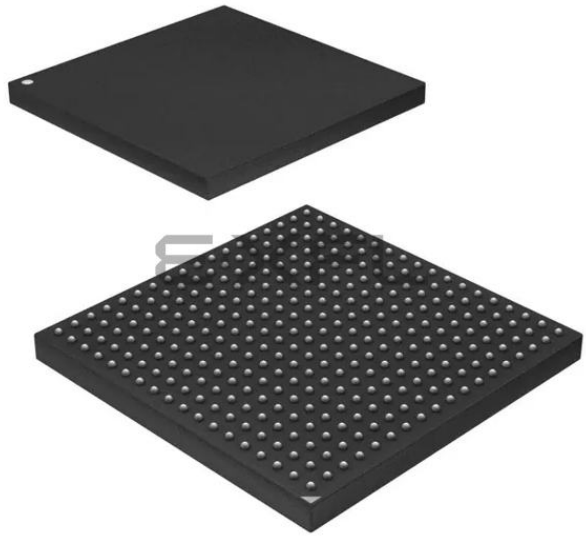


Welcome to E-XFL.COM

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	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/atsama5d35a-cur

Table 10-5 shows the treatment of each different memory type in the Cortex-A5 processor in addition to the architectural requirements.

Table 10-5. Treatment of Memory Attributes

Memory Type Attribute	Shareability	Other attributes	Notes
Strongly Ordered	—	—	—
Device	Non-shareable	—	—
	Shareable	—	—
Normal	Non-shareable	Non-cacheable	Does not access L1 caches
		Write-through cacheable	Treated as non-cacheable
		Write-back cacheable, write allocate	Can dynamically switch to no write allocate, if more than three full cache lines are written in succession
		Write-back cacheable, no write allocate	Treated as non-shareable write-back cacheable, write allocate
	Inner shareable	Non-cacheable	—
		Write-through cacheable	Treated as inner shareable non-cacheable
		Write-back cacheable, write allocate	Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate.
		Write-back cacheable, no write allocate	
	Outer shareable	Non-cacheable	Treated as inner shareable non-cacheable
		Write-through cacheable	
		Write-back cacheable, write allocate	Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate.
		Write-back cacheable, no write allocate	

10.5.3 TLB Organization

TLB Organization is described in the sections that follow:

- Micro TLB
- Main TLB

10.5.3.1 Micro TLB

The first level of caching for the page table information is a micro TLB of 10 entries that is implemented on each of the instruction and data sides. These blocks provide a lookup of the virtual addresses in a single cycle.

The micro TLB returns the physical address to the cache for the address comparison, and also checks the access permissions to signal either a Prefetch Abort or a Data Abort.

All main TLB related maintenance operations affect both the instruction and data micro TLBs, causing them to be flushed. In the same way, any change of the following registers causes the micro TLBs to be flushed:

- Context ID Register (CONTEXTIDR)
- Domain Access Control Register (DACR)
- Primary Region Remap Register (PRRR)
- Normal Memory Remap Register (NMRR)
- Translation Table Base Registers (TTBR0 and TTBR1)

15.2 Embedded Characteristics

- AMBA Advanced High-performance Bus (AHB Lite) Compliant Interfaces
- 32-bit or 64-bit Data Bus
- APB Compliant User Interface
- Configurable Number of Masters (Up to sixteen)
- Configurable Number of Slaves (Up to sixteen)
- One Decoder for Each Master
- Several Possible Boot Memories for Each Master before Remap
- One Remap Function for Each Master
- Support for Long Bursts of 32, 64, 128 and Up to the 256-beat Word Burst AHB Limit
- Enhanced Programmable Mixed Arbitration for Each Slave
  - Round-Robin
  - Fixed Priority
- Programmable Default Master for Each Slave
  - No Default Master
  - Last Accessed Default Master
  - Fixed Default Master
- Deterministic Maximum Access Latency for Masters
- Zero or One Cycle Arbitration Latency for the First Access of a Burst
- Bus Lock Forwarding to Slaves
- Master Number Forwarding to Slaves
- One Special Function Register for Each Slave (Not dedicated)
- Write Protection of User Interface Registers

3. When the instruction loaded at address 0x18 is executed, the program counter is loaded with the value read in AIC_IVR. Reading the AIC_IVR has the following effects:
    - Sets the current interrupt to be the pending and enabled interrupt with the highest priority. The current level is the priority level of the current interrupt.
    - De-asserts the nIRQ line on the processor. Even if vectoring is not used, AIC_IVR must be read in order to de-assert nIRQ.
    - Automatically clears the interrupt, if it has been programmed to be edge-triggered.
    - Pushes the current level and the current interrupt number on to the stack.
    - Returns the value written in the AIC_SVR corresponding to the current interrupt.
4. The previous step has the effect of branching to the corresponding interrupt service routine. This should start by saving the link register (R14_irq) and SPSR_IRQ. The link register must be decremented by four when it is saved if it is to be restored directly into the program counter at the end of the interrupt. For example, the instruction `SUB PC, LR, #4` may be used.
5. Further interrupts can then be unmasked by clearing the “I” bit in CPSR, allowing re-assertion of the nIRQ to be taken into account by the core. This can happen if an interrupt with a higher priority than the current interrupt occurs.
6. The interrupt handler can then proceed as required, saving the registers that will be used and restoring them at the end. During this phase, an interrupt of higher priority than the current level will restart the sequence from step 1.

Note: If the interrupt is programmed to be level sensitive, the source of the interrupt must be cleared during this phase.

7. The “I” bit in CPSR must be set in order to mask interrupts before exiting to ensure that the interrupt is completed in an orderly manner.
8. The End of Interrupt Command Register (AIC_EOICR) must be written in order to indicate to the AIC that the current interrupt is finished. This causes the current level to be popped from the stack, restoring the previous current level if one exists on the stack. If another interrupt is pending, with lower or equal priority than the old current level but with higher priority than the new current level, the nIRQ line is re-asserted, but the interrupt sequence does not immediately start because the “I” bit is set in the core. SPSR_irq is restored. Finally, the saved value of the link register is restored directly into the PC. This has the effect of returning from the interrupt to whatever was being executed before, and of loading the CPSR with the stored SPSR, masking or unmasking the interrupts depending on the state saved in SPSR_irq.

Note: The “I” bit in SPSR is significant. If it is set, it indicates that the ARM core was on the verge of masking an interrupt when the mask instruction was interrupted. Hence, when SPSR is restored, the mask instruction is completed (interrupt is masked).

17.8.4 Fast Interrupt

17.8.4.1 Fast Interrupt Source

The interrupt source 0 is the only source which can raise a fast interrupt request to the processor except if fast forcing is used. The interrupt source 0 is generally connected to a FIQ pin of the product, either directly or through a PIO Controller.

17.8.4.2 Fast Interrupt Control

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

Writing 0x1 in the AIC_IECR (Interrupt Enable Command Register) and AIC_IDCR (Interrupt Disable Command Register) respectively enables and disables the fast interrupt when INTSEL = 0. The bit 0 of AIC_IMR (Interrupt Mask Register) indicates whether the fast interrupt is enabled or disabled.

27.5.4 Output Control

When the I/O line is assigned to a peripheral function, i.e., the corresponding bit in PIO_PSR is at zero, the drive of the I/O line is controlled by the peripheral. Peripheral A or B or C or D depending on the value in PIO_ABCDSR1 and PIO_ABCDSR2 determines whether the pin is driven or not.

When the I/O line is controlled by the PIO Controller, the pin can be configured to be driven. This is done by writing the Output Enable register (PIO_OER) and Output Disable register (PIO_ODR). The results of these write operations are detected in the Output Status register (PIO_OSR). When a bit in this register is at zero, the corresponding I/O line is used as an input only. When the bit is at one, the corresponding I/O line is driven by the PIO Controller.

The level driven on an I/O line can be determined by writing in the Set Output Data register (PIO_SODR) and the Clear Output Data register (PIO_CODR). These write operations, respectively, set and clear the Output Data Status register (PIO_ODSR), which represents the data driven on the I/O lines. Writing in PIO_OER and PIO_ODR manages PIO_OSR whether the pin is configured to be controlled by the PIO Controller or assigned to a peripheral function. This enables configuration of the I/O line prior to setting it to be managed by the PIO Controller.

Similarly, writing in PIO_SODR and PIO_CODR affects PIO_ODSR. This is important as it defines the first level driven on the I/O line.

27.5.5 Synchronous Data Output

Clearing one or more PIO line(s) and setting another one or more PIO line(s) synchronously cannot be done by using PIO_SODR and PIO_CODR registers. It requires two successive write operations into two different registers. To overcome this, the PIO Controller offers a direct control of PIO outputs by single write access to PIO_ODSR. Only bits unmasked by the Output Write Status register (PIO_OWSR) are written. The mask bits in PIO_OWSR are set by writing to the Output Write Enable register (PIO_OWER) and cleared by writing to the Output Write Disable register (PIO_OWDR).

After reset, the synchronous data output is disabled on all the I/O lines as PIO_OWSR resets at 0x0.

27.5.6 Multi-Drive Control (Open Drain)

Each I/O can be independently programmed in open drain by using the multi-drive feature. This feature permits several drivers to be connected on the I/O line which is driven low only by each device. An external pull-up resistor (or enabling of the internal one) is generally required to guarantee a high level on the line.

The multi-drive feature is controlled by the Multi-driver Enable register (PIO_MDER) and the Multi-driver Disable register (PIO_MDDR). The multi-drive can be selected whether the I/O line is controlled by the PIO Controller or assigned to a peripheral function. The Multi-driver Status register (PIO_MDSR) indicates the pins that are configured to support external drivers.

After reset, the multi-drive feature is disabled on all pins, i.e. PIO_MDSR resets at value 0x0.

27.5.7 Output Line Timings

Figure 27-4 shows how the outputs are driven either by writing PIO_SODR or PIO_CODR, or by directly writing PIO_ODSR. This last case is valid only if the corresponding bit in PIO_OWSR is set. Figure 27-4 also shows when the feedback in the Pin Data Status register (PIO_PDSR) is available.

30.17.4 NFC SRAM

30.17.4.1 NFC SRAM Mapping

If the NFC is used to read and write Data from and to the NAND Flash, the configuration depends on the page size, the relevant field is PAGESIZE in HSMC_CFG register. See Table 30-8 to Table 30-12 for detailed mapping.

The NFC SRAM size is 8 Kbytes. The NFC can handle the NAND Flash with a page size of 8 Kbytes or of a lower size (such as 2 Kbytes for example). In case of a 4-Kbyte or lower page size, the NFC SRAM can be split into two banks. The field BANK in the HSMC_BANK register is used to select where NAND flash data are written or read. For 8 Kbytes page size this field is not relevant.

Note that a “ping-pong” mode (write or read to a bank while the NFC writes or reads to another bank) is accessible with the NFC (using two different banks).

If the NFC is not used, the NFC SRAM can be used for a general purpose by the application.

Table 30-8. NFC SRAM Bank Mapping for 512 bytes

Offset	Use	Access
0x00000000–0x000001FF	Main Area Bank 0	Read/Write
0x00000200–0x000003FF	Spare Area Bank 0	Read/Write
0x00001200–0x000013FF	Main Area Bank 1	Read/Write
0x00001400–0x000015FF	Spare Area Bank 1	Read/Write

Table 30-9. NFC SRAM Bank Mapping for 1 Kbyte

Offset	Use	Access
0x00000000–0x000003FF	Main Area Bank 0	Read/Write
0x00000400–0x000005FF	Spare Area Bank 0	Read/Write
0x00001200–0x000015FF	Main Area Bank 1	Read/Write
0x00001600–0x000017FF	Spare Area Bank 1	Read/Write

Table 30-10. NFC SRAM Bank Mapping for 2 Kbytes

Offset	Use	Access
0x00000000–0x000007FF	Main Area Bank 0	Read/Write
0x00000800–0x000009FF	Spare Area Bank 0	Read/Write
0x00001200–0x000019FF	Main Area Bank 1	Read/Write
0x00001A00–0x00001BFF	Spare Area Bank 1	Read/Write

Table 30-11. NFC SRAM Bank Mapping for 4 Kbytes

Offset	Use	Access
0x00000000–0x00000FFF	Main Area Bank 0	Read/Write
0x00001000–0x000011FF	Spare Area Bank 0	Read/Write
0x00001200–0x000021FF	Main Area Bank 1	Read/Write
0x00002200–0x000023FF	Spare Area Bank 1	Read/Write

30.20.36 HSMC Mode Register

Name: HSMC_MODEx [x=0..3]

Address: 0xFFFFC610 [0], 0xFFFFC624 [1], 0xFFFFC638 [2], 0xFFFFC64C [3]

Access: Read/Write

Reset: –

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	TDF_MODE	TDF_CYCLES			
15	14	13	12	11	10	9	8
–	–	–	DBW	–	–	–	BAT
7	6	5	4	3	2	1	0
–	–	EXNW_MODE		–	–	WRITE_MODE	READ_MODE

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

- **READ_MODE: Selection of the Control Signal for Read Operation**

1 (NRD_CTRL): The Read operation is controlled by the NRD signal.

0 (NCS_CTRL): The Read operation is controlled by the NCS signal.

- **WRITE_MODE: Selection of the Control Signal for Write Operation**

1 (NWE_CTRL): The Write operation is controlled by the NWE signal.

0 (NCS_CTRL): The Write operation is controller by the NCS signal.

- **EXNW_MODE: NWAIT Mode**

The NWAIT signal is used to extend the current read or write signal. It is only taken into account during the pulse phase Read and Write controlling signal. When the use of NWAIT is enabled, at least one cycle hold duration must be programmed for the read and write controlling signal.

Value	Name	Description
0	DISABLED	Disabled
1	—	Reserved
2	FROZEN	Frozen Mode
3	READY	Ready Mode

- Disabled: The NWAIT input signal is ignored on the corresponding Chip Select.
- Frozen Mode: If asserted, the NWAIT signal freezes the current read or write cycle. After deassertion, the read/write cycle is resumed from the point where it was stopped.
- Ready Mode: The NWAIT signal indicates the availability of the external device at the end of the pulse of the controlling read or write signal, to complete the access. If high, the access normally completes. If low, the access is extended until NWAIT returns high.

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

Name: DMAC_CTRLAx [x = 0..7]

Address: 0xFFFFE648 (0)[0], 0xFFFFE670 (0)[1], 0xFFFFE698 (0)[2], 0xFFFFE6C0 (0)[3], 0xFFFFE6E8 (0)[4], 0xFFFFE710 (0)[5], 0xFFFFE738 (0)[6], 0xFFFFE760 (0)[7], 0xFFFFE848 (1)[0], 0xFFFFE870 (1)[1], 0xFFFFE898 (1)[2], 0xFFFFE8C0 (1)[3], 0xFFFFE8E8 (1)[4], 0xFFFFE910 (1)[5], 0xFFFFE938 (1)[6], 0xFFFFE960 (1)[7]

Access: Read-write

Reset: 0x00000000

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

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

15	14	13	12	11	10	9	8
BTSIZE							

7	6	5	4	3	2	1	0
BTSIZE							

This register can only be written if the WPEN bit is cleared in “DMAC Write Protect Mode Register” on page 564

- **BTSIZE: Buffer Transfer Size**

The transfer size relates to the number of transfers to be performed, that is, for writes it refers to the number of source width transfers to perform when DMAC is flow controller. For Reads, BTSIZE refers to the number of transfers completed on the Source Interface. When this field is set to 0, the DMAC module is automatically disabled when the relevant channel is enabled.

- **SCSIZE: Source Chunk Transfer Size.**

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

- **DCSIZE: Destination Chunk Transfer Size**

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

32.6.5.4 4:2:2 Planar Mode Frame Buffer Memory Mapping

Table 32-39. 4:2:2 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 32-40. 4:2:2 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	C3[7:0]								C2[7:0]								C1[7:0]								C0[7:0]							

32.6.5.5 4:2:0 Planar Mode Frame Buffer Memory Mapping

In Planar Mode, the three video components Y, Cr and Cb are split into 3 memory areas and stored in a raster-scan order. These three memory planes are contiguous and always aligned on a 32-bit boundary.

Table 32-41. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 32-42. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y7[7:0]								Y6[7:0]								Y5[7:0]								Y4[7:0]							

Table 32-43. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	C3[7:0]								C2[7:0]								C1[7:0]								C0[7:0]							

Table 32-44. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	C7[7:0]								C6:[7:0]								C5[7:0]								C4[7:0]							

32.7.108High End Overlay Layer Configuration 13 Register

Name: LCDC_HEOCFG13

Address: 0xF00303C0

Access: Read-write

Reset: 0x00000000

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

- **SCALEN: Hardware Scaler Enable**

0: Scaler is disabled

1: Scaler is enabled.

- **YFACTOR: Vertical Scaling Factor**

Scaler Vertical Factor.

- **XFACTOR: Horizontal Scaling Factor**

Scaler Horizontal Factor.

32.7.143Hardware Cursor Layer Interrupt Status Register

Name: LCDC_HCRISR

Address: 0xF0030458

Access: Read-only

Reset: 0x00000000

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

- **DMA: End of DMA Transfer**

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

- **DSCR: DMA Descriptor Loaded**

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

- **ADD: Head Descriptor Loaded**

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

- **DONE: End of List Detected**

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

- **OVR: Overflow Detected**

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

34.7.3 UDPHS Interrupt Enable Register

Name: UDPHS_IEN

Address: 0xF8030010

Access: Read-write

31	30	29	28	27	26	25	24
DMA_7	DMA_6	DMA_5	DMA_4	DMA_3	DMA_2	DMA_1	–
23	22	21	20	19	18	17	16
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
15	14	13	12	11	10	9	8
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0
7	6	5	4	3	2	1	0
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	–

- **DET_SUSPD: Suspend Interrupt Enable**

0: Disable Suspend Interrupt.

1: Enable Suspend Interrupt.

- **MICRO_SOF: Micro-SOF Interrupt Enable**

0: Disable Micro-SOF Interrupt.

1: Enable Micro-SOF Interrupt.

- **INT_SOF: SOF Interrupt Enable**

0: Disable SOF Interrupt.

1: Enable SOF Interrupt.

- **ENDRESET: End Of Reset Interrupt Enable**

0: Disable End Of Reset Interrupt.

1: Enable End Of Reset Interrupt. Automatically enabled after USB reset.

- **WAKE_UP: Wake Up CPU Interrupt Enable**

0: Disable Wake Up CPU Interrupt.

1: Enable Wake Up CPU Interrupt.

- **ENDOFRSM: End Of Resume Interrupt Enable**

0: Disable Resume Interrupt.

1: Enable Resume Interrupt.

- **UPSTR_RES: Upstream Resume Interrupt Enable**

0: Disable Upstream Resume Interrupt.

1: Enable Upstream Resume Interrupt.

36.7.62 Broadcast Frames Received Register

Name: GMAC_BCFR

Address: 0xF002815C

Access: Read-only

31	30	29	28	27	26	25	24
BFRX							

23	22	21	20	19	18	17	16
BFRX							

15	14	13	12	11	10	9	8
BFRX							

7	6	5	4	3	2	1	0
BFRX							

- **BFRX: Broadcast Frames Received without Error**

Broadcast frames received without error. This register counts the number of broadcast frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

38.14.11HSMCI Transmit Data Register

Name: HSMCI_TDR

Address: 0xF0000034 (0), 0xF8000034 (1), 0xF8004034 (2)

Access: Write-only

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

- **DATA: Data to Write**

44.8.10 USART Interrupt Mask Register (SPI_MODE)

Name: US_IMR (SPI_MODE)

Address: 0xF001C010 (0), 0xF0020010 (1), 0xF8020010 (2), 0xF8024010 (3)

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
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in “USART Mode Register” on page 1407.

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

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **UNRE: SPI Underrun Error Interrupt Mask**

It is also possible to configure a mailbox in Consumer Mode. In this mode, after each transfer request, a remote frame is automatically sent. The first answer received is stored in the corresponding mailbox data registers.

Several mailboxes can be chained to receive a buffer. They must be configured with the same ID in Receive Mode, except for the last one, which can be configured in Receive with Overwrite Mode. The last mailbox can be used to detect a buffer overflow.

Table 45-4. Receive Mailbox Objects

Object Type	Description
Receive	The first message received is stored in mailbox data registers. Data remain available until the next transfer request.
Receive with overwrite	The last message received is stored in mailbox data register. The next message always overwrites the previous one. The application has to check whether a new message has not overwritten the current one while reading the data registers.
Consumer	A remote frame is sent by the mailbox. The answer received is stored in mailbox data register. This extends Receive mailbox features. Data remain available until the next transfer request.

45.7.2.3 Transmit Mailbox

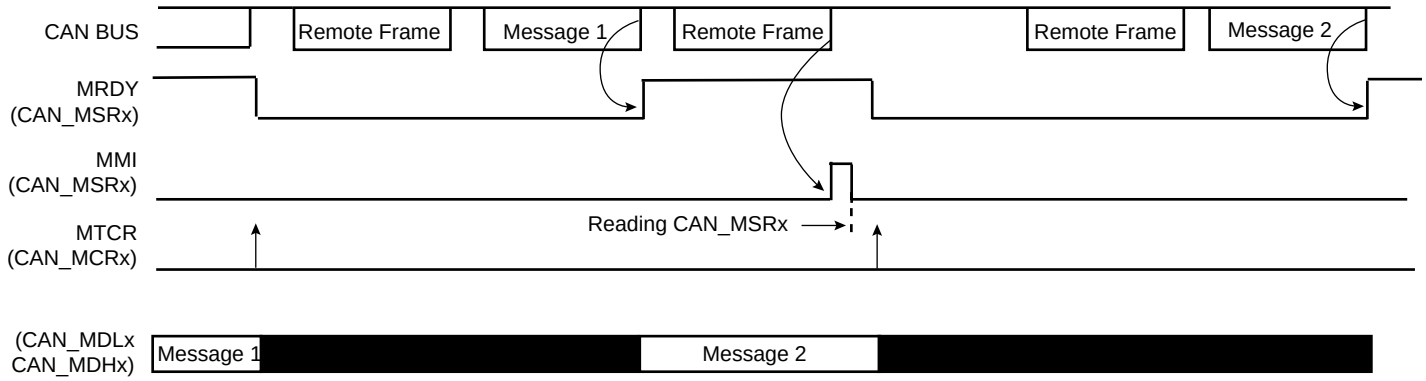
When transmitting a message, the message length and data are written to the transmit mailbox with the correct identifier. For each transmit mailbox, a priority is assigned. The controller automatically sends the message with the highest priority first (set with the field PRIOR in CAN_MMRx).

It is also possible to configure a mailbox in Producer Mode. In this mode, when a remote frame is received, the mailbox data are sent automatically. By enabling this mode, a producer can be done using only one mailbox instead of two: one to detect the remote frame and one to send the answer.

Table 45-5. Transmit Mailbox Objects

Object Type	Description
Transmit	The message stored in the mailbox data registers will try to win the bus arbitration immediately or later according to or not the Time Management Unit configuration (see Section 45.7.3). The application is notified that the message has been sent or aborted.
Producer	The message prepared in the mailbox data registers will be sent after receiving the next remote frame. This extends transmit mailbox features.

Figure 45-17.Producer Handling



Consumer Configuration

A mailbox is in Consumer Mode once the MOT field in the CAN_MMRx has been configured. Message ID and Message Acceptance masks must be set before Receive Mode is enabled.

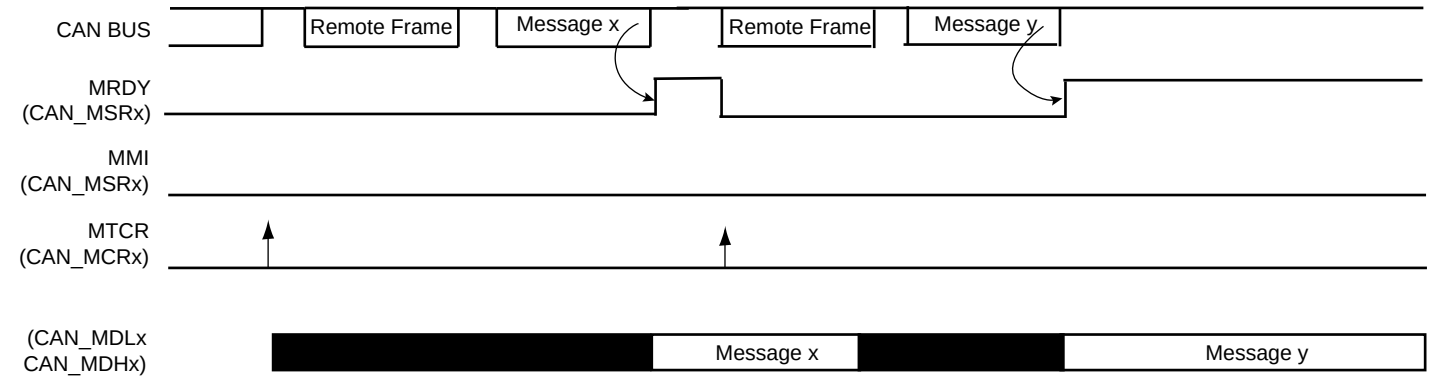
After Consumer Mode is enabled, the MRDY flag in the CAN_MSR is automatically cleared until the first transfer request command. The software application sends a remote frame by setting the MTCR bit in the CAN_MCRx or the MBx bit in the global CAN_TCR. The application is notified of the answer by the MRDY flag set in the CAN_MSRx. The application can read the data contents in the CAN_MDHx and CAN_MDLx registers. An interrupt is pending for the mailbox while the MRDY flag is set. This interrupt can be masked according to the mailbox flag in the CAN_IMR global register.

The MRTR bit in the CAN_MCRx has no effect. This field is used only when using Transmit Mode.

After a remote frame has been sent, the consumer mailbox functions as a reception mailbox. The first message received is stored in the mailbox data registers. If other messages intended for this mailbox have been sent while the MRDY flag is set in the CAN_MSRx, they will be lost. The application is notified by reading the MMI field in the CAN_MSRx. The read operation automatically clears the MMI flag.

If several messages are answered by the Producer, the CAN controller may have one mailbox in consumer configuration, zero or several mailboxes in Receive Mode and one mailbox in Receive with Overwrite Mode. In this case, the consumer mailbox must have a lower number than the Receive with Overwrite mailbox. The transfer command can be triggered for all mailboxes at the same time by setting several MBx fields in the CAN_TCR.

Figure 45-18.Consumer Handling



48.7.17 PWM Output Override Value Register

Name: PWM_OOV

Address: 0xF002C044

Access: Read/Write

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

23	22	21	20	19	18	17	16
–	–	–	–	OOVL3	OOVL2	OOVL1	OOVL0

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	OOVH3	OOVH2	OOVH1	OOVH0

- **OOVHx: Output Override Value for PWMH output of the channel x**

0: Override value is 0 for PWMH output of channel x.

1: Override value is 1 for PWMH output of channel x.

- **OOVLx: Output Override Value for PWML output of the channel x**

0: Override value is 0 for PWML output of channel x.

1: Override value is 1 for PWML output of channel x.

The gain is configurable through the GAIN bit of the Channel Gain Register (ADC_CGR) as shown in Table 49-6.

Table 49-6. Gain of the Sample and Hold Unit: GAIN Bits and DIFF Bit.

GAIN<0:1>	GAIN (DIFF = 0)	GAIN (DIFF = 1)
00	1	0.5
01	1	1
10	2	2
11	4	2

To allow full range, analog offset of the ADC can be configured by the OFFSET bit of the Channel Offset Register (ADC_COR). The Offset is only available in Single Ended Mode.

Table 49-7. Offset of the Sample and Hold Unit: OFFSET DIFF and Gain (G)

OFFSET Bit	OFFSET (DIFF = 0)	OFFSET (DIFF = 1)
0	0	0
1	(G-1)Vrefin/2	

49.8.25 ADC Write Protect Mode Register

Name: ADC_WPMR

Address: 0xF80180E4

Access: Read-write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protect Enable**

0 = Disables the Write Protect if WPKEY corresponds to 0x414443 (“ADC” in ASCII).

1 = Enables the Write Protect if WPKEY corresponds to 0x414443 (“ADC” in ASCII).

Protects the registers:

- “ADC Mode Register” on page 1646
- “ADC Channel Sequence 1 Register” on page 1649
- “ADC Channel Sequence 2 Register” on page 1650
- “ADC Channel Enable Register” on page 1651
- “ADC Channel Disable Register” on page 1652
- “ADC Extended Mode Register” on page 1661
- “ADC Compare Window Register” on page 1662
- “ADC Channel Gain Register” on page 1663
- “ADC Channel Offset Register” on page 1664
- “ADC Analog Control Register” on page 1666
- “ADC Touchscreen Mode Register” on page 1667
- “ADC Trigger Register” on page 1672

- **WPKEY: Write Protect KEY**

Value	Name	Description
0x414443	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0

54.13.2 Timing extraction

54.13.2.1 Read Timings

Table 54-37. SMC Read Signals - NRD Controlled (READ_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	VDDIOM supply	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (nrd hold = 0)						
SMC_1	Data Setup before NRD High	9.6	6.8	—	—	ns
SMC_2	Data Hold after NRD High	0	0	—	—	ns
HOLD SETTINGS (nrd hold ≠ 0)						
SMC_3	Data Setup before NRD High	9.4	6.6	—	—	ns
SMC_4	Data Hold after NRD High	0	0	—	—	ns
HOLD or NO HOLD SETTINGS (nrd hold ≠ 0, nrd hold =0)						
SMC_5	NBS0/A0, NBS1, NBS2/A1, NBS3, A2 - A25 Valid before NRD High	(nrd setup + nrd pulse)* t_{CPMCK} + 2.4	(nrd setup + nrd pulse)* t_{CPMCK} + 2.1	—	—	ns
SMC_6	NCS low before NRD High	(nrd setup + nrd pulse - ncs rd setup) * t_{CPMCK} + 2	(nrd setup + nrd pulse - ncs rd setup) * t_{CPMCK} + 1.9	—	—	ns
SMC_7	NRD Pulse Width	nrd pulse * t_{CPMCK} + 1.8	nrd pulse * t_{CPMCK} +1.2	—	—	ns

Table 54-38. SMC Read Signals - NCS Controlled (READ_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	VDDIOM supply	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (ncs rd hold = 0)						
SMC_8	Data Setup before NCS High	18.5	16.2	—	—	ns
SMC_9	Data Hold after NCS High	0	0	—	—	ns
HOLD SETTINGS (ncs rd hold ≠ 0)						
SMC_{10}	Data Setup before NCS High	17	14.7	—	—	ns
SMC_{11}	Data Hold after NCS High	0	0	—	—	ns
HOLD or NO HOLD SETTINGS (ncs rd hold ≠ 0, ncs rd hold = 0)						
SMC_{12}	NBS0/A0, NBS1, NBS2/A1, NBS3, A2 - A25 valid before NCS High	(ncs rd setup + ncs rd pulse)* t_{CPMCK} + 19.1	(ncs rd setup + ncs rd pulse)* t_{CPMCK} + 18.8	—	—	ns
SMC_{13}	NRD low before NCS High	(ncs rd setup + ncs rd pulse - nrd setup)* t_{CPMCK} + 16.8	(ncs rd setup + ncs rd pulse - nrd setup)* t_{CPMCK} +16.1	—	—	ns
SMC_{14}	NCS Pulse Width	ncs rd pulse length * t_{CPMCK} + 2	ncs rd pulse length * t_{CPMCK} + 1.5	—	—	ns