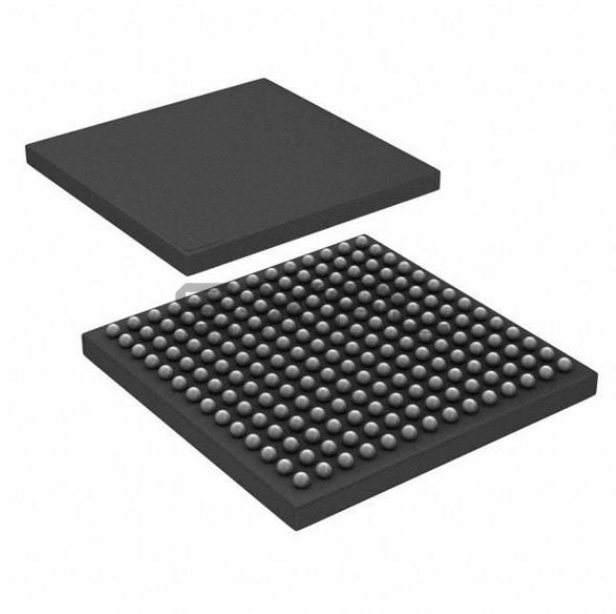


Welcome to E-XFL.COM

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	ARM® Cortex®-A5
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	500MHz
Co-Processors/DSP	Multimedia; NEON™ MPE
RAM Controllers	LPDDR1, LPDDR2, LPDDR3, DDR2, DDR3, DDR3L, QSPI
Graphics Acceleration	Yes
Display & Interface Controllers	Keyboard, LCD, Touchscreen
Ethernet	10/100Mbps (1)
SATA	-
USB	USB 2.0 + HSIC
Voltage - I/O	3.3V
Operating Temperature	-40°C ~ 85°C (TA)
Security Features	ARM TZ, Boot Security, Cryptography, RTIC, Secure Fusebox, Secure JTAG, Secure Memory, Secure RTC
Package / Case	196-TFBGA, CSBGA
Supplier Device Package	196-TFBGA (11x11)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsama5d23a-cur

21.8.4.2 Fast Interrupt Control

The fast interrupt logic of the AIC has no priority controller. The mode of interrupt source 0 is programmed with AIC_SMR and INTSEL = 0; the PRIOR field of this register is not used even if it reads what has been written. AIC_SMR.SRCTYPE enables programming the fast interrupt source to be rising-edge triggered or falling-edge triggered or high-level sensitive or low-level sensitive.

Writing 0x1 in AIC_IECR and AIC_IDCR respectively enables and disables the fast interrupt when INTSEL = 0. Bit 0 of AIC_IMR indicates whether the fast interrupt is enabled or disabled.

21.8.4.3 Fast Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and associated status bits.

Assuming that:

1. The Advanced Interrupt Controller has been programmed, AIC_SVR is loaded with the fast interrupt service routine address, and interrupt source 0 is enabled.
2. The Instruction at address 0x1C (FIQ exception vector address) is required to vector the fast interrupt. Load the PC with the absolute address of the interrupt handler.
3. The user does not need nested fast interrupts.

When nFIQ is asserted, if bit “F” of CPSR is 0, the sequence is:

1. The CPSR is stored in SPSR_fiq, the current value of the program counter is loaded in the FIQ link register (R14_FIQ) and the program counter (R15) is loaded with 0x1C. In the following cycle, during fetch at address 0x20, the ARM core adjusts R14_fiq, decrementing it by four.
2. The ARM core enters FIQ mode.
3. The routine must read AIC1_CISR to know if the interrupt is the FIQ or a Secure Internal interrupt.

```
        ldr         r1, =REG_SAIC_CISR
        ldr         r1, [r1]
        cmp         r1, #AIC_CISR_NFIQ
        beq         get_fiqvec_addr
```

   If FIQ is active, it is processed in priority, even if another interrupt is active.

```
get_irqvec_addr
        ldr         r14, =REG_SAIC_IVR
        b           read_vec
get_fiqvec_addr
        ldr         r14, =REG_SAIC_FVR
 read_vec
        ldr         r0, [r14]
```

   Now r0 contains the correct vector address, IVR for a Secure Internal interrupt or FVR for FIQ.

   The system can branch to the routine pointed to by r0.

```
FIQ_Handler_Branch
        mov     r14, pc
        bx      r0
```

4. The previous step enables branching to the corresponding interrupt service routine. It is not necessary to save the link register R14_fiq and SPSR_fiq if nested fast interrupts are not needed.
5. The Interrupt Handler can then proceed as required. It is not necessary to save registers R8 to R13 because the FIQ mode has its own dedicated registers and registers R8 to R13 are banked. The other registers, R0 to R7, must be saved before being used, and restored at the end (before the next step).

   Note: If the fast interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase in order to de-assert interrupt source 0.

6. Finally, Link register R14_fiq is restored into the PC after decrementing it by four (with instruction **`SUB PC, LR, #4`** for example). This has the effect of returning from the interrupt to whatever was being executed before, loading the CPSR with the SPSR and masking or unmasking the fast interrupt depending on the state saved in the SPSR.

   Note: The “F” bit in SPSR is significant. If it is set, it indicates that the ARM core was just about to mask FIQ interrupts when the mask instruction was interrupted. Hence, when the SPSR is restored, the interrupted instruction is completed (FIQ is masked).

Another way to handle the fast interrupt is to map the interrupt service routine at the address of the ARM vector 0x1C. This method does not use vectoring, so that reading AIC_FVR must be performed at the very beginning of the handler operation. However, this method saves the execution of a branch instruction.

© 2017 Microchip Technology Inc.

21.9.4 AIC Interrupt Vector Register

Name: AIC_IVR

Address: 0xFC020010 (AIC), 0xF803C010 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
IRQV							

23	22	21	20	19	18	17	16
IRQV							

15	14	13	12	11	10	9	8
IRQV							

7	6	5	4	3	2	1	0
IRQV							

IRQV: Interrupt Vector Register

The Interrupt Vector Register contains the vector programmed by the user in the Source Vector Register corresponding to the current interrupt.

The Source Vector Register is indexed using the current interrupt number when the Interrupt Vector Register is read.

When there is no current interrupt, the Interrupt Vector Register reads the value stored in AIC_SPU.

© 2017 Microchip Technology Inc.

to 63. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Reset command is now issued.

8. A pause of at least t_{INIT5} must be observed before issuing any commands.
9. A Calibration command is issued to the low-power DDR3-SDRAM. Program the type of calibration in the Configuration register (MPDDRC_CR): set the ZQ field to 3. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 10. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The ZQ Calibration command is now issued. Program the type of calibration in the MPDDRC_CR: set the ZQ field to 2.
10. A Mode register Write command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 1. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
11. A Mode register Write command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 2. The Mode register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular CAS Latency. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
12. A Mode register Write command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field 7 and the MRS field to 3. The Mode register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular Drive Strength and Slew Rate. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
13. A Mode register Write command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 16. The Mode register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular Partial Array Self Refresh (PASR). Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Write command is now issued.
14. In the DDR Configuration register (SFR_DDRCFG), the application must write a ‘1’ to bits 17 and 16 to open the input buffers.
15. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the Mode register (MPDDRC_MR). The application must configure the MODE field to 1 in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command.
16. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 5. The Mode register Read command cycle is used to read the LPDDR3 Manufacturer ID from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. The LPDDR3 Manufacturer ID is set in register MPDDRC_MD. See Section 36.7.8 “MPDDRC Memory Device Register”.
17. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 6. The Mode register Read command cycle is used to read the Revision ID1 from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Revision ID1 is set in register MPDDRC_MD. See Section 36.7.8 “MPDDRC Memory Device Register”.
18. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 8. The Mode register Read command cycle is used to read memory organization (I/O width, Density, Type) from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Memory organization is set in register MPDDRC_MD. See Section 36.7.8 “MPDDRC Memory Device Register”.
19. A Mode register Read command is issued to the low-power DDR3-SDRAM. In MPDDRC_MR, configure the MODE field to 7 and the MRS field to 0. The Mode register Read command cycle is used to read the device information (RZQI, DAI) from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode register Read command is now issued. Device information RZQI is set in register Timing Calibration (see Section 36.7.11 “MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register”) and DAI is set in Mode register (see Section 36.7.1 “MPDDRC Mode Register”).
20. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowl-

Figure 36-11: Burst Read Access, Latency = 2, DDR-SDRAM Devices

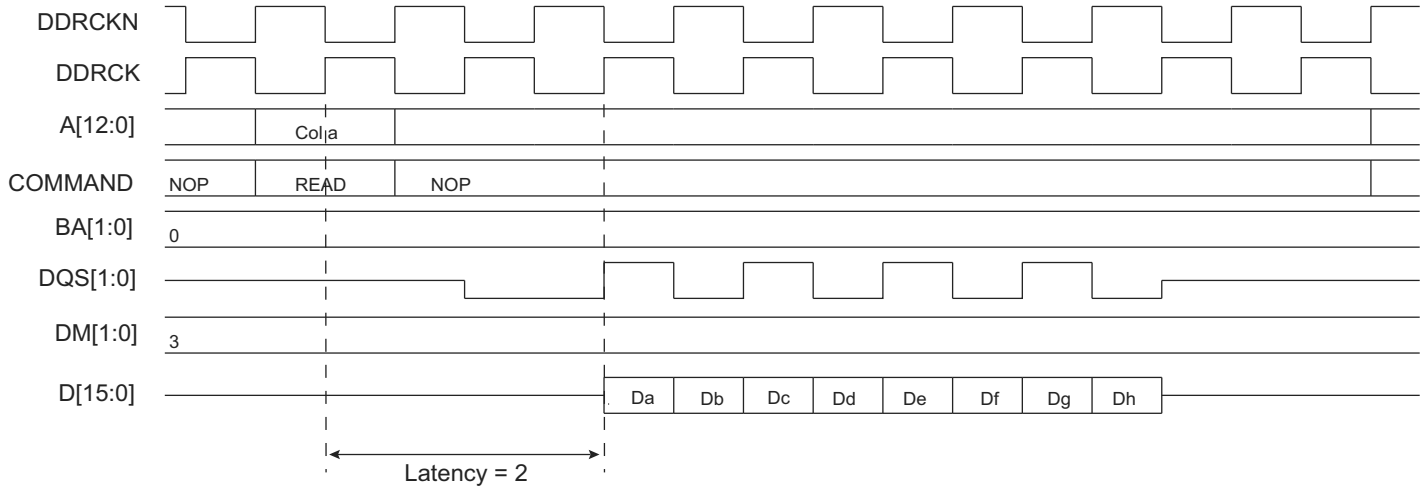
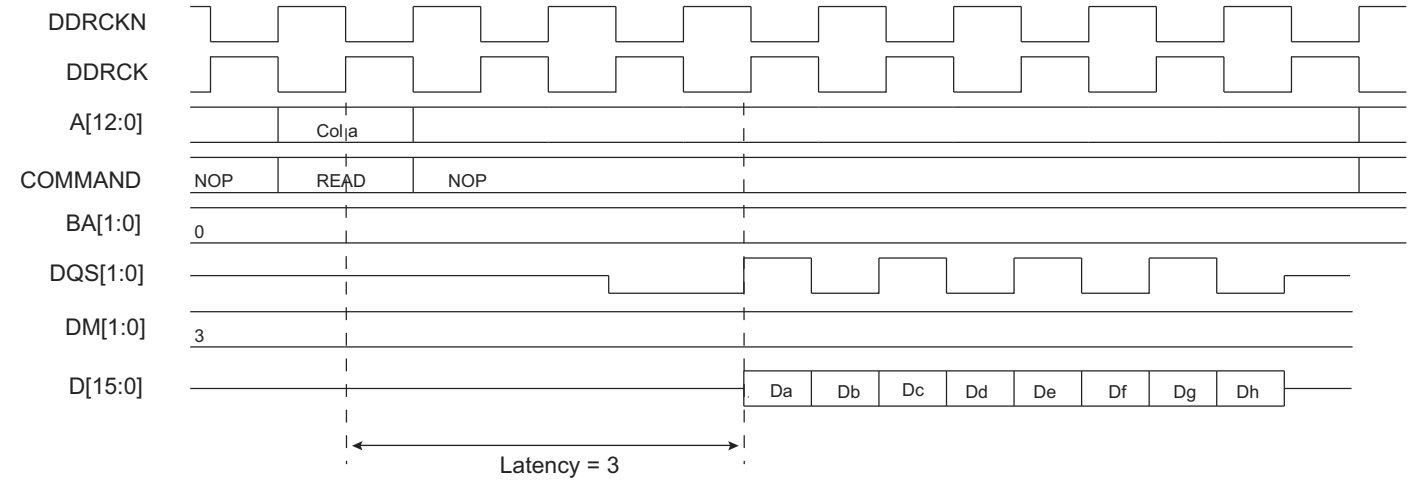


Figure 36-12: Burst Read Access, Latency = 3, DDR2-SDRAM Devices



36.5.2.1 All Banks Autorefresh

The All Banks Autorefresh command performs a refresh operation on all banks. An autorefresh command is used to refresh the external device. Refresh addresses are generated internally by the DDR-SDRAM device and incremented after each autorefresh automatically. The MPDDRC generates these autorefresh commands periodically. A timer is loaded in the MPDDRC_RTR with the value that indicates the number of clock cycles between refresh cycles (see Section 36.7.2 “MPDDRC Refresh Timer Register”). When the MPDDRC initiates a refresh of the DDR-SDRAM device, internal memory accesses are not delayed. However, if the CPU tries to access the DDR-SDRAM device, the slave indicates that the device is busy. A refresh request does not interrupt a burst transfer in progress. This feature is activated by setting Per-bank Refresh bit (REF_PB) to 0 in the MPDDRC_RTR (see Section 36.7.2 “MPDDRC Refresh Timer Register”).

36.5.2.2 Per-bank Autorefresh

The low-power DDR2-SDRAM and low-power DDR3-SDRAM embeds a new Per-bank Refresh command which performs a refresh operation on the bank scheduled by the bank counter in the memory device. The Per-bank Refresh command is executed in a fixed sequence order of round-robin type: “0-1-2-3-4-5-6-7-0-1-...”. The bank counter is automatically cleared upon issuing a RESET command or when exiting from Self-refresh mode, in order to ensure the synchronism between SDRAM memory device and the MPDDRC. The bank addressing for the Per-bank Refresh count is the same as established in the Single-bank Precharge command. This feature is activated by setting the Per-bank Refresh bit (REF_PB) to 1 in the MPDDRC_RTR (see Section 36.7.2 “MPDDRC Refresh Timer Register”). This feature masks the latency due to the refresh procedure. The target bank is inaccessible during the Per-bank Refresh cycle period (t_{RFCpb}), however other banks within the device are accessible and may be addressed during the “Per-bank Refresh” cycle. During the REFpb operation, any bank other than the one being refreshed can be maintained in active state or accessed by a read or a write command. When the “Per-bank Refresh” cycle is completed, the affected bank will be in idle state.

© 2017 Microchip Technology Inc.

37.20.19 PMECC Redundancy x Register

Name: HSMC_PMECCx [x=0..13] [sec_num=0..7]

Address: 0xF80140B0 [0][0] .. 0xF80140E4 [13][0]
0xF80140F0 [0][1] .. 0xF8014124 [13][1]
0xF8014130 [0][2] .. 0xF8014164 [13][2]
0xF8014170 [0][3] .. 0xF80141A4 [13][3]
0xF80141B0 [0][4] .. 0xF80141E4 [13][4]
0xF80141F0 [0][5] .. 0xF8014224 [13][5]
0xF8014230 [0][6] .. 0xF8014264 [13][6]
0xF8014270 [0][7] .. 0xF80142A4 [13][7]

Access: Read-only

31	30	29	28	27	26	25	24
ECC							

23	22	21	20	19	18	17	16
ECC							

15	14	13	12	11	10	9	8
ECC							

7	6	5	4	3	2	1	0
ECC							

ECC: BCH Redundancy

This register contains the remainder of the division of the codeword by the generator polynomial.

39.7.17 Base Layer Channel Disable Register

Name: LCDC_BASECHDR

Address: 0xF0000044

Access: Write-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
–	–	–	–	–	–	–	CHRST

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

CHDIS: Channel Disable

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

CHRST: Channel Reset

0: No effect

1: Resets the layer immediately. The frame is aborted.

© 2017 Microchip Technology Inc.

39.7.105 High-End Overlay Configuration Register 10

Name: LCDC_HEOCFG10

Address: 0xF00003B4

Access: Read/Write

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

23	22	21	20	19	18	17	16
RKEY							

15	14	13	12	11	10	9	8
GKEY							

7	6	5	4	3	2	1	0
BKEY							

RKEY: Red Color Component Chroma Key

Reference Red chroma key used to match the Red color of the current overlay.

GKEY: Green Color Component Chroma Key

Reference Green chroma key used to match the Green color of the current overlay.

BKEY: Blue Color Component Chroma Key

Reference Blue chroma key used to match the Blue color of the current overlay.

© 2017 Microchip Technology Inc.

43.7 Audio Class D Amplifier (CLASSD) User Interface

Table 43-5: Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	CLASSD_CR	Write-only	–
0x04	Mode Register	CLASSD_MR	Read/Write	0x00010022
0x08	Interpolator Mode Register	CLASSD_INTPMR	Read/Write	0x00304E4E
0x0C	Interpolator Status Register	CLASSD_INTSR	Read-only	0x00000000
0x10	Transmit Holding Register	CLASSD_THR	Read/Write	0x00000000
0x14	Interrupt Enable Register	CLASSD_IER	Write-only	–
0x18	Interrupt Disable Register	CLASSD_IDR	Write-only	–
0x1C	Interrupt Mask Register	CLASSD_IMR	Read/Write	0x00000000
0x20	Interrupt Status Register	CLASSD_ISR	Read-only	0x00000000
0x24–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	CLASSD_WPMR	Read/Write	0x00000000
0xE8–0xFC	Reserved	–	–	–

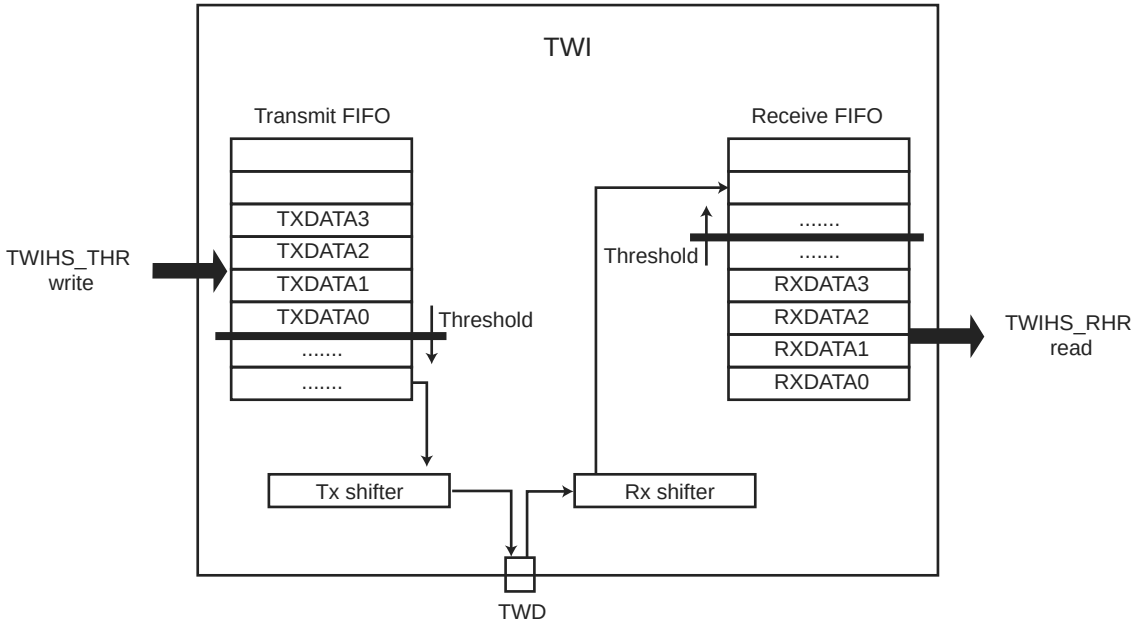
© 2017 Microchip Technology Inc.

46.6.3.15 FIFOs

The TWIHS includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in the TWIHS_CR. It is recommended to disable both Master and Slave modes before enabling or disabling the FIFO (MSDIS and SVDIS bit in TWIHS_CR).

Writing the FIFOEN bit to ‘1’ will enable a 16-data Transmit FIFO and a 16-data Receive FIFO.

Figure 46-28: FIFOs Block Diagram



46.7.17 TWIHS SleepWalking Matching Register

Name: TWIHS_SWMR

Address: 0xF802804C (0), 0xFC02804C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
DATAM							

23	22	21	20	19	18	17	16
–	SADR3						

15	14	13	12	11	10	9	8
–	SADR2						

7	6	5	4	3	2	1	0
–	SADR1						

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

SADR1: Slave Address 1

Slave address 1. The TWIHS module matches on this additional address if SADR1EN bit is enabled.

SADR2: Slave Address 2

Slave address 2. The TWIHS module matches on this additional address if SADR2EN bit is enabled.

SADR3: Slave Address 3

Slave address 3. The TWIHS module matches on this additional address if SADR3EN bit is enabled.

DATAM: Data Match

The TWIHS module extends the SleepWalking matching process to the first received data, comparing it with DATAM if DATAMEN bit is enabled.

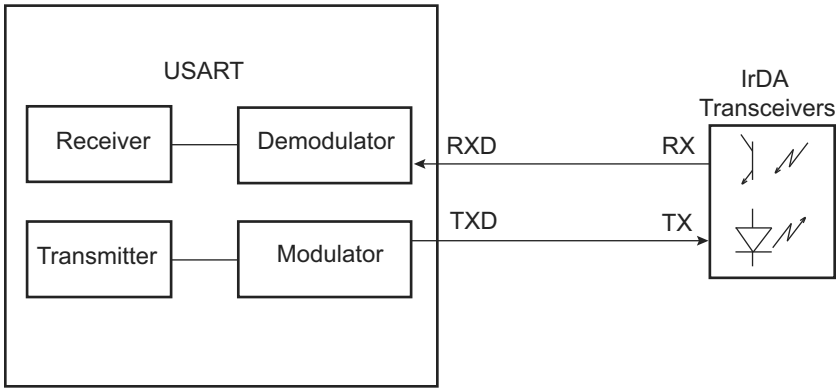
© 2017 Microchip Technology Inc.

47.7.5 IrDA Mode

The USART features an IrDA mode supplying half-duplex point-to-point wireless communication. It embeds the modulator and demodulator which allows a glueless connection to the infrared transceivers, as shown in Figure 47-33. The modulator and demodulator are compliant with the IrDA specification version 1.1 and support data transfer speeds ranging from 2.4 kbit/s to 115.2 kbit/s.

The USART IrDA mode is enabled by setting the FLEX_US_MR.USART_MODE field to the value 0x8. The IrDA Filter Register (FLEX_US_IF) allows configuring the demodulator filter. The USART transmitter and receiver operate in a normal Asynchronous mode and all parameters are accessible. Note that the modulator and the demodulator are activated.

Figure 47-33: Connection to IrDA Transceivers



The receiver and the transmitter must be enabled or disabled according to the direction of the transmission to be managed.

To receive IrDA signals, the following needs to be done:

- Disable TX and Enable RX
- Configure the TXD pin as PIO and set it as an output to 0 (to avoid LED transmission). Disable the internal pullup (better for power consumption).
- Receive data

47.7.5.1 IrDA Modulation

For baud rates up to and including 115.2 kbit/s, the RZI modulation scheme is used. “0” is represented by a light pulse of 3/16th of a bit time. Some examples of signal pulse duration are shown in Table 47-12.

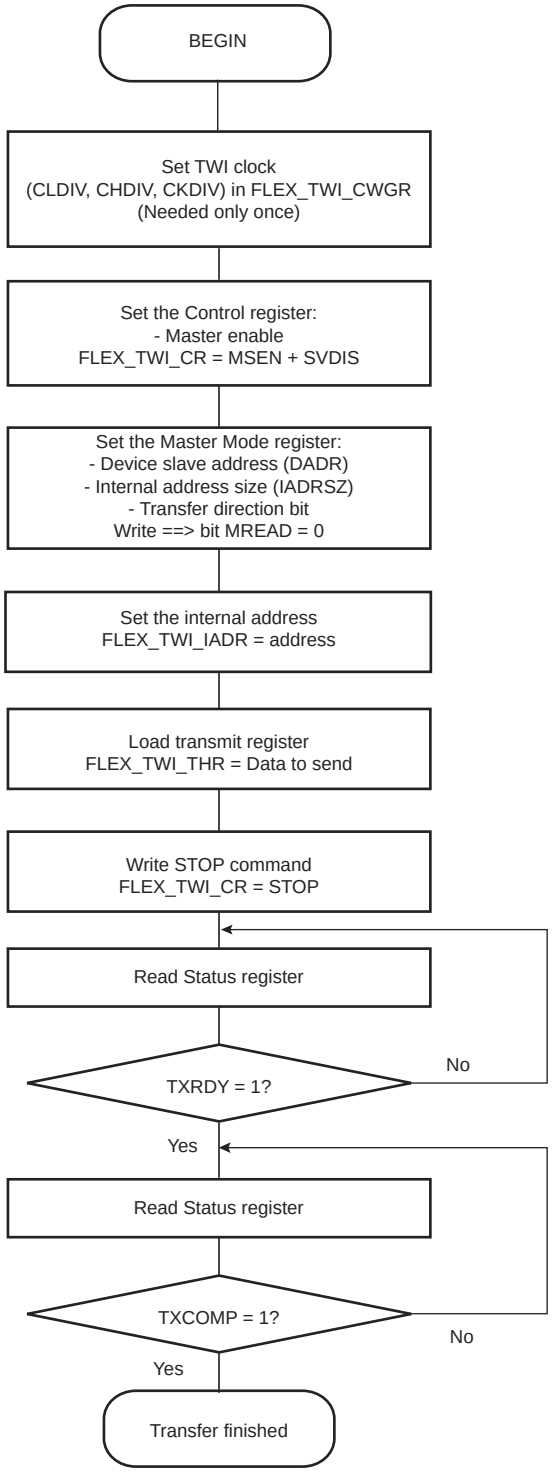
Table 47-12: IrDA Pulse Duration

Baud Rate	Pulse Duration (3/16)
2.4 kbit/s	78.13 µs
9.6 kbit/s	19.53 µs
19.2 kbit/s	9.77 µs
38.4 kbit/s	4.88 µs
57.6 kbit/s	3.26 µs
115.2 kbit/s	1.63 µs

Figure 47-34 shows an example of character transmission.

© 2017 Microchip Technology Inc.

Figure 47-96: TWI Write Operation with Single Data Byte and Internal Address



© 2017 Microchip Technology Inc.

To prevent any single software error from corrupting TWI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable) bit in the TWI Write Protection Mode Register (FLEX_TWI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the TWI Write Protection Status Register (FLEX_TWI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_TWI_WPSR.

The following register(s) can be write-protected when WPEN is set:

- TWI Slave Mode Register
- TWI Clock Waveform Generator Register
- TWI SMBus Timing Register
- TWI SleepWalking Matching Register

© 2017 Microchip Technology Inc.

47.10.66 TWI Status Register

Name: FLEX_TWI_SR

Address: 0xF8034620 (0), 0xF8038620 (1), 0xFC010620 (2), 0xFC014620 (3), 0xFC018620 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	SDA	SCL
23	22	21	20	19	18	17	16
LOCK	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK
15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCLWS	ARBLST	NACK
7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP

TXCOMP: Transmission Completed (cleared by writing FLEX_TWI_THR)

TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both the holding register and the internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in Figure 47-87 and in Figure 47-89.

TXCOMP used in Slave mode:

0: As soon as a Start is detected.

1: After a Stop or a Repeated Start + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in Figure 47-116, Figure 47-117, Figure 47-118 and Figure 47-119.

RXRDY: Receive Holding Register Ready (cleared when reading FLEX_TWI_RHR)

When FIFOs are disabled:

0: No character has been received since the last FLEX_TWI_RHR read operation.

1: A byte has been received in FLEX_TWI_RHR since the last read.

RXRDY behavior in Master mode can be seen in Figure 47-89.

RXRDY behavior in Slave mode can be seen in Figure 47-114, Figure 47-117, Figure 47-118 and Figure 47-119.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read

1: At least one unread data is in the Receive FIFO

RXRDY behavior with FIFO enabled is illustrated in Section 47.9.6.6 “TXRDY and RXRDY Behavior”.

© 2017 Microchip Technology Inc.

47.10.73 TWI Transmit Holding Register

Name: FLEX_TWI_THR

Address: 0xF8034634 (0), 0xF8038634 (1), 0xFC010634 (2), 0xFC014634 (3), 0xFC018634 (4)

Access: Write-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
TXDATA							

Note: If FIFO is enabled (FIFOEN bit in FLEX_TWI_CR) and FLEX_TWI_FMR.TXRDYM = 0, refer to Section 47.9.6.7 ”TWI Single Data Mode” for details.

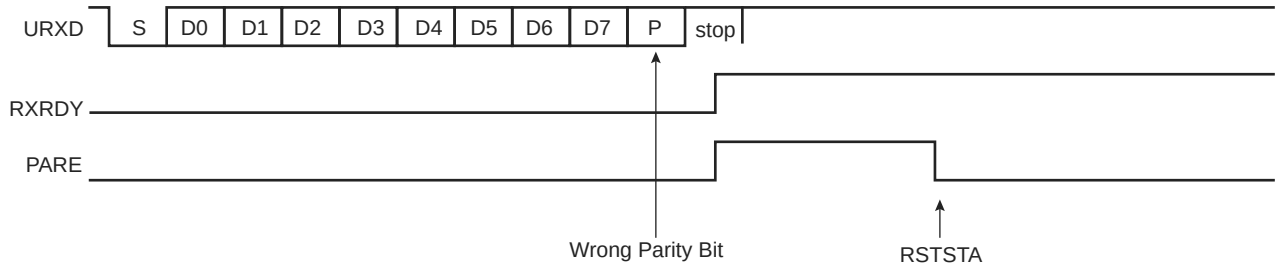
TXDATA: Master or Slave Transmit Holding Data

© 2017 Microchip Technology Inc.

48.5.2.5 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 Mode Register (UART_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in UART_SR is set at the same time RXRDY is set. The parity bit is cleared when UART_CR is written with the bit RSTSTA (Reset Status) at 1. If a new character is received before the reset status command is written, the PARE bit remains at 1.

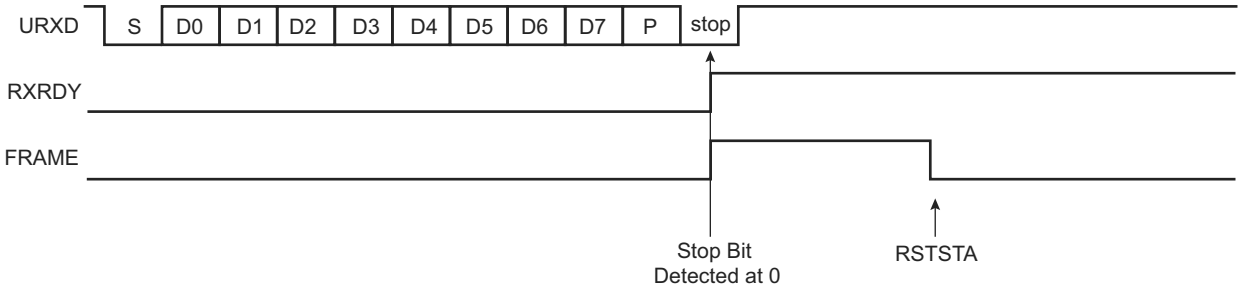
Figure 48-7: Parity Error



48.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in UART_SR is set at the same time the RXRDY bit is set. The FRAME bit remains high until the Control Register (UART_CR) is written with the bit RSTSTA at 1.

Figure 48-8: Receiver Framing Error



48.5.2.7 Receiver Digital Filter

The UART embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in the FILTER bit of UART_MR. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

48.5.2.8 Receiver Timeout

The Receiver Timeout provides support in handling variable-length frames. This feature detects an idle condition on the RXD line. When a timeout is detected, the bit TIMEOUT in the UART_SR rises and can generate an interrupt, thus indicating to the driver an end of frame.

The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Timeout register (UART_RTOR). If the TO field is written to 0, the Receiver Timeout is disabled and no timeout is detected. The TIMEOUT bit in the UART_SR remains at 0. Otherwise, the receiver loads an 8-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the TIMEOUT bit UART_SR rises. Then, the user can either:

- stop the counter clock until a new character is received. This is performed by writing a one to the STTTO (start Timeout) bit in the UART_CR. In this case, the idle state on RXD before a new character is received does not provide a timeout. This prevents having to handle an interrupt before a character is received and allows waiting for the next idle state on RXD after a frame is received, or
- obtain an interrupt while no character is received. This is performed by writing a one to the RETTO (Reload and Start Timeout) bit in the UART_CR. If RETTO is performed, the counter starts counting down immediately from the TO value. This enables generation of a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

© 2017 Microchip Technology Inc.

51.13.52 SDMMC Retuning Interrupt Status Enable Register

Name: SDMMC_RTISTER

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	TEVT

TEVT: Retuning Timer Event

0 (MASKED): The TEVT status flag in SDMMC_RTISTR is masked.

1 (ENABLED): The TEVT status flag in SDMMC_RTISTR is enabled.

© 2017 Microchip Technology Inc.

63. True Random Number Generator (TRNG)

63.1 Description

The True Random Number Generator (TRNG) passes the American *NIST Special Publication 800-22* (*A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*) and the *Diehard Suite of Tests*.

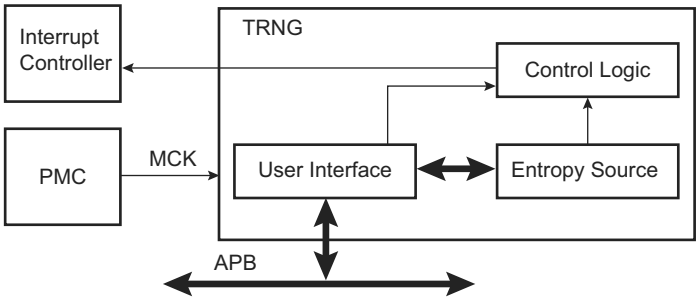
The TRNG may be used as an entropy source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3.

63.2 Embedded Characteristics

- Passes *NIST Special Publication 800-22* Test Suite
- Passes *Diehard Suite of Tests*
- May be Used as Entropy Source for seeding a NIST-approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3
- Provides a 32-bit Random Number Every 84 Clock Cycles

63.3 Block Diagram

Figure 63-1: TRNG Block Diagram



63.4 Product Dependencies

63.4.1 Power Management

The TRNG interface may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the TRNG user interface clock. The user interface clock is independent from any clock that may be used in the entropy source logic circuitry. The source of entropy can be enabled before enabling the user interface clock.

63.4.2 Interrupt Sources

The TRNG interface has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TRNG.

Table 63-1: Peripheral IDs

Instance	ID
TRNG	47

© 2017 Microchip Technology Inc.

Figure 64-3: Schedule Principle

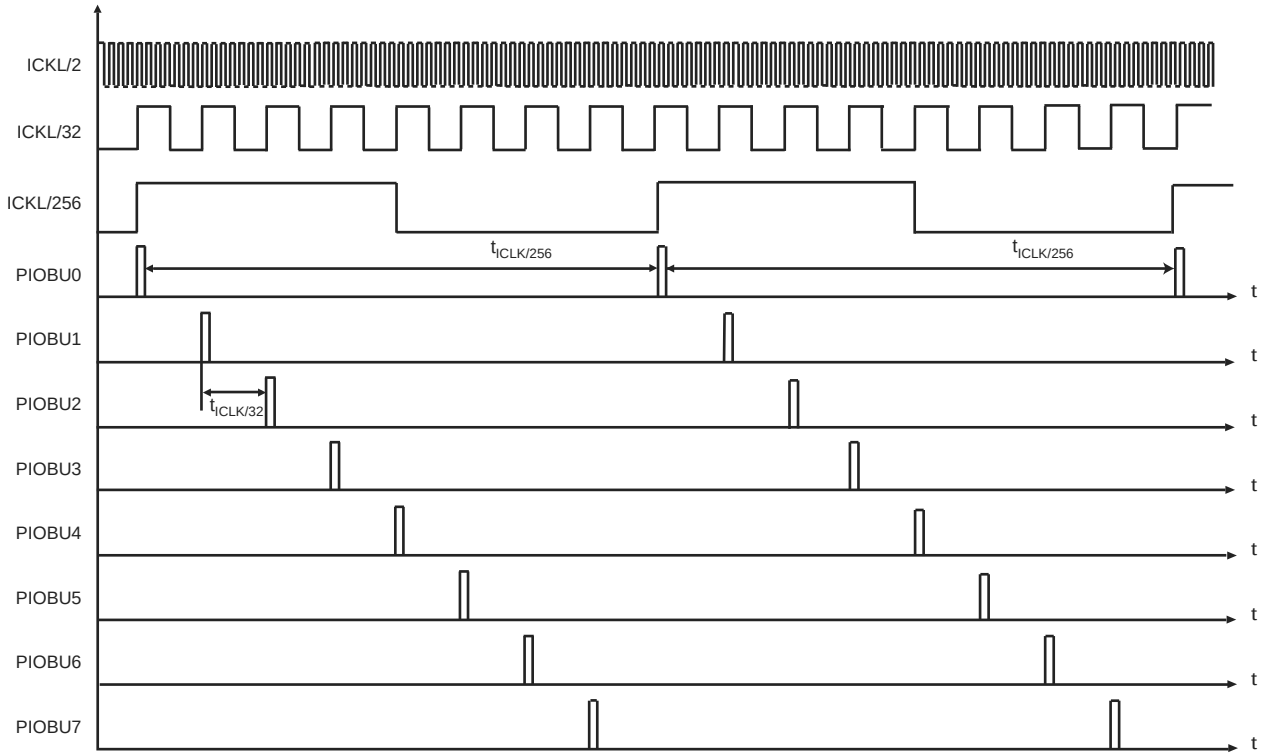


Table 64-3: Timings vs. f_{ICLK}

Timing	f_{ICLK} (kHz)			Units
	Min = 38	Typ = 64	Max = 90	
$t_{ICLK/2}$	53	31	22	µs
$t_{ICLK/32}$	842	500	356	µs
$t_{ICLK/256}$	6.74	4.00	2.84	ms

- PIOBUx Alarm Filtering in Static Mode

It is possible to filter the PIOBUx alarm detection by programming SECUMOD_PIOBUx.PIOBU_AFV. The steps are as follows:

1. A 9-bit counter is incremented each time the value present on the corresponding input is not the expected one.
2. An alarm is sent to the Protection Unit if the counter value reaches the value programmed in PIOBU_AFV.

The previous 9-bit counter is reset only if the value present on the input is correct and stable for a continuous programmable period defined by SECUMOD_PIOBUx.PIOBU_RFV (a second counter is used for that operation). See Figure 64-4.

© 2017 Microchip Technology Inc.

Table 72-4: SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 34. “Static Memory Controller (SMC)”
	Section 34.17.3 “NFC Initialization”: instances of “rbn” changed to “Ready/Busy”
	Section 34.20.3 “NFC Status Register”: bit RB_EDGE3 (bit 27) replaced by RB_EDGE0 (bit 24); updated RB_RISE and RB_FALL bit descriptions
	Bit RB_EDGE3 (bit 27) replaced by RB_EDGE0 (bit 24) in Section 34.20.4 “NFC Interrupt Enable Register”, Section 34.20.5 “NFC Interrupt Disable Register” and Section 34.20.6 “NFC Interrupt Mask Register”
	Deleted invalid addresses in Section 34.20.30 “PMECC Error Location SIGMA0 Register” and Section 34.20.31 “PMECC Error Location SIGMAx Register”
	Section 34.20.32 “PMECC Error Location x Register”: register index “x=0..23” corrected to “x=0..31”
	Section 34.20.36 “Timings Register”: removed RBNSEL field
	Section 35. “DMA Controller (XDMAC)”
	Added XDMAC_CCx.CSIZE configuration to Table 35-2 “DMA Channels Definition (XDMAC0)” and Table 35-3 “DMA Channels Definition (XDMAC1)”
	Table 35-5 “Register Mapping”:
	- XDMAC_GCFG access Read-only corrected to Read/Write
	- XDMAC_GWAC access Read-only corrected to Read/Write
	Section 35.9.2 “XDMAC Global Configuration Register”: access Read-only corrected to Read/Write
	Section 35.9.3 “XDMAC Global Weighted Arbiter Configuration Register”: access Read-only corrected to Read/Write
	Section 36. “LCD Controller (LCDC)”
	Updated ”Section 36.2 “Embedded Characteristics”
	Updated Section 36.6.1.1 “Pixel Clock Period Configuration”
	Section 37. “Ethernet MAC (GMAC)”
	Table 37-1 “GMAC Connections in Different Modes”: added table Note on GTXCK
	Updated Section 37.5.3 “Interrupt Sources”
	Section 37.7.1.2 “Receive Buffer List” and Section 37.7.1.3 “Transmit Buffer List”: added note at end of sections on queue pointer intilaization
	Section 37.8.107 “GMAC Transmit Buffer Queue Base Address Register Priority Queue x” and Section 37.8.108 “GMAC Receive Buffer Queue Base Address Register Priority Queue x”: changed sentence on register initialization
	Section 39. “USB Host High Speed Port (UHPHS)”
	Section 39.2 “Embedded Characteristics”: “X Hosts (A and B) High Speed (EHCI)” corrected to “2 Hosts (A and B) High Speed (EHCI)”
	Table 39-2 “Register Mapping”: inserted reserved offset 0x0C
	Section 39.7.19 “EHCI: REG06 - AHB Error Status”: instances of “INSNREG[8:4]” changed to “INSNREG06[8:4]”
	Section 40. “Audio Class D Amplifier (CLASSD)”
	Section 40.2 “Embedded Characteristics”: DSP clock frequency “11.289 MHz” corrected to “11.2896 MHz”
	Section 40.5.2 “Power Management”: field name “NOVRLAP” corrected to “NOVRVAL”
	Figure 40-21 “Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Single-ended Outputs”: changed title (was “Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Differential Outputs”)
	Section 42. “Synchronous Serial Controller (SSC)”
	Figure 42-19 “Interrupt Block Diagram”: “RXSYNC” renamed to “RXSYN”; “TXSYNC” renamed to “TXSYN”
	Section 42.8.10 “Register Write Protection”: in first sentence, “AIC behavior” corrected to “SSC behavior”