADC_TSMR)
- a continuous trigger, meaning the ADC Controller restarts the next sequence as soon as it finishes the current one
- a periodic trigger, which is defined by programming the TRGPER field in ADC_TRGR

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence according to configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, ADC_TSMR.

If a hardware trigger is selected, the start of a conversion is triggered after a delay starting at each rising edge of the selected signal. Due to asynchronous handling, the delay may vary in a range of two peripheral clock periods to one ADC clock period.

RXEN: Receive Enable

0: Receive is disabled.

1: Receive is enabled.

43.4 I/O Lines Description

Table 43-1: I/O Lines Description

Name	Description	Type
LCD_PWM	Contrast control signal, using Pulse Width Modulation	Output
LCD_HSYNC	Horizontal Synchronization Pulse	Output
LCD_VSYNC	Vertical Synchronization Pulse	Output
LCD_DAT[23:0]	LCD 24-bit data bus	Output
LCD_DEN	Data Enable	Output
LCD_DISP	Display Enable signal	Output
LCD_PCLK	Pixel Clock	Output

43.5 Product Dependencies

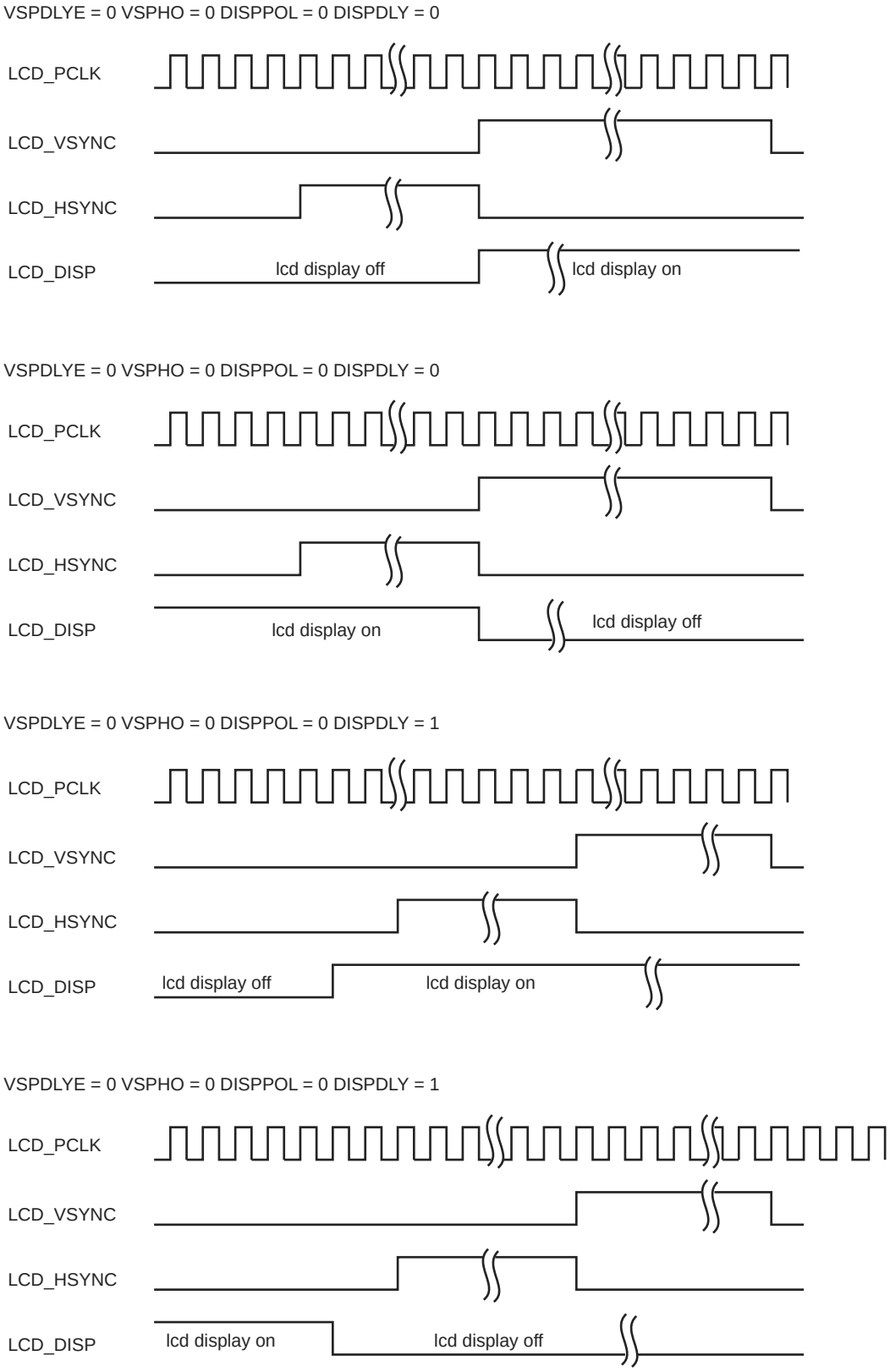
43.5.1 I/O Lines

The pins used for interfacing the LCDC may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the LCDC are not used by the application, they can be used for other purposes by the PIO Controller.

Table 43-2: I/O Lines

Instance	Signal	I/O Line	Peripheral
LCDC	LCDDAT0	PC0	A
LCDC	LCDDAT1	PC1	A
LCDC	LCDDAT2	PC2	A
LCDC	LCDDAT3	PC3	A
LCDC	LCDDAT4	PC4	A
LCDC	LCDDAT5	PC5	A
LCDC	LCDDAT6	PC6	A
LCDC	LCDDAT7	PC7	A
LCDC	LCDDAT8	PC8	A
LCDC	LCDDAT9	PC9	A
LCDC	LCDDAT10	PC10	A
LCDC	LCDDAT11	PC11	A
LCDC	LCDDAT12	PC12	A
LCDC	LCDDAT13	PC13	A
LCDC	LCDDAT14	PC14	A
LCDC	LCDDAT15	PC15	A
LCDC	LCDDAT16	PC16	A
LCDC	LCDDAT17	PC17	A
LCDC	LCDDAT18	PC18	A
LCDC	LCDDAT19	PC19	A
LCDC	LCDDAT20	PC20	A

Figure 43-5: DISP Signal Timing Diagram



47.14 DDRSDRC Timings

The DDRSDRC controller satisfies the timings of standard DDR2, LP-DDR, SDR and LP-SDR modules.

DDR2, LP-DDR and SDR timings are specified by the JEDEC standard.

Supported speed grade limitations:

- DDR2-400 limited at 133 MHz clock frequency (1.8V, 30 pF on data/control, 10 pF on CK/CK#)
- LP-DDR (1.8V, 30 pF on data/control, 10pF on CK)
  t_{cyc} = 5.0 ns, f_{max} = 125 MHz
  t_{cyc} = 6.0 ns, f_{max} = 110 MHz
  t_{cyc} = 7.5 ns, f_{max} = 95 MHz
- SDR-100 (3.3V, 50 pF on data/control, 10 pF on CK)
- SDR-133 (3.3V, 50 pF on data/control, 10 pF on CK)
- LP-SDR-133 (1.8V, 30 pF on data/control, 10 pF on CK)

47.15 Peripheral Timings

47.15.1 SPI

47.15.1.1 Maximum SPI Frequency

The following formulas give maximum SPI frequency in Master read and write modes and in Slave read and write modes.

- Master Write Mode

  The SPI only sends data to a slave device such as an LCD, for example. The limit is given by SPI_2 (or SPI_5) timing. Since it gives a maximum frequency above the maximum pad speed (see Section 47.9 “I/Os”), the maximum SPI frequency is defined by the pin FreqMax value.

- Master Read Mode

  $$f_{SPCK}Max = \frac{1}{SPI_0(orSPI_3) + t_{valid}}$$

  t_{valid} is the slave time response to output data after deleting an SPCK edge. F or a non-volatile memory with t_{valid} (or t_v) = 12 ns Max, f_{SPCK}Max = 47.1 MHz @ V_{DDIO} = 3.3V.

- Slave Read Mode

  In slave mode, SPCK is the input clock for the SPI. The max SPCK frequency is given by setup and hold timings SPI_7/SPI_8(or SPI_{10}/SPI_{11}). Since this gives a frequency well above the pad limit, the limit in slave read mode is given by SPCK pad.

- Slave Write Mode

  $$f_{SPCK}Max = \frac{1}{SPI_6(orSPI_9) + t_{setup}}$$

  t_{setup} is the setup time from the master before sampling data (12 ns).

  This gives f_{SPCK}Max = 44.6 MHz @ V_{DDIO} = 3.3V.

47.15.1.2 Timing Conditions

Timings are given assuming a capacitance load on MISO, SPCK and MOSI.

Table 47-31: Capacitance Load for MISO, SPCK and MOSI (product dependent)

	Corner	
Supply	**Max**	**Min**
3.3V	40 pF	5 pF
1.8V	20 pF	5 pF

Table 47-32: SPI Timings with 3.3V Peripheral Supply (Continued)

Symbol	Parameter	Conditions	Min	Max	Unit
SPI_7	MOSI Setup time before SPCK rises		2.7	–	ns
SPI_8	MOSI Hold time after SPCK rises		0.2	–	ns
SPI_9	SPCK rising to MISO		2.5	13.8	ns
SPI_{10}	MOSI Setup time before SPCK falls		2.2	–	ns
SPI_{11}	MOSI Hold time after SPCK falls		0.6	–	ns
SPI_{12}	NPCS0 setup to SPCK rising		4.3	–	ns
SPI_{13}	NPCS0 hold after SPCK falling		0	–	ns
SPI_{14}	NPCS0 setup to SPCK falling		3.8	–	ns
SPI_{15}	NPCS0 hold after SPCK rising		0	–	ns
SPI_{16}	NPCS0 falling to MISO valid		–	14.5	ns

Table 47-33: SPI Timings with 1.8V Peripheral Supply

Symbol	Parameter	Conditions	Min	Max	Unit
Master Mode					
SPI_{SPCK}	SPI Clock		–	66	MHz
SPI_0	MISO Setup time before SPCK rises		16.3	–	ns
SPI_1	MISO Hold time after SPCK rises		0	–	ns
SPI_2	SPCK rising to MOSI		0	6.9	ns
SPI_3	MISO Setup time before SPCK falls		15.1	–	ns
SPI_4	MISO Hold time after SPCK falls		0	–	ns
SPI_5	SPCK falling to MOSI		0	7.0	ns
Slave Mode					
SPI_6	SPCK falling to MISO		3.5	16.8	ns
SPI_7	MOSI Setup time before SPCK rises		2.9	–	ns
SPI_8	MOSI Hold time after SPCK rises		0.3	–	ns
SPI_9	SPCK rising to MISO		3.3	16.4	ns
SPI_{10}	MOSI Setup time before SPCK falls		2.4	–	ns
SPI_{11}	MOSI Hold time after SPCK falls		0.7	–	ns
SPI_{12}	NPCS0 setup to SPCK rising		4.5	–	ns
SPI_{13}	NPCS0 hold after SPCK falling		0	–	ns
SPI_{14}	NPCS0 setup to SPCK falling		3.9	–	ns
SPI_{15}	NPCS0 hold after SPCK rising		0	–	ns
SPI_{16}	NPCS0 falling to MISO valid		–	17.3	ns

Table 48-7: BGA247 Ball Information

Ball pitch	0.5 mm +/- 0.05
Ball Diameter	0.3 mm +/- 0.05

Table 48-8: BGA247 Soldering Information

Ball Land	0.35 mm +/- 0.05
Solder Mask Opening	0.27 mm +/- 0.05

Table 48-9: Device and BGA247 Package Maximum Weight

177	mg

Table 48-10: BGA247 Package Characteristics

Moisture Sensitivity Level	3

Table 48-11: BGA247 Package Reference

JEDEC Drawing Reference	none
JESD97 Classification	e1

Table 48-12: 247-ball TFBGA – Recommended Soldering Profile from J-STD-20

Profile Feature	**Green Package**
Average Ramp-up Rate (217°C to Peak)	3°C/sec. max.
Preheat Temperature 175°C ±25°C	180 sec. max.
Temperature Maintained Above 217°C	60 sec. to 150 sec.
Time within 5°C of Actual Peak Temperature	20 sec. to 40 sec.
Peak Temperature Range	260 +0°C
Ramp-down Rate	6°C/sec. max.
Time 25°C to Peak Temperature	8 min. max.

Note: It is recommended to apply a soldering temperature higher than 250°C.

A maximum of three reflow passes is allowed per component.

Doc. Rev. 11063L	Comments (Continued)
28-Oct-15	Section 29. ”Static Memory Controller (SMC)” (cont’d) Section 29.9 ”Standard Read and Write Protocols”: renamed, updated and moved subsection “Write Protected Registers” to Section 29.15 ”Register Write Protection” Section 29.14.2 ”Byte Access Type in Page Mode”: instance of “SMC_REGISTER” corrected to “SMC Mode Register (SMC_MODE)” Table 29-9 ”Register Mapping”: removed registers SMC_DELAY1–SMC_DELAY8 (offset range 0xC0–0xDC now reserved) Added sentence about disabling write protection in Section 29.16.1 ”SMC Setup Register”, Section 29.16.2 ”SMC Pulse Register” and Section 29.16.3 ”SMC Cycle Register” Removed section “SMC DELAY I/O Register” Section 29.16.4 ”SMC Mode Register”: added sentence about disabling write protection; updated descriptions of fields READ_MODE, WRITE_MODE, EXNW_MODE, BAT, DBW, and PS Updated Section 29.16.5 ”SMC Write Protection Mode Register” and Section 29.16.6 ”SMC Write Protection Status Register”
	Section 30. ”DDR SDR SDRAM Controller (DDRSDRC)” Removed instances of or references to “temperature compensated self refresh”, “TCR” field, and acronym “TCSR” Section 30.5.1 ”SDRAM Controller Write Cycle”: added note defining TWRD Section 30.5.4 ”Power Management”: added note specifying that possible SDRAM constraint of 4K cycles of burst auto-refresh is not supported Figure 30-12 "Single Read Access, Row Closed, Latency = 3, DDR2-SDRAM Device": corrected “Latency = 2” to “Latency = 3” and inserted third cycle Figure 30-16 "Burst Read Access, Latency = 2, SDR-SDRAM Devices": removed DQS[1:0] waveform Updated Section 30.5.6 ”Register Write Protection” Section 30.6.3 ”SDR-SDRAM Address Mapping for 32-bit Memory Data Bus Width”: updated footnote 2 Table 30-16 ”Register Mapping”: added reserved offset 0x28 Removed “Reset” line from individual register descriptions (reset values are provided in Table 30-16 ”Register Mapping”) Section 30.7.7 ”DDRSDRC Low-power Register”: updated TIMEOUT field description Section 30.7.10 ”DDRSDRC High Speed Register”: updated DIS_ANTICIP_READ bit description Section 30.7.11 ”DDRSDRC Write Protection Mode Register”: updated WPEN bit description
	Section 31. ”DMA Controller (DMAC)” Section 31.2 ”Embedded Characteristics”: added bullet “Register Write Protection” Added Section 31.5 ”Product Dependencies” Section 31.6.3.1 ”Software Handshaking”: instance of “last transaction register” corrected to “Software Last Transfer Flag Register” Section 31.6.4.3 ”Ending Multi-buffer Transfers”: in second paragraph, “automatic mode is disabled by writing a ‘1’ in DMAC_CTRLBx.AUTO bit” corrected to “automatic mode is disabled by clearing the DMAC_CTRLBx.AUTO bit” Section 31.6.6 ”Disabling a Channel Prior to Transfer Completion”: in last paragraph, “by writing a ‘1’ to the DMAC_CHER.RESx field register” changed to “by setting the DMAC_CHDR.RESx bit” Section 31.6.6.1 ”Abnormal Transfer Termination”: - in first sentence, “channel enable bit, DMAC_CHDR.ENAx” corrected to “channel enable bit, DMAC_CHER.ENAx” - in second paragraph, “global enable bit in the DMAC Configuration Register (DMAC_EN.ENABLE bit)” corrected to “general enable bit in the DMAC Enable Register (DMAC_EN.ENABLE)” Section 31.6.7 ”Register Write Protection”: updated title (was “Write Protection Registers”) and content