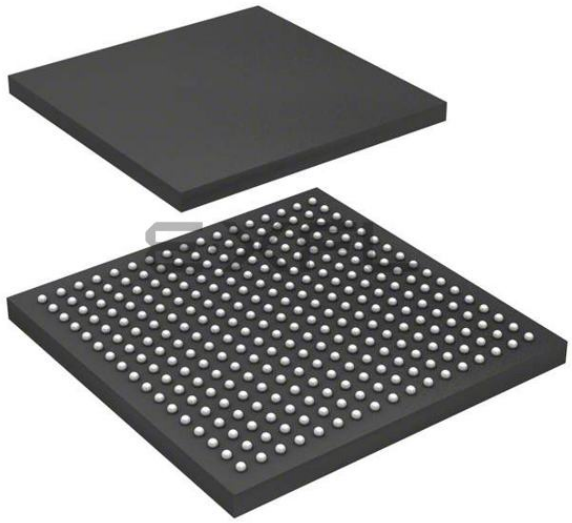


**Welcome to E-XFL.COM**

## 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 ~ 105°C (TA) |
| Security Features | ARM TZ, Boot Security, Cryptography, RTIC, Secure Fusebox, Secure JTAG, Secure Memory, Secure RTC |
| Package / Case | 289-LFBGA |
| Supplier Device Package | 289-LFBGA (14x14) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/atsama5d28a-cnr |

### 14.5.23 L2CC Clean Invalidate Index Register

**Name:** L2CC_CIIR

**Address:** 0x00A007F8

**Access:** Read/Write

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| – | WAY | | | – | – | – | – |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | IDX | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IDX | | | – | – | – | – | C |

**C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

**IDX: Index Number**

**WAY: Way Number**

### 19.3.4 Security Configuration Register

**Name:** SFR_SECURE

**Address:** 0xF8030028

**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 |
| – | – | – | – | – | – | – | FUSE |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| – | – | – | – | – | – | – | ROM |

**ROM: Disable Access to ROM Code**

This bit is writable once only. When the ROM is secured, only a reset signal can clear this bit.

0: ROM is enabled.

1: ROM is disabled.

**FUSE: Disable Access to Fuse Controller**

This bit is writable once only. When the Fuse Controller is secured, only a reset signal can clear this bit.

0: Fuse Controller is enabled.

1: Fuse Controller is disabled.

#### 26.5.6.2 UTC Mode

The update of the UTC time field must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER.

Once the second event occurs, the user must stop the RTC by setting the UPDTIM field in the Control Register (RTC_CR).

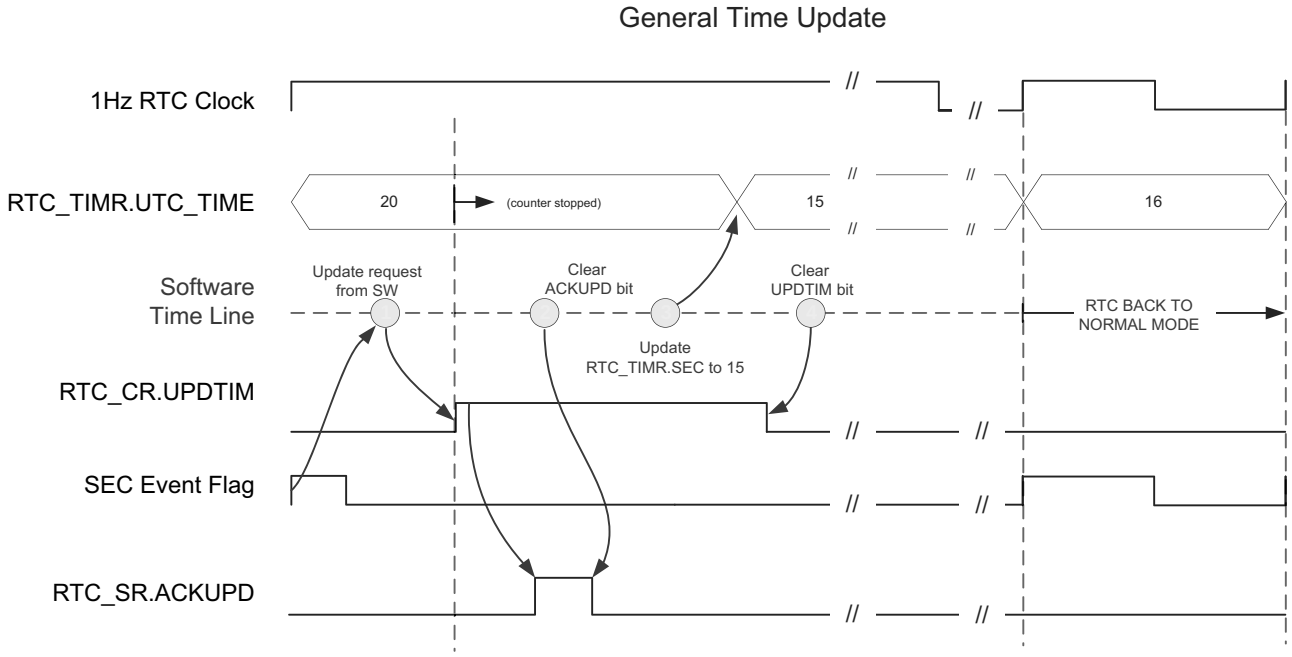
The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register.

Once the update is finished, the user must write UPDTIM to 0 in the RTC_CR.

The timing sequence of the UTC time update is described in Figure 26-4 ”UTC Time Update Timing Diagram”.

In successive update operations, the user must wait for at least one second after resetting the UPDTIM bit in the RTC_CR before setting this bit again. This is done by waiting for the SEC flag in the RTC_SR before setting UPDTIM bit. After resetting UPDTIM, the SEC flag must also be cleared.

**Figure 26-4: UTC Time Update Timing Diagram**



In order to calibrate the 12 MHz oscillator frequency, SEL must be set to 1 and a correct frequency value must be configured in the CAL field.

It is possible to restart, at anytime, a measurement of the frequency of the selected clock by means of the RCMEAS bit in the Clock Generator Main Clock Frequency register (CKGR_MCFR). Thus, when MAINFRDY flag reads 1, another read access on CKGR_MCFR provides an image of the frequency of the Main clock on MAINF field. The software can calculate the error with an expected frequency and correct PMC_OCR.CAL accordingly. This may be used to compensate frequency drift due to derating factors such as temperature and/or voltage.

### 32.5.3 8 to 24 MHz Crystal Oscillator

After reset, the 8 to 24 MHz crystal oscillator is disabled and is not selected as the source of MAINCK.

As the source of MAINCK, the 8 to 24 MHz crystal oscillator provides an accurate frequency. The software enables or disables this oscillator in order to reduce power consumption via CKGR_MOR.MOSCXTEN.

When disabling this oscillator by clearing the CKGR_MOR.MOSCXTEN bit, the PMC_SR.MOSCXTS bit is automatically cleared, indicating the 8 to 24 MHz crystal oscillator is off.

When enabling this oscillator, the user must initiate the startup time counter. This startup time depends on the characteristics of the external device connected to this oscillator. Refer to Section 66. “Electrical Characteristics” for the startup time.

When CKGR_MOR.MOSCXTEN and CKGR_MOR.MOSCXTST are written to enable this oscillator, the PMC_SR.MOSCXTS bit is cleared and the counter starts counting down on the Slow clock divided by 8 from the MOSCXTST value. When the counter reaches 0, the PMC_SR.MOSCXTS is set, indicating that the 8 to 24 MHz crystal oscillator is stabilized. Setting MOSCXTS in the PMC Interrupt Mask register (PMC_IMR) triggers an interrupt to the processor.

### 32.5.4 Main Clock Source Selection

The source of the Main clock can be selected from the following:

- embedded 12 MHz RC oscillator
- 8 to 24 MHz crystal oscillator
- an XRCGB crystal resonator

The advantage of the Main RC oscillator is its fast startup time. By default, this oscillator is selected to start the system and it must be selected prior to entering Wait mode.

The advantage of the Main crystal oscillator is its high level of accuracy.

The selection is made by writing CKGR_MOR.MOSCSEL. The switch of the Main clock source is glitch-free, so there is no need to run out of SLCK or PLLACK in order to change the selection. PMC_SR.MOSCSELS indicates when the switch sequence is done.

Setting PMC_IMR.MOSCSELS triggers an interrupt to the processor.

The 8 to 24 MHz crystal oscillator can be bypassed by setting the CKGR_MOR.MOSCXTBY to accept an external Main clock on XIN (refer to Section 32.5.5 “Bypassing the 8 to 24 MHz Crystal Oscillator”).

MOSCRCEN, MOSCSEL, MOSCXTEN and MOSCXTBY bits are located in the PMC Clock Generator Main Oscillator Register (CKGR_MOR).

After a VDDBU power-on reset, the default configuration is MOSCRCEN = 1, MOSCXTEN = 0 and MOSCSEL = 0, allowing the 12 MHz RC oscillator to start as Main clock.

It takes two 32 kHz (typical) clock cycles to detect and switch from the 8 to 24 MHz crystal oscillator to the 12 MHz RC oscillator if the source Master clock (MCK) is Main clock (MAINCK), or three 32 kHz (typical) cycles if the source of MCK is PLLACK or UPLLCK.

A clock failure detection activates a fault output that is connected to the Pulse Width Modulator (PWM) Controller. With this connection, the PWM controller is able to force its outputs and to protect the driven device, if a clock failure is detected.

The user can know the status of the clock failure detector at any time by reading bit PMC_SR.FOS.

This fault output remains active until the defect is detected and until it is cleared by the bit FOCLR in the PMC Fault Output Clear register (PMC_FOCR).

## 33.18 32.768 kHz Crystal Oscillator Frequency Monitor

The frequency of the 32.768 kHz crystal oscillator can be monitored by means of logic driven by the 12 MHz RC oscillator known as a reliable clock source. This function is enabled by configuring the XT32KFME bit of CKGR_MOR.

The error flag XT32KERR in PMC_SR is asserted when the 32.768 kHz crystal oscillator frequency is out of the ±10% nominal frequency value (i.e., 32.768 kHz). The error flag can be cleared only if the Slow clock frequency monitoring is disabled.

The monitored clock frequency is declared invalid if at least four consecutive clock period measurement results are over the nominal period ±10%.

Due to the possible frequency variation of the embedded 12 MHz RC oscillator acting as reference clock for the monitor logic, any Slow clock crystal frequency deviation over ±10% of the nominal frequency is systematically reported as an error by means of PMC_SR.XT32KERR. Between -1% and -10% and +1% and +10%, the error is not systematically reported.

Thus, only a crystal running at a 32.768 kHz frequency ensures that the error flag is not asserted. The permitted drift of the crystal is 10000 ppm (1%), which allows any standard crystal to be used.

The error flag can be defined as an interrupt source of the PMC by setting PMC_IER.XT32KERR.

## 33.19 Programming Sequence

**Note 1:** If the 8 to 24 MHz crystal oscillator is not required, PLL can be directly configured (begin with Step 9. or Step 10.) else this oscillator must be started (begin with Step 5.).

5. Enable the 8 to 24 MHz crystal oscillator by setting CKGR_MOR.MOSCXTEN. The user can define a startup time. This can be achieved by writing a value in CKGR_MOR.MOSCXTST. Once this register has been correctly configured, the user must wait for PMC_SR.MOSCXTS to be set. This can be done either by polling MOSCXTS or by waiting for the interrupt line to be raised if the associated interrupt source (MOSCXTS) has been enabled in PMC_IER.
6. Switch the MAINCK to the 8 to 24 MHz crystal oscillator by setting CKGR_MOR.MOSCSEL.
7. Wait for PMC_SR.MOSCSELS to be set to ensure the switchover is complete.
8. Check the Main clock frequency:

   The Main clock frequency can be measured via CKGR_MCFR.

   Read CKGR_MCFR until the MAINFRDY field is set, after which the user can read the field CKGR_MCFR.MAINF by performing an additional read. This provides the number of Main clock cycles that have been counted during a period of 16 Slow clock cycles.

   If MAINF = 0, switch the MAINCK to the 12 MHz RC oscillator by clearing CKGR_MOR.MOSCSEL. If MAINF ≠ 0, proceed to Step 9.
9. Set the PLLA and divider (if not required, proceed to Step 10.)

   All parameters needed to configure PLLA and the divider are located in CKGR_PLLAR.

   The MULA field is the PLLA multiplier factor. This parameter can be programmed between 0 and 127. If MULA is cleared, PLLA is turned off, otherwise the PLLA output frequency is PLLA input frequency multiplied by (MULA + 1).

   The PLLACOUNT field specifies the number of Slow clock cycles before LOCKA bit is set in PMC_SR after CKGR_PLLAR has been written.

   Once CKGR_PLLAR has been written, the user must wait for the LOCKA bit to be set in PMC_SR. This can be done either by polling LOCKA in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKA) has been enabled in PMC_IER. All parameters in CKGR_PLLAR can be programmed in a single write operation. If at some stage parameter MULA or DIVA is modified, LOCKA bit goes low to indicate that PLLA is not yet ready. When PLLA is locked, LOCKA is set again.

   The user must wait for the LOCKA bit to be set before using the PLLA output clock.
10. Set Bias and High-speed PLL (UPLL) for UTMI

## 39.7 LCD Controller (LCDC) User Interface

**Table 39-55: Register Mapping**

| Offset | Register | Name | Access | Reset |
|---|---|---|---|---|
| 0x00000000 | LCD Controller Configuration Register 0 | LCDC_LCDCFG0 | Read/ Write | 0x00000000 |
| 0x00000004 | LCD Controller Configuration Register 1 | LCDC_LCDCFG1 | Read/ Write | 0x00000000 |
| 0x00000008 | LCD Controller Configuration Register 2 | LCDC_LCDCFG2 | Read/ Write | 0x00000000 |
| 0x0000000C | LCD Controller Configuration Register 3 | LCDC_LCDCFG3 | Read/ Write | 0x00000000 |
| 0x00000010 | LCD Controller Configuration Register 4 | LCDC_LCDCFG4 | Read/ Write | 0x00000000 |
| 0x00000014 | LCD Controller Configuration Register 5 | LCDC_LCDCFG5 | Read/ Write | 0x00000000 |
| 0x00000018 | LCD Controller Configuration Register 6 | LCDC_LCDCFG6 | Read/ Write | 0x00000000 |
| 0x0000001C | Reserved | – | – | – |
| 0x00000020 | LCD Controller Enable Register | LCDC_LCDEN | Write-only | – |
| 0x00000024 | LCD Controller Disable Register | LCDC_LCDDIS | Write-only | – |
| 0x00000028 | LCD Controller Status Register | LCDC_LCDSR | Read-only | 0x00000000 |
| 0x0000002C | LCD Controller Interrupt Enable Register | LCDC_LCDIER | Write-only | – |
| 0x00000030 | LCD Controller Interrupt Disable Register | LCDC_LCDIDR | Write-only | – |
| 0x00000034 | LCD Controller Interrupt Mask Register | LCDC_LCDIMR | Read-only | 0x00000000 |
| 0x00000038 | LCD Controller Interrupt Status Register | LCDC_LCDISR | Read-only | 0x00000000 |
| 0x0000003C | LCD Controller Attribute Register | LCDC_ATTR | Write-only | – |
| 0x00000040 | Base Layer Channel Enable Register | LCDC_BASECHER | Write-only | – |
| 0x00000044 | Base Layer Channel Disable Register | LCDC_BASECHDR | Write-only | – |
| 0x00000048 | Base Layer Channel Status Register | LCDC_BASECHSR | Read-only | 0x00000000 |
| 0x0000004C | Base Layer Interrupt Enable Register | LCDC_BASEIER | Write-only | – |
| 0x00000050 | Base Layer Interrupt Disabled Register | LCDC_BASEIDR | Write-only | – |
| 0x00000054 | Base Layer Interrupt Mask Register | LCDC_BASEIMR | Read-only | 0x00000000 |
| 0x00000058 | Base Layer Interrupt Status Register | LCDC_BASEISR | Read-only | 0x00000000 |
| 0x0000005C | Base DMA Head Register | LCDC_BASEHEAD | Read/ Write | 0x00000000 |
| 0x00000060 | Base DMA Address Register | LCDC_BASEADDR | Read/ Write | 0x00000000 |
| 0x00000064 | Base DMA Control Register | LCDC_BASECTRL | Read/ Write | 0x00000000 |
| 0x00000068 | Base DMA Next Register | LCDC_BASENEXT | Read/ Write | 0x00000000 |

### 39.7.150 Post Processing Configuration Register 2

**Name:** LCDC_PPCFG2

**Address:** 0xF0000574

**Access:** Read/Write

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| XSTRIDE | | | | | | | |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| XSTRIDE | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| XSTRIDE | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| XSTRIDE | | | | | | | |

**XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

**Table 40-18: Register Mapping (Continued)**

| Offset[1] [2] | Register | Name | Access | Reset |
|---|---|---|---|---|
| 0x14C | Carrier Sense Errors Register | GMAC_CSE | Read-only | 0x0000_0000 |
| 0x150 | Octets Received Low Received Register | GMAC_ORLO | Read-only | 0x0000_0000 |
| 0x154 | Octets Received High Received Register | GMAC_ORHI | Read-only | 0x0000_0000 |
| 0x158 | Frames Received Register | GMAC_FR | Read-only | 0x0000_0000 |
| 0x15C | Broadcast Frames Received Register | GMAC_BCFR | Read-only | 0x0000_0000 |
| 0x160 | Multicast Frames Received Register | GMAC_MFR | Read-only | 0x0000_0000 |
| 0x164 | Pause Frames Received Register | GMAC_PFR | Read-only | 0x0000_0000 |
| 0x168 | 64 Byte Frames Received Register | GMAC_BFR64 | Read-only | 0x0000_0000 |
| 0x16C | 65 to 127 Byte Frames Received Register | GMAC_TBFR127 | Read-only | 0x0000_0000 |
| 0x170 | 128 to 255 Byte Frames Received Register | GMAC_TBFR255 | Read-only | 0x0000_0000 |
| 0x174 | 256 to 511 Byte Frames Received Register | GMAC_TBFR511 | Read-only | 0x0000_0000 |
| 0x178 | 512 to 1023 Byte Frames Received Register | GMAC_TBFR1023 | Read-only | 0x0000_0000 |
| 0x17C | 1024 to 1518 Byte Frames Received Register | GMAC_TBFR1518 | Read-only | 0x0000_0000 |
| 0x180 | 1519 to Maximum Byte Frames Received Register | GMAC_TMXBFR | Read-only | 0x0000_0000 |
| 0x184 | Undersize Frames Received Register | GMAC_UFR | Read-only | 0x0000_0000 |
| 0x188 | Oversize Frames Received Register | GMAC_OFR | Read-only | 0x0000_0000 |
| 0x18C | Jabbers Received Register | GMAC_JR | Read-only | 0x0000_0000 |
| 0x190 | Frame Check Sequence Errors Register | GMAC_FCSE | Read-only | 0x0000_0000 |
| 0x194 | Length Field Frame Errors Register | GMAC_LFFE | Read-only | 0x0000_0000 |
| 0x198 | Receive Symbol Errors Register | GMAC_RSE | Read-only | 0x0000_0000 |
| 0x19C | Alignment Errors Register | GMAC_AE | Read-only | 0x0000_0000 |
| 0x1A0 | Receive Resource Errors Register | GMAC_RRE | Read-only | 0x0000_0000 |
| 0x1A4 | Receive Overrun Register | GMAC_ROE | Read-only | 0x0000_0000 |
| 0x1A8 | IP Header Checksum Errors Register | GMAC_IHCE | Read-only | 0x0000_0000 |
| 0x1AC | TCP Checksum Errors Register | GMAC_TCE | Read-only | 0x0000_0000 |
| 0x1B0 | UDP Checksum Errors Register | GMAC_UCE | Read-only | 0x0000_0000 |
| 0x1B4–0x1B8 | Reserved | – | – | – |
| 0x1BC | 1588 Timer Increment Sub-nanoseconds Register | GMAC_TISUBN | Read/Write | 0x0000_0000 |
| 0x1C0 | 1588 Timer Seconds High Register | GMAC_TSH | Read/Write | 0x0000_0000 |
| 0x1C4–0x1CC | Reserved | – | – | – |
| 0x1D0 | 1588 Timer Seconds Low Register | GMAC_TSL | Read/Write | 0x0000_0000 |
| 0x1D4 | 1588 Timer Nanoseconds Register | GMAC_TN | Read/Write | 0x0000_0000 |
| 0x1D8 | 1588 Timer Adjust Register | GMAC_TA | Write-only | – |
| 0x1DC | 1588 Timer Increment Register | GMAC_TI | Read/Write | 0x0000_0000 |
| 0x1E0 | PTP Event Frame Transmitted Seconds Low Register | GMAC_EFTSL | Read-only | 0x0000_0000 |

**RXRDY_TXKL: Received OUT Data Interrupt Disable**

0: No effect.

1: Disable Received OUT Data Interrupt.

**TX_COMPLT: Transmitted IN Data Complete Interrupt Disable**

0: No effect.

1: Disable Transmitted IN Data Complete Interrupt.

**TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Disable**

0: No effect.

1: Disable TX Packet Ready/Transaction Error Interrupt.

**ERR_FL_ISO: Error Flow Interrupt Disable**

0: No effect.

1: Disable Error Flow ISO Interrupt.

**ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Disable**

0: No effect.

1: Disable Error CRC ISO/Error Number of Transaction Interrupt.

**ERR_FLUSH: bank flush error Interrupt Disable**

0: No effect.

1: Disable Bank Flush Error Interrupt.

**BUSY_BANK: Busy Bank Interrupt Disable**

0: No effect.

1: Disable Busy Bank Interrupt.

**SHRT_PCKT: Short Packet Interrupt Disable**

For OUT endpoints:

0: No effect.

1: Disable Short Packet Interrupt.

For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

### 42.7.7 UHPHS USB Frame Index Register

**Name:** UHPHS_FRINDEX

**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 |
| – | | FI | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FI | | | | | | | |

This register is used by the host controller to index into the periodic frame list. The register updates every 125 µs (once each micro-frame). Bits [N:3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the UHPHS_USBCMD register (see **Section 42.7.4 “UHPHS USB Command Register”**).

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the Halted state as indicated by the HCHalted bit (UHPHS_USBSTS register, **Section 42.7.5 “UHPHS USB Status Register”**). A write to this register while the Run/Stop bit is set to 1 (UHPHS_USBCMD register, **Section 42.7.4 “UHPHS USB Command Register”**) produces undefined results. Writes to this register also affect the SOF value.

**FI: Frame Index**

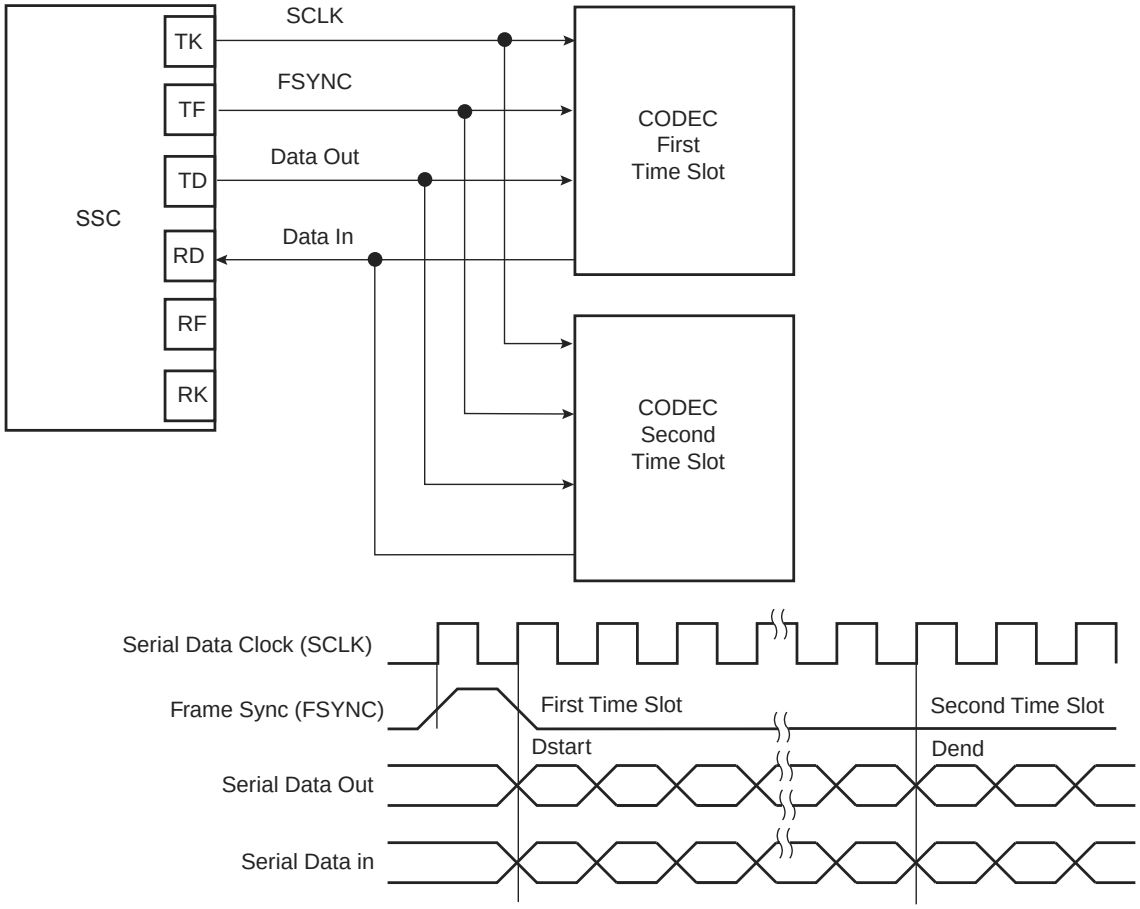
The value in this register increments at the end of each time frame (e.g. micro-frame). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed eight times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the UHPHS_USBCMD register.

| USBCMD [Frame List Size] | Number Elements | N |
|---|---|---|
| 00b | (1024) | 12 |
| 01b | (512) | 11 |
| 10b | (256) | 10 |
| 11b | Reserved | – |

The SOF frame number value for the bus SOF token is derived or alternatively managed from this register. The value of FRINDEX must be 125 µs (1 micro-frame) ahead of the SOF token value. The SOF value may be implemented as an 11-bit shadow register. For this discussion, this shadow register is 11 bits and is named SOFV. SOFV updates every eight micro-frames (1 millisecond). An example implementation to achieve this behavior is to increment SOFV each time the FRINDEX[2:0] increments from 0 to 1.

Software must use the value of FRINDEX to derive the current micro-frame number, both for high-speed isochronous scheduling purposes and to provide the “get micro-frame number” function required for client drivers. Therefore, the value of FRINDEX and the value of SOFV must be kept consistent if chip is reset or software writes to FRINDEX. Writes to FRINDEX must also write-through FRINDEX[13:3] to SOFV[10:0]. In order to keep the update as simple as possible, software should never write a FRINDEX value where the three least significant bits are 111b or 000b.

**Figure 45-5: Time Slot Application Block Diagram**




## 45.6 Pin Name List

**Table 45-1: I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| RF | Receive Frame Synchronization | Input/Output |
| RK | Receive Clock | Input/Output |
| RD | Receive Data | Input |
| TF | Transmit Frame Synchronization | Input/Output |
| TK | Transmit Clock | Input/Output |
| TD | Transmit Data | Output |

**Figure 46-33: User Sends Data While the Bus is Busy**

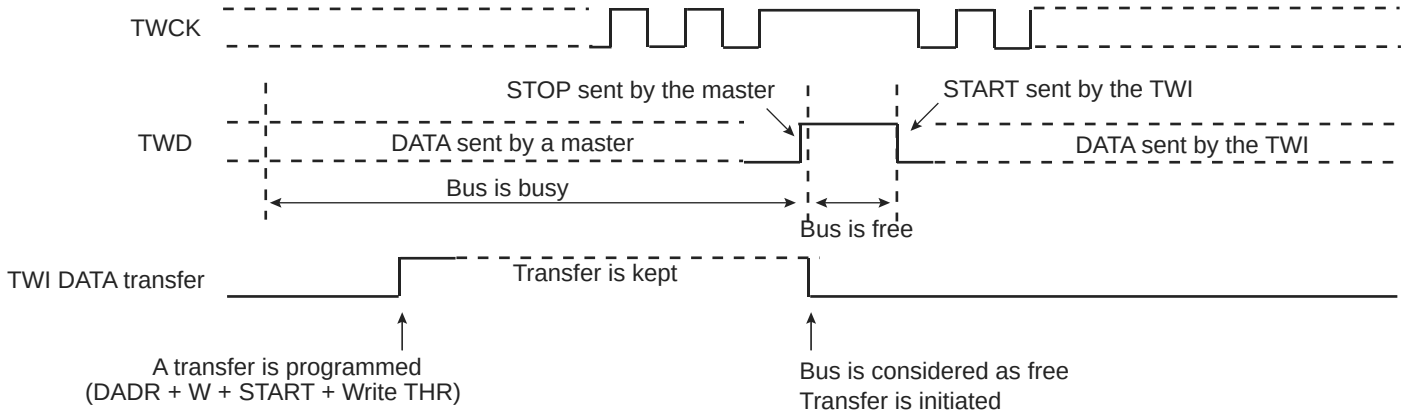
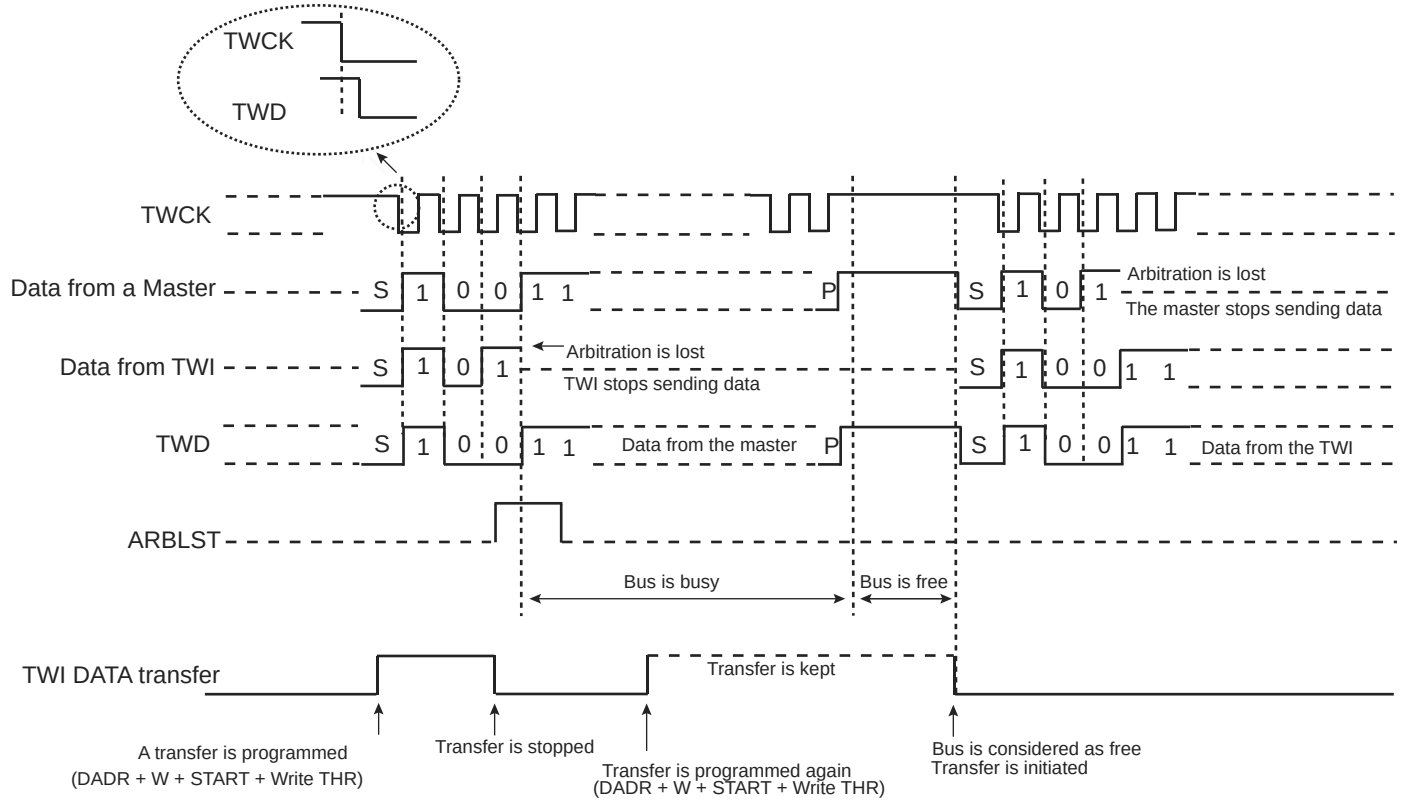



**Figure 46-34: Arbitration Cases**




The flowchart shown in Figure 46-35 gives an example of read and write operations in Multimaster mode.

**Figure 47-48: Master Node Configuration, NACT = SUBSCRIBE**

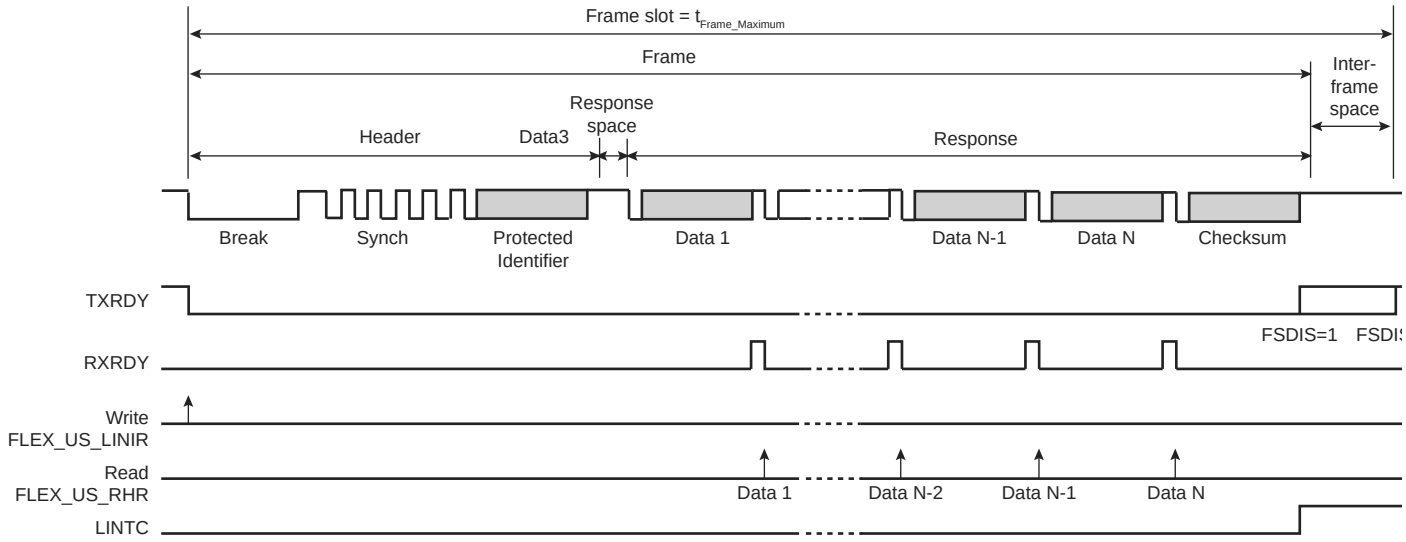
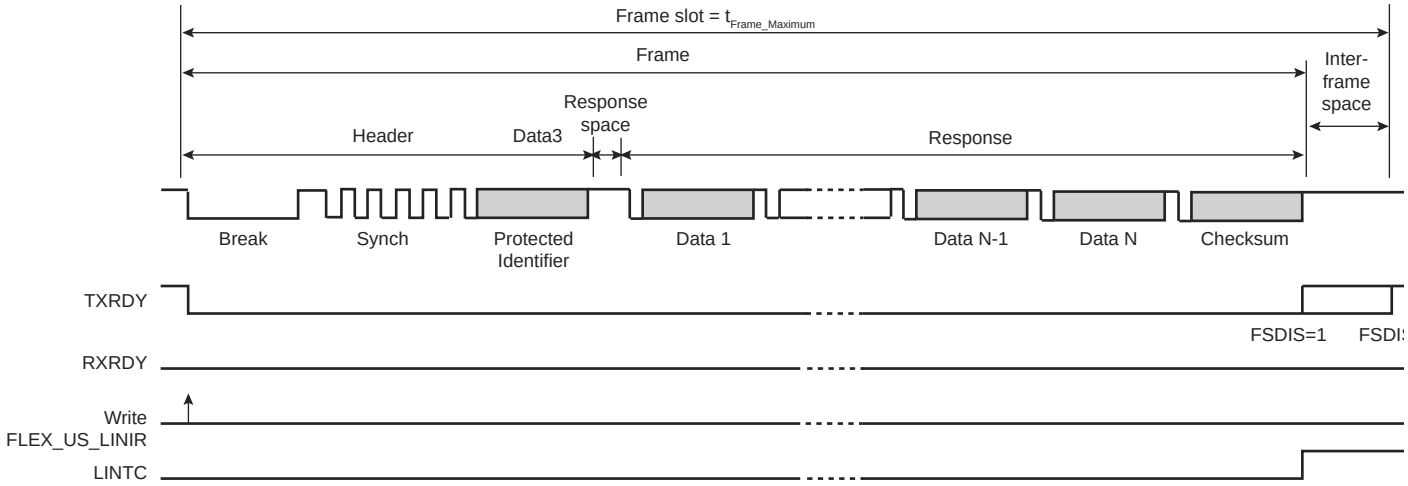



**Figure 47-49: Master Node Configuration, NACT = IGNORE**




- Slave Node Configuration
- Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
- Write FLEX_US_MR.USART_MODE to select the LIN mode and the slave node configuration.
- Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.
- Wait until FLEX_US_CSR.LINID rises.
- Check LINISFE and LINPE errors.
- Read FLEX_US_RHR.IDCHR.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM and DLC in FLEX_US_LINMR to configure the frame transfer.

**IMPORTANT**: If the NACT configuration for this frame is PUBLISH, FLEX_US_LINMR must be written with NACT = PUBLISH even if this field is already correctly configured, in order to set the TXREADY flag and the corresponding write transfer request.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the LIN controller sends the response.
  - Wait until FLEX_US_CSR.TXRDY rises.
  - Write FLEX_US_THR.TCHR to send a byte.
  - If all the data have not been written, repeat the two previous steps.
  - Wait until FLEX_US_CSR. LINTC rises.
  - Check the LIN errors.
- Case 2: NACT = SUBSCRIBE, the USART receives the response.

**Figure 47-63: TXRDY in Single Data Mode and TXRDYM = 0**

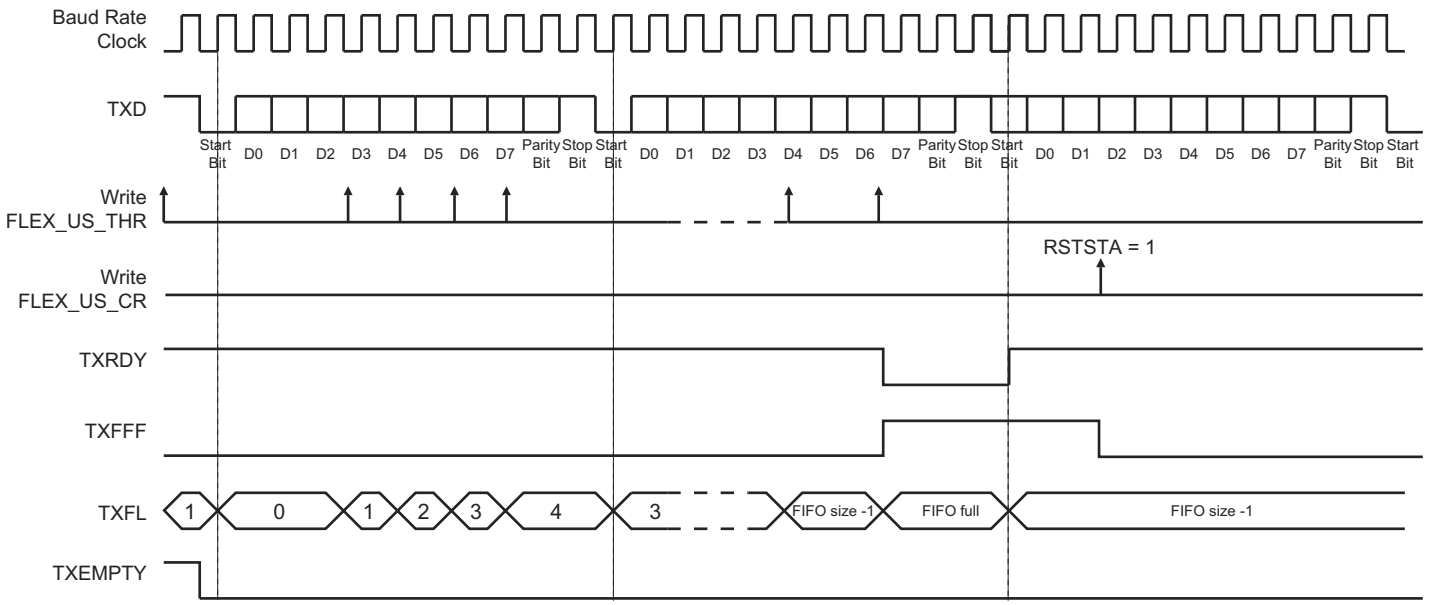
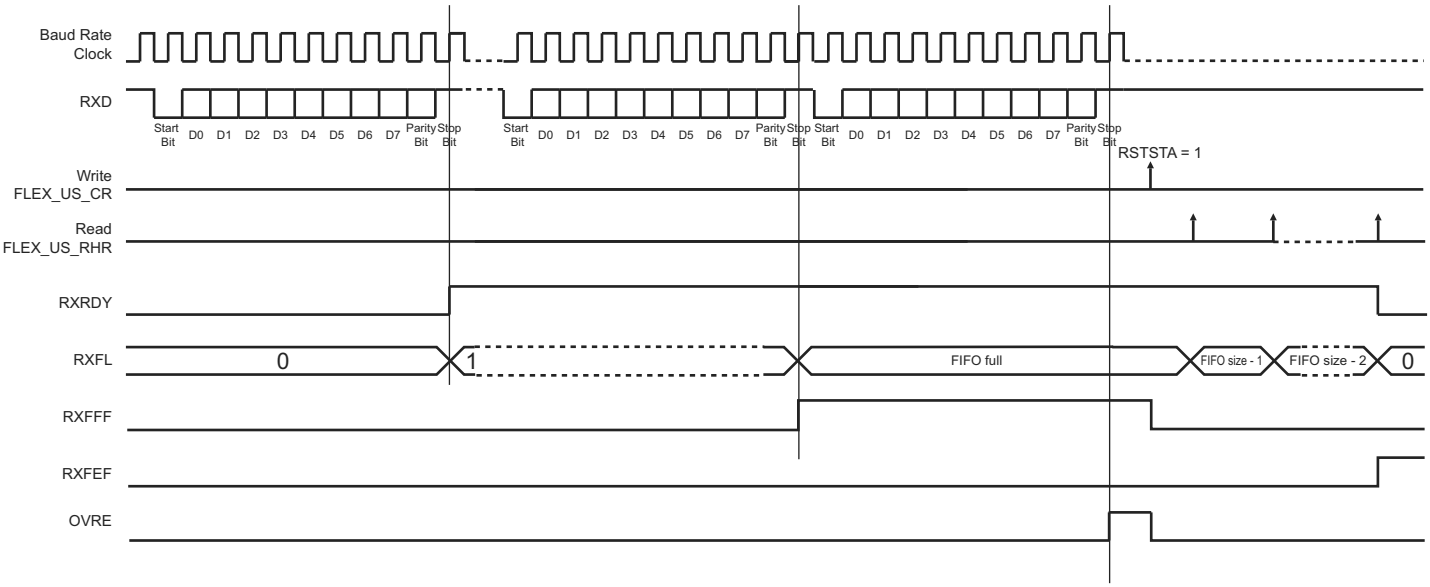



**Figure 47-64: RXRDY in Single Data Mode and RXRDYM = 0**




#### 47.7.11.6 USART Single Data Mode

In Single Data mode, only one data is written every time FLEX_US_THR is accessed, and only one data is read every time FLEX_US_RHR is accessed.

When FLEX_US_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_US_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

If FLEX_US_MR.MODE9 is set, or if FLEX_US_MR.USART_MODE is set to either LIN_MASTER or LIN_SLAVE, the FIFOs must operate in Single Data mode.

See Section 47.10.20 ”USART Receive Holding Register” and Section 47.10.22 ”USART Transmit Holding Register”.

- DMAC

- When a read transfer is stopped at the block gap initiated by a Stop At Block Gap Request (STPBGR).

The SDMMC stops a read operation at the start of the interrupt cycle by driving the Read Wait (DAT[2] line) or by stopping the SD Clock. If the Read Wait signal is already driven (due to the fact that the data buffer cannot receive data), the SDMMC can continue to stop the read operation by driving the Read Wait signal. It is necessary to support the Read Wait in order to use the Suspend/ Resume operation.

In the case of write transactions:

This status indicates that a write transfer is executing on the bus. A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1. Refer to section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for details on timing.

This bit is set in either of the following cases:

- After the end bit of the write command.

- When writing 1 to SDMMC_BGCR.CONTR (Continue Request) to continue a write transfer.

This bit is cleared in either of the following cases:

- When the card releases Write Busy of the last data block. If the card does not drive a Busy signal for 8 SDCLK, the SDMMC considers the card drive “Not Busy”. In the case of ADMA2, the last block is designated by the last transfer of the Descriptor Table.

- When the card releases Write Busy prior to wait for write transfer as a result of a Stop At Block Gap Request (STPBGR).

Command with Busy:

This status indicates whether a command that indicates Busy (ex. erase command for memory) is executing on the bus. This bit is set to 1 after the end bit of the command with Busy and cleared when Busy is deasserted. A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1. Refer to Figures 2.11 to 2.13 in the “SD Host Controller Simplified Specification V3.00” .

0: DAT line inactive.

1: DAT line active.

**WTACT: Write Transfer Active**

This bit indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the SDMMC. Refer to section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit is set to 1 in either of the following conditions:

- After the end bit of the write command.

- When a write operation is restarted by writing a 1 to SDMMC_BGCR.CONTR (Continue Request).

This bit is cleared to 0 in either of the following conditions:

- After getting the CRC status of the last data block as specified by the transfer count (single and multiple). In case of ADMA2, transfer count is designated by the descriptor table.

- After getting the CRC status of any block where a data transmission is about to be stopped by a Stop At Block Gap Request (STPBGR) of SDMMC_BGCR.

During a write transaction and as the result of the Stop At Block Gap Request (STPBGR) being set, a change from 1 to 0 raises the Block Gap Event (BLKGE) status flag in SDMMC_NISTR if SDMMC_NISTER.BLKGE is set to 1. An interrupt is generated if BLKGE is set to 1 in SDMMC_NISIER. This status is useful to determine whether nonDAT line commands can be issued during Write Busy.

**RTACT: Read Transfer Active**

This bit is used to detect completion of a read transfer. Refer to section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit is set to 1 in either of the following conditions:

- After the end bit of the read command.

- When a read operation is restarted by writing a 1 to SDMMC_BGCR.CONTR (Continue Request).

This bit is cleared to 0 in either of the following conditions:

- When the last data block as specified by Transfer Block Size (BLKSIZE) is transferred to the system.

- In case of ADMA2, end of read is designated by the descriptor table.

### 51.13.25 SDMMC Error Interrupt Status Enable Register (SD_SDIO)

**Name:** SDMMC_EISTER (SD_SDIO)

**Access:** Read/Write

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | ADMA | ACMD |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CURLIM | DATEND | DATCRC | DATTEO | CMDIDX | CMDEND | CMDCRC | CMDTEO |

**CMDTEO: Command Timeout Error Status Enable**

0 (MASKED): The CMDTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDTEO status flag in SDMMC_EISTR is enabled.

**CMDCRC: Command CRC Error Status Enable**

0 (MASKED): The CMDCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDCRC status flag in SDMMC_EISTR is enabled.

**CMDEND: Command End Bit Error Status Enable**

0 (MASKED): The CMDEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDEND status flag in SDMMC_EISTR is enabled.

**CMDIDX: Command Index Error Status Enable**

0 (MASKED): The CMDIDX status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDIDX status flag in SDMMC_EISTR is enabled.

**DATTEO: Data Timeout Error Status Enable**

0 (MASKED): The DATTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATTEO status flag in SDMMC_EISTR is enabled.

**DATCRC: Data CRC Error Status Enable**

0 (MASKED): The DATCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATCRC status flag in SDMMC_EISTR is enabled.

**DATEND: Data End Bit Error Status Enable**

0 (MASKED): The DATEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATEND status flag in SDMMC_EISTR is enabled.

**CURLIM: Current Limit Error Status Enable**

0 (MASKED): The CURLIM status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CURLIM status flag in SDMMC_EISTR is enabled.

**ACMD: Auto CMD Error Status Enable**

0 (MASKED): The ACMD status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ACMD status flag in SDMMC_EISTR is enabled.

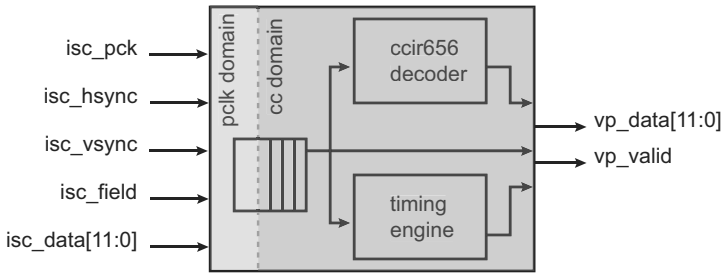
**ADMA: ADMA Error Status Enable**

0 (MASKED): The ADMA status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ADMA status flag in SDMMC_EISTR is enabled.

### 52.5.5 Parallel Front End (PFE) Module

**Figure 52-14: PFE Block Diagram**




The Parallel Front End module performs data resampling across clock domain boundary. It includes a CCIR656 decoder used to convert a standard ITU-R BT.656 stream to 24-bit digital video. It also generates pixels, syncs flags and valid signals to the main video pipeline. It ouputs field, video and synchronization signals. The PFE can optionally crop and limit the incoming pixel stream to a predefined horizontal and vertical value. By default, the PFE only relies on the cmos sensor horizontal and vertical reference to sample the incoming pixel stream. A pixel is sampled if, and only if, the vertical and horizontal synchronizations are valid and a pixel clock edge is detected. ISC_PFE_CFG0.BPS shows the number of bits per sample. The PFE module outputs a 12-bit data on the vp_data[11:0] bus, and asserts the vp_valid signal when the data can be sampled.

| PFE VP_DATA Mapping | Raw Bayer 12-bit | Raw Bayer 10-bit | YUV422 8-bit | YUV422 10-bit | Mono 12-bit |
|---|---|---|---|---|---|
| VP_DATA[11] | RGGB[11] | RGGB[9] | YC422[7] | YC422[9] | Y[11] |
| VP_DATA[10] | RGGB[10] | RGGB[8] | YC422[6] | YC422[8] | Y[10] |
| VP_DATA[9] | RGGB[9] | RGGB[7] | YC422[5] | YC422[7] | Y[9] |
| VP_DATA[8] | RGGB[8] | RGGB[6] | YC422[4] | YC422[6] | Y[8] |
| VP_DATA[7] | RGGB[7] | RGGB[5] | YC422[3] | YC422[5] | Y[7] |
| VP_DATA[6] | RGGB[6] | RGGB[4] | YC422[2] | YC422[4] | Y[6] |
| VP_DATA[5] | RGGB[5] | RGGB[3] | YC422[1] | YC422[3] | Y[5] |
| VP_DATA[4] | RGGB[4] | RGGB[2] | YC422[0] | YC422[2] | Y[4] |
| VP_DATA[3] | RGGB[3] | RGGB[1] | YC422[7] or 0 | YC422[1] | Y[3] |
| VP_DATA[2] | RGGB[2] | RGGB[0] | YC422[6] or 0 | YC422[0] | Y[2] |
| VP_DATA[1] | RGGB[1] | RGGB[9] or 0 | YC422[5] or 0 | YC422[9] or 0 | Y[1] |
| VP_DATA[0] | RGGB[0] | RGGB[8] or 0 | YC422[4] or 0 | YC422[8] or 0 | Y[0] |

**Note:** When ISC_PFE_CFG0.REP is set, missing VP_DATA LSBs are replaced with replicated LSBs of the incoming stream, otherwise they are forced to zero.

The PFE module also includes logic to synchronize capture request with the incoming pixel stream. Two operating modes are available: Single Shot and Continuous Acquisition. When the ISC_PFE_CFG0.CONT field is cleared, the ISC transfers a single image to memory,

### 52.6.47 ISC Subsampling 4:2:2 to 4:2:0 Control Register

**Name:** ISC_SUB420_CTRL

**Address:** 0xF00083CC

**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 |
| – | – | – | FILTER | – | – | – | ENABLE |

**ENABLE: 4:2:2 to 4:2:0 Vertical Subsampling Filter Enable (Center Aligned)**

0: Subsampler disabled

1: Subsampler enabled

**FILTER: Interlaced or Progressive Chrominance Filter**

0: Progressive filter {0.5, 0.5}

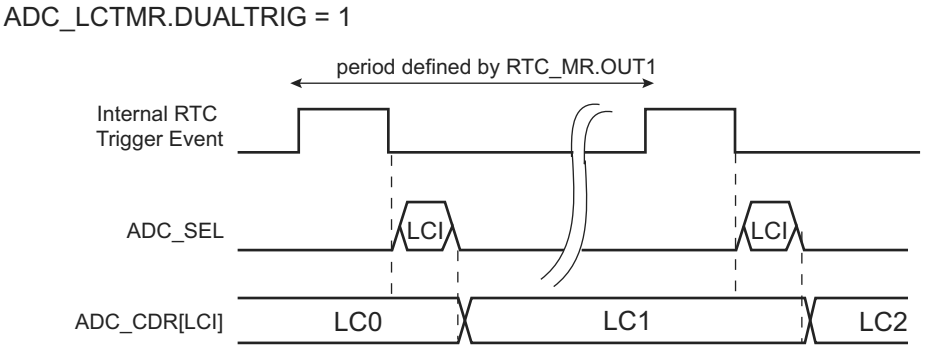
1: Field-dependent filter, top field filter is {0.75, 0.25}, bottom field filter is {0.25, 0.75}

## 60.5 Advanced Encryption Standard (AES) User Interface

**Table 60-5: Register Mapping**

| Offset | Register | Name | Access | Reset |
|---|---|---|---|---|
| 0x00 | Control Register | AES_CR | Write-only | – |
| 0x04 | Mode Register | AES_MR | Read/Write | 0x0 |
| 0x08–0x0C | Reserved | – | – | – |
| 0x10 | Interrupt Enable Register | AES_IER | Write-only | – |
| 0x14 | Interrupt Disable Register | AES_IDR | Write-only | – |
| 0x18 | Interrupt Mask Register | AES_IMR | Read-only | 0x0 |
| 0x1C | Interrupt Status Register | AES_ISR | Read-only | 0x0 |
| 0x20 | Key Word Register 0 | AES_KEYWR0 | Write-only | – |
| 0x24 | Key Word Register 1 | AES_KEYWR1 | Write-only | – |
| 0x28 | Key Word Register 2 | AES_KEYWR2 | Write-only | – |
| 0x2C | Key Word Register 3 | AES_KEYWR3 | Write-only | – |
| 0x30 | Key Word Register 4 | AES_KEYWR4 | Write-only | – |
| 0x34 | Key Word Register 5 | AES_KEYWR5 | Write-only | – |
| 0x38 | Key Word Register 6 | AES_KEYWR6 | Write-only | – |
| 0x3C | Key Word Register 7 | AES_KEYWR7 | Write-only | – |
| 0x40 | Input Data Register 0 | AES_IDATAR0 | Write-only | – |
| 0x44 | Input Data Register 1 | AES_IDATAR1 | Write-only | – |
| 0x48 | Input Data Register 2 | AES_IDATAR2 | Write-only | – |
| 0x4C | Input Data Register 3 | AES_IDATAR3 | Write-only | – |
| 0x50 | Output Data Register 0 | AES_ODATAR0 | Read-only | 0x0 |
| 0x54 | Output Data Register 1 | AES_ODATAR1 | Read-only | 0x0 |
| 0x58 | Output Data Register 2 | AES_ODATAR2 | Read-only | 0x0 |
| 0x5C | Output Data Register 3 | AES_ODATAR3 | Read-only | 0x0 |
| 0x60 | Initialization Vector Register 0 | AES_IVR0 | Write-only | – |
| 0x64 | Initialization Vector Register 1 | AES_IVR1 | Write-only | – |
| 0x68 | Initialization Vector Register 2 | AES_IVR2 | Write-only | – |
| 0x6C | Initialization Vector Register 3 | AES_IVR3 | Write-only | – |
| 0x70 | Additional Authenticated Data Length Register | AES_AADLENR | Read/Write | – |
| 0x74 | Plaintext/Ciphertext Length Register | AES_CLENR | Read/Write | – |
| 0x78 | GCM Intermediate Hash Word Register 0 | AES_GHASHR0 | Read/Write | – |
| 0x7C | GCM Intermediate Hash Word Register 1 | AES_GHASHR1 | Read/Write | – |
| 0x80 | GCM Intermediate Hash Word Register 2 | AES_GHASHR2 | Read/Write | – |
| 0x84 | GCM Intermediate Hash Word Register 3 | AES_GHASHR3 | Read/Write | – |
| 0x88 | GCM Authentication Tag Word Register 0 | AES_TAGR0 | Read-only | – |
| 0x8C | GCM Authentication Tag Word Register 1 | AES_TAGR1 | Read-only | – |

**Figure 65-10: Only Last Channel Measurement Triggered at Low Speed (ADC_CHSR[LCI] = 0 and ADC_TRGR.TRGMOD = 0)**




Notes: ADC_SEL: Command to the ADC analog cell
LCx: Last channel value
LCI: Last channel index

### 65.6.13 Enhanced Resolution Mode and Digital Averaging Function

#### 65.6.13.1 Enhanced Resolution Mode

The Enhanced Resolution mode is enabled if the OSR field is configured to 1 or 2 in ADC_EMR. The enhancement is based on a digital averaging function.

There is no averaging on the last index channel if the measure is triggered by an RTC event.

In this mode, the ADC Controller will trade off conversion speed against accuracy by averaging multiple samples, thus providing a digital low-pass filter function.

The selected oversampling ratio applies to all enabled channels when triggered by an RTC event.

$$\text{ADC\_LCDR.LDATA} = \frac{1}{M} \times \sum_{k=0}^{k=N-1} \text{ADC}(k)$$

where N and M are given in the table below.

**Table 65-5: Digital Averaging Function Configuration versus OSR Values**

| ADC_EMR.OSR Value | ADC_LCDR.LDATA Length | N Value | M Value | Full Scale Value | Maximum Value |
|---|---|---|---|---|---|
| 0 | 12 bits | 1 | 1 | 4095 | 4095 |
| 1 | 13 bits | 4 | 2 | 8191 | 8190 |
| 2 | 14 bits | 16 | 4 | 16383 | 16381 |

The average result is valid in ADC_CDRx (x corresponds to the index of the channel) only if the EOCn flag is set in ADC_ISR and if the OVREn flag is cleared in ADC_OVER. The average result for all channels is valid in ADC_LCDR only if DRDY is set and GOVRE is cleared in ADC_ISR.

Note that ADC_CDRs are not buffered. Therefore, when an averaging sequence is ongoing, the value in these registers changes after each averaging sample. However, overrun flags in ADC_OVER rise as soon as the first sample of an averaging sequence is received. Thus the previous averaged value is not read, even if the new averaged value is not ready.

Consequently, when an overrun flag rises in ADC_OVER, it means that the previous unread data is lost but it does not mean that this data has been overwritten by the new averaged value as the averaging sequence concerning this channel can still be ongoing.