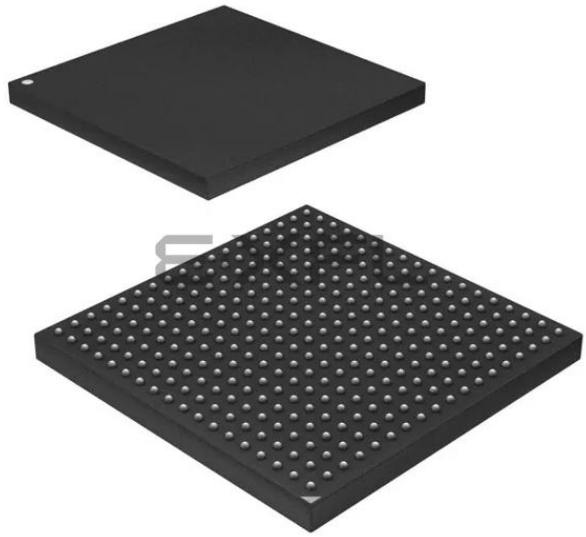


Welcome to E-XFL.COM

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	536MHz
Co-Processors/DSP	-
RAM Controllers	LPDDR, LPDDR2, DDR2
Graphics Acceleration	No
Display & Interface Controllers	LCD, Touchscreen
Ethernet	10/100/1000Mbps (1)
SATA	-
USB	USB 2.0 (3)
Voltage - I/O	1.2V, 1.8V, 3.3V
Operating Temperature	-40°C ~ 85°C (TA)
Security Features	AES, SHA, TDES, TRNG
Package / Case	324-LFBGA
Supplier Device Package	324-LFBGA (15x15)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsama5d33a-cur

20.3.1 General Purpose Backup Register x

Name: SYS_GPBRx

Address: 0xFFFFFE60

Access: Read/Write

31	30	29	28	27	26	25	24
GPBR_VALUE							

23	22	21	20	19	18	17	16
GPBR_VALUE							

15	14	13	12	11	10	9	8
GPBR_VALUE							

7	6	5	4	3	2	1	0
GPBR_VALUE							

These registers are reset at first power-up and on each loss of VDDBU.

- **GPBR_VALUE: Value of GPBR x**

26.17.25 PMC Peripheral Clock Status Register 1

Name: PMC_PCSR1

Address: 0xFFFFFD08

Access: Read-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

- **PIDx: Peripheral Clock x Status**

0: The corresponding peripheral clock is disabled.

1: The corresponding peripheral clock is enabled.

Note: PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”.

26.17.26 PMC Peripheral Control Register

Name: PMC_PCR

Address: 0xFFFFFD0C

Access: Read/Write

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

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

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

7	6	5	4	3	2	1	0
–	PID						

- **PID: Peripheral ID**

Peripheral ID selection from PID2 to the maximum PID number. This refers to identifiers as defined in the section “Peripheral Identifiers”.

Only the following peripherals can have a DIV value greater than 0: ADC, SSCx, CANx, USARTx, UARTx, TWIx, SPIx, and TCx.

- **CMD: Command**

0: Read mode

1: Write mode

- **DIV: Divisor Value**

Only the following peripherals can be configured with divided clock (DIV > 0): ADC, SSCx, CANx, USARTx, UARTx, TWIx, SPIx, and TCx.

Among the PIDs supporting the divided clock, some require a DIV value configuration matching the maximum peripheral frequency. Refer to section “Power Consumption versus Modes” in the “Electrical Characteristics”.

Value	Name	Description
0	PERIPH_DIV_MCK	Peripheral clock is MCK
1	PERIPH_DIV2_MCK	Peripheral clock is MCK/2
2	PERIPH_DIV4_MCK	Peripheral clock is MCK/4
3	PERIPH_DIV8_MCK	Peripheral clock is MCK/8

DIV must not be changed while peripheral is in use or when the peripheral clock is enabled.

- **EN: Enable**

0: The selected peripheral clock is disabled.

1: The selected peripheral clock is enabled.

27.6.28 PIO Input Filter Slow Clock Status Register

Name: PIO_IFSCSR

Address: 0xFFFFF288 (PIOA), 0xFFFFF488 (PIOB), 0xFFFFF688 (PIOC), 0xFFFFF888 (PIOD), 0xFFFFFA88 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

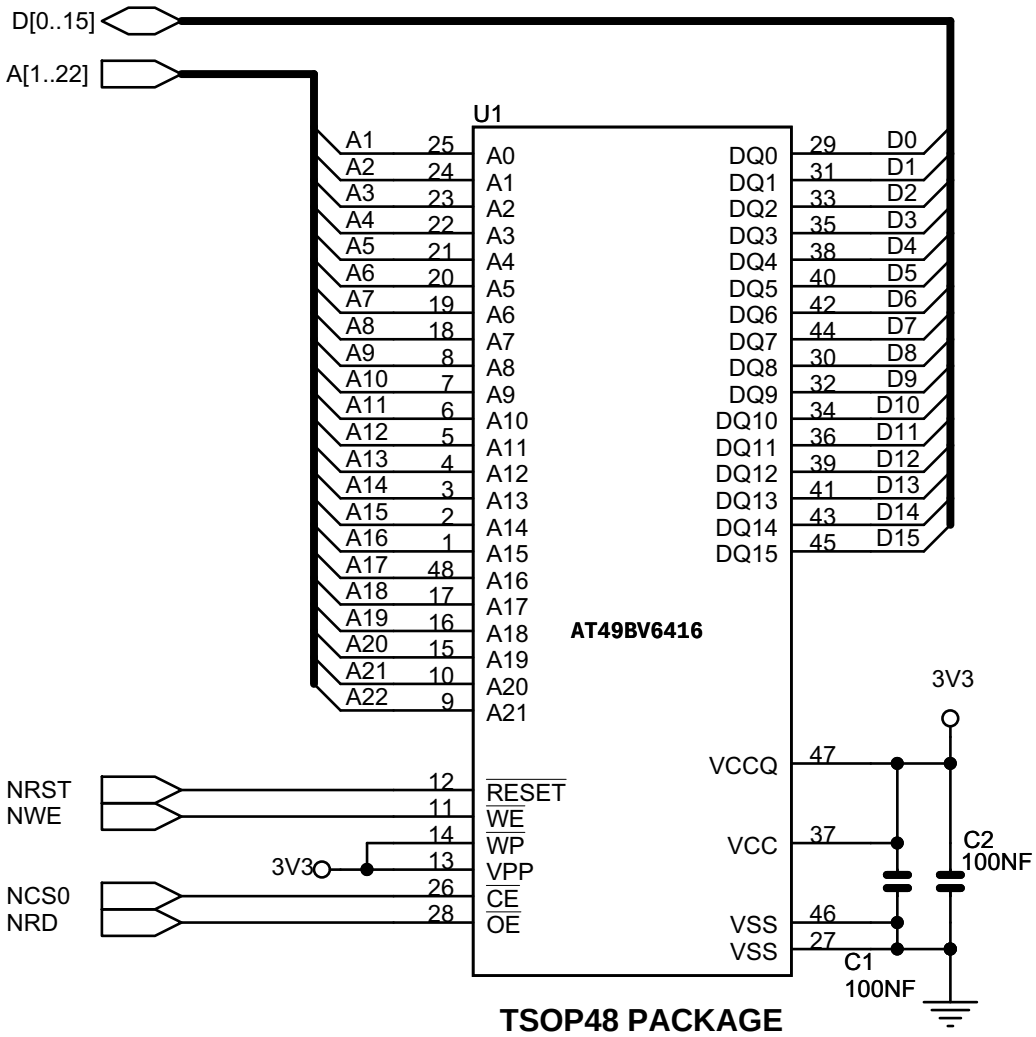
- **P0–P31: Glitch or Debouncing Filter Selection Status**

0: The glitch filter is able to filter glitches with a duration < $t_{peripheral\ clock}/2$.

1: The debouncing filter is able to filter pulses with a duration < $t_{div\_slck}/2$.

28.2.2.3 NOR Flash on NCS0

Hardware Configuration



Software Configuration

The default configuration for the Static Memory Controller, byte select mode, 16-bit data bus, Read/Write controlled by Chip Select, allows boot on 16-bit non-volatile memory at slow clock.

For another configuration, configure the Static Memory Controller CS0 Setup, Pulse, Cycle and Mode depending on Flash timings and system bus frequency.

30.13.2 TDF Optimization Enabled (TDF_MODE = 1)

When the TDF_MODE of the HSMC_MODE register is set to 1 (TDF optimization is enabled), the SMC takes advantage of the setup period of the next access to optimize the number of wait states cycle to insert.

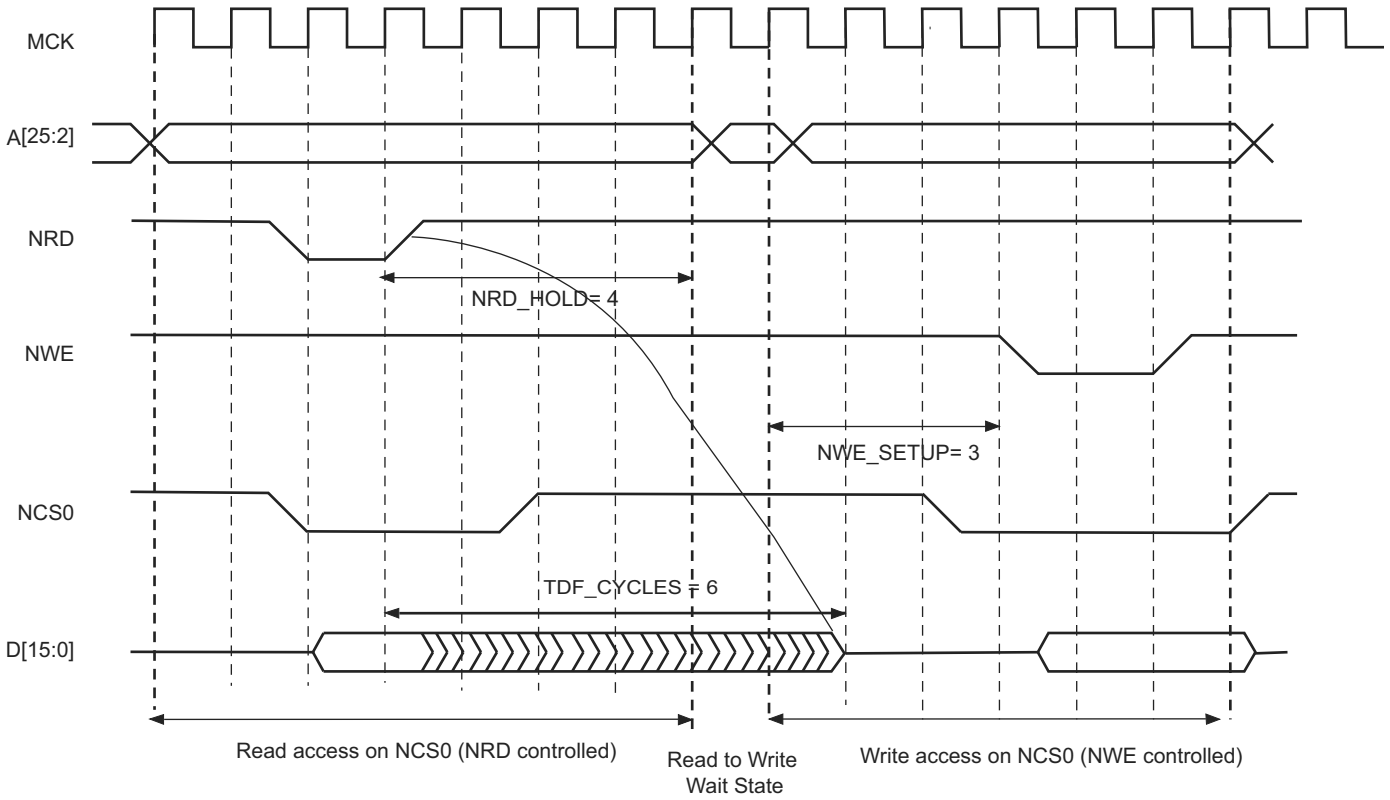
Figure 30-19 shows a read access controlled by NRD, followed by a write access controlled by NWE, on Chip Select 0. Chip Select 0 has been programmed with:

NRD_HOLD = 4; READ_MODE = 1 (NRD controlled)

NWE_SETUP = 3; WRITE_MODE = 1 (NWE controlled)

TDF_CYCLES = 6; TDF_MODE = 1 (optimization enabled).

Figure 30-19. TDF Optimization: No TDF wait states are inserted if the TDF period is over when the next access begins



30.13.3 TDF Optimization Disabled (TDF_MODE = 0)

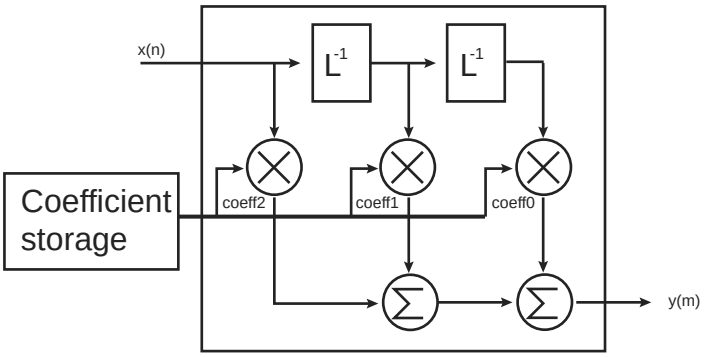
When optimization is disabled, TDF wait states are inserted at the end of the read transfer, so that the data float period ends when the second access begins. If the hold period of the read1 controlling signal overlaps the data float period, no additional TDF wait states will be inserted.

Figure 30-20, Figure 30-21 and Figure 30-22 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.

Figure 32-8. Vertical Resampler Filter Architecture



32.6.9.2 Horizontal Scaler

The XMEMSIZE field of the LCDC_HEOCFG4 register indicates the horizontal size minus one of the image in the system memory. The XSIZE field of the LCDC_HEOCFG3 register contains the horizontal size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to one. The scaling factor is programmed in the XFACTOR field of the LCDC_HEOCFG13 register. Use the following algorithm to find the XFACTOR value:

$$XFACTOR_{1st} = floor\left(\frac{8 \times 256 \times XMEMSIZE - 256 \times XPHIDEF}{XSIZE}\right)$$

$$XFACTOR_{1st} = XFACTOR_{1st} + 1$$

$$XMEMSIZE_{max} = floor\left(\frac{XFACTOR_{1st} \times XSIZE + 256 \times XPHIDEF}{2048}\right)$$

$$\begin{cases} XFACTOR = XFACTOR_{1st} - 1 & when(XMEMSIZE_{max} > XMEMSIZE) \\ XFACTOR = XFACTOR_{1st} & otherwise \end{cases}$$

32.6.9.3 Vertical Scaler

The YMEMSIZE field of the LCDC_HEOCFG4 register indicates the vertical size minus one of the image in the system memory. The YSIZE field of the LCDC_HEOCFG3 register contains the vertical size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to one. The scaling factor is programmed in the YFACTOR field of the LCDC_HEOCFG13 register.

$$YFACTOR_{1st} = floor\left(\frac{8 \times 256 \times YMEMSIZE - 256 \times YPHIDEF}{YSIZE}\right)$$

$$YFACTOR_{1st} = YFACTOR_{1st} + 1$$

$$YMEMSIZE_{max} = floor\left(\frac{YFACTOR_{1st} \times YSIZE + 256 \times YPHIDEF}{2048}\right)$$

$$\begin{cases} YFACTOR = YFACTOR_{1st} - 1 & when(YMEMSIZE_{max} > YMEMSIZE) \\ YFACTOR = YFACTOR_{1st} & otherwise \end{cases}$$

32.6.10 Hardware Cursor

The LCD module integrates a hardware cursor database. This layer features only a minimal set of color among 1, 2, 4 and 8 bpp palletized and 16 bpp to 32 bpp true color. The cursor size is limited to 128 x 128 pixels.

35.7.4 UHPHS USB Command Register

Name: UHPHS_USBCMD

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
ITC							
15	14	13	12	11	10	9	8
–				ASPME	–	ASPMC	
7	6	5	4	3	2	1	0
LHCR	IAAD	ASE	PSE	FLS		HCRESET	RS

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed.

- **RS: Run/Stop (read/write)**

0: Stop (default value).

1: Run.

When set to 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to 1. When this bit is set to 0, the Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 micro-frames after software clears the Run bit. The HC Halted bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write 1 to this field unless the host controller is in the Halted state (i.e., HCHalted in the UHPHS_USBSTS register is 1). Doing so will yield undefined results.

- **HCRESET: Host Controller Reset (read/write)**

This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset.

When software writes a 1 to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports.

PCI Configuration registers are not affected by this reset. All operational registers, including port registers and port state machines, are set to their initial values. Port ownership reverts to the companion host controller(s) with side effects. Software must reinitialize the host controller in order to return the host controller to an operational state.

This bit is set to 0 by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a 0 to this register.

Software should not set this bit to 1 when the HCHalted bit in the UHPHS_USBSTS register is 0. Attempting to reset an actively running host controller will result in undefined behavior.

Table 36-5. Transmit Buffer Descriptor Entry

Bit	Function
Word 0	
31:0	Byte address of buffer
Word 1	
31	Used—must be zero for the GMAC to read data to the transmit buffer. The GMAC sets this to one for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again.
30	Wrap—marks last descriptor in transmit buffer descriptor list. This can be set for any buffer within the frame.
29	Retry limit exceeded, transmit error detected
28	Reserved.
27	Transmit frame corruption due to AHB error—set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted). Also set if single frame is too large for configured packet buffer memory size.
26	Late collision, transmit error detected. Late collisions only force this status bit to be set in gigabit mode.
25:23	Reserved
22:20	Transmit IP/TCP/UDP checksum generation offload errors: 000: No Error. 001: The Packet was identified as a VLAN type, but the header was not fully complete, or had an error in it. 010: The Packet was identified as a SNAP type, but the header was not fully complete, or had an error in it. 011: The Packet was not of an IP type, or the IP packet was invalidly short, or the IP was not of type IPv4/IPv6. 100: The Packet was not identified as VLAN, SNAP or IP. 101: Non supported packet fragmentation occurred. For IPv4 packets, the IP checksum was generated and inserted. 110: Packet type detected was not TCP or UDP. TCP/UDP checksum was therefore not generated. For IPv4 packets, the IP checksum was generated and inserted. 111: A premature end of packet was detected and the TCP/UDP checksum could not be generated.
19:17	Reserved
16	No CRC to be appended by MAC. When set, this implies that the data in the buffers already contains a valid CRC, hence no CRC or padding is to be appended to the current frame by the MAC. This control bit must be set for the first buffer in a frame and will be ignored for the subsequent buffers of a frame. Note that this bit must be clear when using the transmit IP/TCP/UDP checksum generation offload, otherwise checksum generation and substitution will not occur.
15	Last buffer, when set this bit will indicate the last buffer in the current frame has been reached.
14	Reserved
13:0	Length of buffer

Table 36-8. Example of Delay Request Frame in 1588 Version 1 Format (Continued)

Frame Segment	Value
Source IP port (Octets 34–35)	—
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–42)	—
Version PTP (Octet 43)	01
Other stuff (Octets 44–73)	—
Control (Octet 74)	01
Other stuff (Octets 75–168)	—

For 1588 version 1 messages, sync and delay request frames are indicated by the GMAC if the frame type field indicates TCP/IP, UDP protocol is indicated, the destination IP address is 224.0.1.129/130/131 or 132, the destination UDP port is 319 and the control field is correct.

The control field is 0x00 for sync frames and 0x01 for delay request frames.

For 1588 version 2 messages, the type of frame is determined by looking at the message type field in the first byte of the PTP frame. Whether a frame is version 1 or version 2 can be determined by looking at the version PTP field in the second byte of both version 1 and version 2 PTP frames.

In version 2 messages sync frames have a message type value of 0x0, delay_req have 0x1, Pdelay_Req have 0x2 and Pdelay_Resp have 0x3.

Table 36-9. Example of Sync Frame in 1588 Version 2 (UDP/IPv4) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	—
UDP (Octet 23)	11
IP stuff (Octets 24–29)	—
IP DA (Octets 30–33)	E0000181
Source IP port (Octets 34–35)	—
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–41)	—
Message type (Octet 42)	00
Version PTP (Octet 43)	02

Table 36-10. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—

36.9.64 GMAC Late Collisions Register

Name: GMAC_LC

Address: 0xF0028144

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
–	–	–	–	–	–	LCOL	

7	6	5	4	3	2	1	0
LCOL							

- **LCOL: Late Collisions**

This register counts the number of late collisions occurring after the slot time (512 bits) has expired. In 10/100 mode, late collisions are counted twice i.e., both as a collision and a late collision. In gigabit mode, a late collision causes the transmission to be aborted, thus the single and multi collision registers are not updated.

37.4.2.1 FIFO

The FIFO depths are 128 bytes for receive and 128 bytes for transmit and are a function of the system clock speed, memory latency and network speed.

Data is typically transferred into and out of the FIFOs in bursts of four words. For receive, a bus request is asserted when the FIFO contains four words and has space for 28 more. For transmit, a bus request is generated when there is space for four words, or when there is space for 27 words if the next transfer is to be only one or two words.

Thus the bus latency must be less than the time it takes to load the FIFO and transmit or receive three words (112 bytes) of data.

At 100 Mbit/s, it takes 8960 ns to transmit or receive 112 bytes of data. In addition, six master clock cycles should be allowed for data to be loaded from the bus and to propagate through the FIFOs. For a 133 MHz master clock this takes 45 ns, making the bus latency requirement 8915 ns.

37.4.2.2 Receive Buffers

Received frames, including CRC/FCS optionally, are written to receive buffers stored in memory. Each receive buffer is 128 bytes long. The start location for each receive buffer is stored in memory in a list of receive buffer descriptors at a location pointed to by the Receive Buffer Queue Pointer Register (EMAC_RBQP). The receive buffer start location is a word address. For the first buffer of a frame, the start location can be offset by up to three bytes depending on the value written to bits 14 and 15 of the Network Configuration Register (EMAC_NCFGR). If the start location of the buffer is offset the available length of the first buffer of a frame is reduced by the corresponding number of bytes.

Each list entry consists of two words, the first being the address of the receive buffer and the second being the receive status. If the length of a receive frame exceeds the buffer length, the status word for the used buffer is written with zeroes except for the ‘Start of Frame’ bit and the offset bits, if appropriate. Bit 0 of the address field is written to one to show the buffer has been used. The receive buffer manager then reads the location of the next receive buffer and fills that with receive frame data. The final buffer descriptor status word contains the complete frame status. Refer to Table 37-1 for details of the receive buffer descriptor list.

Table 37-1. Receive Buffer Descriptor Entry

Bit	Function
Word 0	
31:2	Address of beginning of buffer
1	Wrap - marks last descriptor in receive buffer descriptor list.
0	Ownership - needs to be zero for the EMAC to write data to the receive buffer. The EMAC sets this to one once it has successfully written a frame to memory. Software has to clear this bit before the buffer can be used again.
Word 1	
31	Global all ones broadcast address detected
30	Multicast hash match
29	Unicast hash match
28	External address match
27	Reserved for future use
26	Specific address register 1 match
25	Specific address register 2 match
24	Specific address register 3 match

37.6.27.16 Excessive Length Errors Register

Name: EMAC_ELE

Address: 0xF802C078

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
EXL							

- **EXL: Excessive Length Errors**

An 8-bit register counting the number of frames received exceeding 1518 bytes (1536 if bit 8 set in EMAC_NCFGR) in length but do not have either a CRC error, an alignment error nor a receive symbol error.

38.8.6 READ_SINGLE_BLOCK Operation using DMA Controller

38.8.6.1 Block Length is Multiple of 4

1. Wait until the current command execution has successfully completed.
    1. Check that CMDRDY and NOTBUSY are asserted in HSMCI_SR.
2. Program the block length in the card. This value defines the value *block_length.*
3. Program the block length in the HSMCI Configuration Register with *block_length* value.
4. Set RDPROOF bit in HSMCI_MR to avoid overflow.
5. Configure the fields of the HSMCI_DMA register as follows:
    1. ROPT bit is configured to 0.
    2. OFFSET field is configured to 0.
    3. CHKSIZE is user defined.
    4. DMAEN is set to true to enable DMAC hardware handshaking in the HSMCI. This bit was previously set to false.
6. Issue a READ_SINGLE_BLOCK command.
7. Program the DMA controller.
    1. Read the channel register to choose an available (disabled) channel.
    2. Clear any pending interrupts on the channel from the previous DMA transfer by reading the DMAC_EBCISR.
    3. Program the channel registers.
    4. The DMAC_SADDRx for Channel x must be set with the starting address of the HSMCI_FIFO address.
    5. The DMAC_DADDRx for Channel x must be word aligned.
    6. Configure the fields of the DMAC_CTRLAx register for Channel x as follows:

    –DST_WIDTH is set to WORD.

    –SRC_WIDTH is set to WORD.

    –SCSIZE must be set according to the value of HSMCI_DMA.CHKSIZE.

    –BTSIZE is programmed with *block_length/4.*

    7. Configure the fields of the DMAC_CFGx register for Channel x 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 DST_DSCR and SRC_DSCR are set (descriptor fetch is disabled).

    –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.

    8. Configure the fields of the DMAC_CFGx register for Channel x as follows:

    –FIFOCFG defines the watermark of the DMA 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.

    –Enable Channel x, writing one to DMAC_CHER[x]. The DMAC is ready and waiting for request.
8. Wait for XFRDONE in the HSMCI_SR.

38.8.6.2 Block Length is Not Multiple of 4 and Padding Not Used (HSMCI_DMA.ROPT = 0)

In the previous DMA transfer flow (block length multiple of 4), the DMA controller is configured to use only WORD AHB access. When the block length is no longer a multiple of 4 this is no longer true. The DMA controller is programmed to copy exactly the block length number of bytes using two transfer descriptors.

38.9 SD/SDIO Card Operation

The High Speed MultiMedia Card Interface allows processing of SD Memory (Secure Digital Memory Card) and SDIO (SD Input Output) Card commands.

SD/SDIO cards are based on the MultiMedia Card (MMC) format, but are physically slightly thicker and feature higher data transfer rates, a lock switch on the side to prevent accidental overwriting and security features. The physical form factor, pin assignment and data transfer protocol are forward-compatible with the High Speed MultiMedia Card with some additions. SD slots can actually be used for more than flash memory cards. Devices that support SDIO can use small devices designed for the SD form factor, such as GPS receivers, Wi-Fi or Bluetooth adapters, modems, barcode readers, IrDA adapters, FM radio tuners, RFID readers, digital cameras and more.

SD/SDIO is covered by numerous patents and trademarks, and licensing is only available through the Secure Digital Card Association.

The SD/SDIO Card communication is based on a 9-pin interface (Clock, Command, 4 x Data and 3 x Power lines). The communication protocol is defined as a part of this specification. The main difference between the SD/SDIO Card and the High Speed MultiMedia Card is the initialization process.

The SD/SDIO Card Register (HSMCI_SDCR) allows selection of the Card Slot and the data bus width.

The SD/SDIO Card bus allows dynamic configuration of the number of data lines. After power up, by default, the SD/SDIO Card uses only DAT0 for data transfer. After initialization, the host can change the bus width (number of active data lines).

38.9.1 SDIO Data Transfer Type

SDIO cards may transfer data in either a multi-byte (1 to 512 bytes) or an optional block format (1 to 511 blocks), while the SD memory cards are fixed in the block transfer mode. The TRTYP field in the HSMCI Command Register (HSMCI_CMDR) allows to choose between SDIO Byte or SDIO Block transfer.

The number of bytes/blocks to transfer is set through the BCNT field in the HSMCI Block Register (HSMCI_BLKR). In SDIO Block mode, the field BLKLEN must be set to the data block size while this field is not used in SDIO Byte mode.

An SDIO Card can have multiple I/O or combined I/O and memory (called Combo Card). Within a multi-function SDIO or a Combo card, there are multiple devices (I/O and memory) that share access to the SD bus. In order to allow the sharing of access to the host among multiple devices, SDIO and combo cards can implement the optional concept of suspend/resume (Refer to the SDIO Specification for more details). To send a suspend or a resume command, the host must set the SDIO Special Command field (IOSPCMD) in the HSMCI Command Register.

38.9.2 SDIO Interrupts

Each function within an SDIO or Combo card may implement interrupts (Refer to the SDIO Specification for more details). In order to allow the SDIO card to interrupt the host, an interrupt function is added to a pin on the DAT[1] line to signal the card's interrupt to the host. An SDIO interrupt on each slot can be enabled through the HSMCI Interrupt Enable Register. The SDIO interrupt is sampled regardless of the currently selected slot.

38.14.1 HSMCI Control Register

Name: HSMCI_CR

Address: 0xF0000000 (0), 0xF8000000 (1), 0xF8004000 (2)

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
SWRST	–	–	–	PWSDIS	PWSEN	MCIDIS	MCIEN

- **MCIEN: Multi-Media Interface Enable**

0: No effect.

1: Enables the Multi-Media Interface if MCDIS is 0.

- **MCIDIS: Multi-Media Interface Disable**

0: No effect.

1: Disables the Multi-Media Interface.

- **PWSEN: Power Save Mode Enable**

0: No effect.

1: Enables the Power Saving Mode if PWSDIS is 0.

Warning: Before enabling this mode, the user must set a value different from 0 in the PWSDIV field of the HSMCI_MR.

- **PWSDIS: Power Save Mode Disable**

0: No effect.

1: Disables the Power Saving Mode.

- **SWRST: Software Reset**

0: No effect.

1: Resets the HSMCI. A software triggered hardware reset of the HSMCI is performed.

38.14.8 HSMCI Completion Signal Timeout Register

Name: HSMCI_CSTOR

Address: 0xF000001C (0), 0xF800001C (1), 0xF800401C (2)

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
–	CSTOMUL			CSTOCYC			

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

- **CSTOCYC: Completion Signal Timeout Cycle Number**

This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. Its value is calculated by (CSTOCYC x Multiplier).

- **CSTOMUL: Completion Signal Timeout Multiplier**

This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. Its value is calculated by (CSTOCYC x Multiplier).

These fields determine the maximum number of Master Clock cycles that the HSMCI waits between the end of the data transfer and the assertion of the completion signal. The data transfer comprises data phase and the optional busy phase. If a non-DATA ATA command is issued, the HSMCI starts waiting immediately after the end of the response until the completion signal.

Multiplier is defined by CSTOMUL as shown in the following table:

Value	Name	Description
0	1	CSTOCYC x 1
1	16	CSTOCYC x 16
2	128	CSTOCYC x 128
3	256	CSTOCYC x 256
4	1024	CSTOCYC x 1024
5	4096	CSTOCYC x 4096
6	65536	CSTOCYC x 65536
7	1048576	CSTOCYC x 1048576

If the data time-out set by CSTOCYC and CSTOMUL has been exceeded, the Completion Signal Time-out Error flag (CSTOE) in the HSMCI Status Register (HSMCI_SR) rises.

41.9.3 SSC Receive Clock Mode Register

Name: SSC_RCMR

Address: 0xF0008010 (0), 0xF800C010 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
PERIOD							
23	22	21	20	19	18	17	16
STTDLY							
15	14	13	12	11	10	9	8
–	–	–	STOP	START			
7	6	5	4	3	2	1	0
CKG		CKI	CKO			CKS	

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

- **CKS: Receive Clock Selection**

Value	Name	Description
0	MCK	Divided Clock
1	TK	TK Clock signal
2	RK	RK pin

- **CKO: Receive Clock Output Mode Selection**

Value	Name	Description
0	NONE	None, RK pin is an input
1	CONTINUOUS	Continuous Receive Clock, RK pin is an output
2	TRANSFER	Receive Clock only during data transfers, RK pin is an output

- **CKI: Receive Clock Inversion**

0: The data inputs (Data and Frame Sync signals) are sampled on Receive Clock falling edge. The Frame Sync signal output is shifted out on Receive Clock rising edge.

1: The data inputs (Data and Frame Sync signals) are sampled on Receive Clock rising edge. The Frame Sync signal output is shifted out on Receive Clock falling edge.

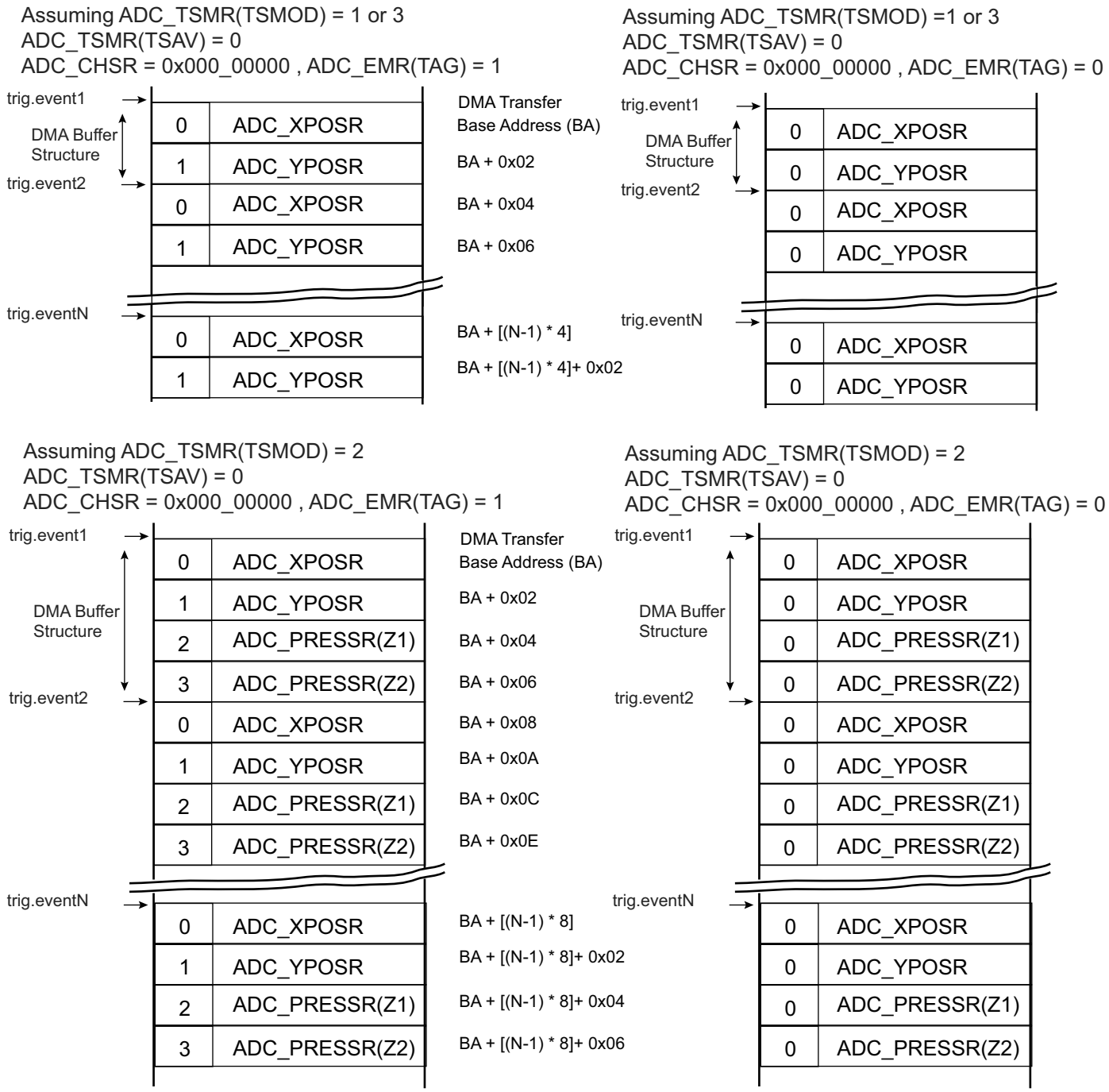
CKI affects only the Receive Clock and not the output clock signal.

- **CKG: Receive Clock Gating Selection**

Value	Name	Description
0	CONTINUOUS	None
1	EN_RF_LOW	Receive Clock enabled only if RF Low
2	EN_RF_HIGH	Receive Clock enabled only if RF High

There is no change in buffer structure whatever the value of PENDET bit configuration in ADC_TSMR but it is recommended to use the pen detection function for buffer post-processing (refer to Section 49.6.14.4 “Pen Detection Status”).

Figure 49-16. Buffer Structure When Only Touchscreen Channels are Enabled



49.6.14.3Interleaved Channels

When both classic ADC channels (CH4/CH5 up to CH12 are set in ADC_CHSR) and touchscreen conversions are required (TSMODE ≠ 0 in ADC_TSMR) the structure of the buffer differs according to TSAV and TSFREQ values.

Table 60-1. SAMA5D3 Datasheet Rev. 11121F Revision History (Continued)

Date	Comments
02-Feb-16	Section 17. “Watchdog Timer (WDT)” Section 17.1 “Description”: in last sentence, “debug mode or idle mode” changed to “Debug mode or Sleep mode (Idle mode)” Section 17.4 “Functional Description”: - inserted sentence “When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.” - in last sentence, “debug state or in idle mode” changed to “debug state or in Sleep mode” Section 17.5.1 “Watchdog Timer Control Register”: added note on modification of WDT_CR values Section 17.5.2 “Watchdog Timer Mode Register”: updated note on modification of WDT_MR values; added note to WDDIS bit description
	Section 19. “Shutdown Controller (SHDWC)” Updated Section 19.2 “Embedded Characteristics”
	Section 20. “General Purpose Backup Registers (GPBR)” Updated Section 20.1 “Description” and Section 20.2 “Embedded Characteristics”
	Section 22. “Real-time Clock (RTC)” Section 22.1 “Description”: in second paragraph, “a two-hundred-year Gregorian calendar” changed to “a Gregorian calendar” Updated Section 22.2 “Embedded Characteristics” Section 22.5 “Functional Description”: “The valid year range is 1900 to 2099 in Gregorian mode, a two-hundred-year calendar” changed to “The valid year range is up to 2099 in Gregorian mode” Table 22-2 ”Register Mapping”: added offset 0xCC as reserved Section 22.6.1 “RTC Control Register”: updated descriptions of bits UPDTIM, UPDCAL and CALEVSEL Section 22.6.3 “RTC Time Register”: deleted sentence “All non-significant bits read zero.” Section 22.6.4 “RTC Calendar Register”: updated CENT field decription; deleted sentence “All non-significant bits read zero.”
	Section 23. “Slow Clock Controller (SCKC)” Updated Section 23.1 “Description” Updated Section 23.4 “Functional Description”
	Section 25. “Clock Generator” Updated Section 25.2 “Embedded Characteristics” Figure 25-1 “Clock Generator Block Diagram”: updated oscillator descriptions Figure 25-3 “Main Clock Block Diagram”: removed “MOSCRCF” block Removed section “Main Clock Selection” Added Section 25.4 “Slow Clock” Revised Section 25.5 “Main Clock” Section 25.7 “UTMI Phase Lock Loop Programming”: updated first and second paragraphs
	Section 26. “Power Management Controller (PMC)” Section 26.2 “Embedded Characteristics”: in Processor Clock bullet, “when entering the processor in Sleep Mode” changed to “when processor is entering Idle mode” Figure 26-1 “General Clock Block Diagram”: added signal from PMC_MCKR to DDRCK Section 26.4 “Master Clock Controller”: inserted note relative to modification of fields MDIV and CSS Updated Section 26.5 “Processor Clock Controller” Revised Section 26.6 “LCDC Clock Controller”