Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	ARM® Cortex®-M4
Core Size	32-Bit Single-Core
Speed	120MHz
Connectivity	EBI/EMI, I²C, IrDA, Memory Card, SPI, SSC, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	79
Program Memory Size	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	160K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 16x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-TFBGA
Supplier Device Package	100-TFBGA (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsam4sd16ca-cu

4.2.4 64-lead LQFP and QFN Pinout

Table 4-4. 64-pin SAM4SD32/SD16/SA16/S16/S8/S4/S2 Pinout

1	ADVREF	17	GND	33	TDI/PB4	49	TDO/TRACESWO/PB5
2	GND	18	VDDIO	34	PA6/PGMNOE	50	JTAGSEL
3	PB0/AD4	19	PA16/PGMD4	35	PA5/PGMRDY	51	TMS/SWDIO/PB6
4	PB1/AD5	20	PA15/PGMD3	36	PA4/PGMNCMD	52	PA31
5	PB2/AD6	21	PA14/PGMD2	37	PA27/PGMD15	53	TCK/SWCLK/PB7
6	PB3/AD7	22	PA13/PGMD1	38	PA28	54	VDDCORE
7	VDDIN	23	PA24/PGMD12	39	NRST	55	ERASE/PB12
8	VDDOUT	24	VDDCORE	40	TST	56	DDM/PB10
9	PA17/PGMD5/AD0	25	PA25/PGMD13	41	PA29	57	DDP/PB11
10	PA18/PGMD6/AD1	26	PA26/PGMD14	42	PA30	58	VDDIO
11	PA21/PGMD9/AD8	27	PA12/PGMD0	43	PA3	59	PB13/DAC0
12	VDDCORE	28	PA11/PGMM3	44	PA2/PGMEN2	60	GND
13	PA19/PGMD7/AD2	29	PA10/PGMM2	45	VDDIO	61	XOUT/PB8
14	PA22/PGMD10/AD9	30	PA9/PGMM1	46	GND	62	XIN/PGMCK/PB9
15	PA23/PGMD11	31	PA8/XOUT32/PGMM0	47	PA1/PGMEN1	63	PB14/DAC1
16	PA20/PGMD8/AD3	32	PA7/XIN32/PGMNVALID	48	PA0/PGMEN0	64	VDDPLL

Note: The bottom pad of the QFN package must be connected to ground.

11.2.3 PIO Controller C Multiplexing

Table 11-4. Multiplexing on PIO Controller C (PIOC)

I/O Line	Peripheral A	Peripheral B	Peripheral C	Extra Function	System Function	Comments
PC0	D0	PWML0				100-pin version
PC1	D1	PWML1				100-pin version
PC2	D2	PWML2				100-pin version
PC3	D3	PWML3				100-pin version
PC4	D4	NPCS1				100-pin version
PC5	D5					100-pin version
PC6	D6					100-pin version
PC7	D7					100-pin version
PC8	NWE					100-pin version
PC9	NANDOE					100-pin version
PC10	NANDWE					100-pin version
PC11	NRD					100-pin version
PC12	NCS3			AD12(1)		100-pin version
PC13	NWAIT	PWML0		AD10(1)		100-pin version
PC14	NCS0					100-pin version
PC15	NCS1	PWML1		AD11(1)		100-pin version
PC16	A21/NANDALE					100-pin version
PC17	A22/NANDCLE					100-pin version
PC18	A0	PWMH0				100-pin version
PC19	A1	PWMH1				100-pin version
PC20	A2	PWMH2				100-pin version
PC21	A3	PWMH3				100-pin version
PC22	A4	PWML3				100-pin version
PC23	A5	TIOA3				100-pin version
PC24	A6	TIOB3				100-pin version
PC25	A7	TCLK3				100-pin version
PC26	A8	TIOA4				100-pin version
PC27	A9	TIOB4				100-pin version
PC28	A10	TCLK4				100-pin version
PC29	A11	TIOA5		AD13(1)		100-pin version
PC30	A12	TIOB5		AD14(1)		100-pin version
PC31	A13	TCLK5				100-pin version

Note: 1. To select this extra function, refer to Section 42.5.3 “Analog Inputs”.

- Prefetches instructions ahead of execution
- Speculatively prefetches from branch target addresses.

12.4.2.4 Software Ordering of Memory Accesses

The order of instructions in the program flow does not always guarantee the order of the corresponding memory transactions. This is because:

- The processor can reorder some memory accesses to improve efficiency, providing this does not affect the behavior of the instruction sequence.
- The processor has multiple bus interfaces
- Memory or devices in the memory map have different wait states
- Some memory accesses are buffered or speculative.

"Memory System Ordering of Memory Accesses" describes the cases where the memory system guarantees the order of memory accesses. Otherwise, if the order of memory accesses is critical, the software must include memory barrier instructions to force that ordering. The processor provides the following memory barrier instructions:

DMB

The *Data Memory Barrier* (DMB) instruction ensures that outstanding memory transactions complete before subsequent memory transactions. See "DMB" .

DSB

The *Data Synchronization Barrier* (DSB) instruction ensures that outstanding memory transactions complete before subsequent instructions execute. See "DSB" .

ISB

The *Instruction Synchronization Barrier* (ISB) ensures that the effect of all completed memory transactions is recognizable by subsequent instructions. See "ISB" .

MPU Programming

Use a DSB followed by an ISB instruction or exception return to ensure that the new MPU configuration is used by subsequent instructions.

12.4.2.5 Bit-banding

A bit-band region maps each word in a *bit-band alias* region to a single bit in the *bit-band region*. The bit-band regions occupy the lowest 1 MB of the SRAM and peripheral memory regions.

The memory map has two 32 MB alias regions that map to two 1 MB bit-band regions:

- Accesses to the 32 MB SRAM alias region map to the 1 MB SRAM bit-band region, as shown in Table 12-6.
- Accesses to the 32 MB peripheral alias region map to the 1 MB peripheral bit-band region, as shown in Table 12-7.

Table 12-6. SRAM Memory Bit-banding Regions

Address Range	Memory Region	Instruction and Data Accesses
0x20000000–0x200FFFFF	SRAM bit-band region	Direct accesses to this memory range behave as SRAM memory accesses, but this region is also bit-addressable through bit-band alias.
0x22000000–0x23FFFFFF	SRAM bit-band alias	Data accesses to this region are remapped to bit-band region. A write operation is performed as read-modify-write. Instruction accesses are not remapped.

12.6.5.24 USUB16 and USUB8

Unsigned Subtract 16 and Unsigned Subtract 8

Syntax

```
op{cond}{Rd,} Rn, Rm
```

where

op is any of:

USUB16 Unsigned Subtract 16.

USUB8 Unsigned Subtract 8.

cond is an optional condition code, see "Conditional Execution" .

Rd is the destination register.

Rn is the first operand register.

Rm is the second operand register.

Operation

Use these instructions to subtract 16-bit and 8-bit data before writing the result to the destination register:

The USUB16 instruction:

1. Subtracts each halfword from the second operand register from the corresponding halfword of the first operand register.
2. Writes the unsigned result in the corresponding halfwords of the destination register.

The USUB8 instruction:

1. Subtracts each byte of the second operand register from the corresponding byte of the first operand register.
2. Writes the unsigned byte result in the corresponding byte of the destination register.

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not change the flags.

Examples

```
USUB16 R1, R0  ; Subtracts halfwords in R0 from corresponding halfword of R1
               ; and writes to corresponding halfword in R1USUB8  R4, R0, R5
               ; Subtracts bytes of R5 from corresponding byte in R0 and
               ; writes to the corresponding byte in R4.
```

12.8.3.7 Software Trigger Interrupt Register

Name: NVIC_STIR

Access: Write-only

Reset: 0x000000000

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

Write to this register to generate an interrupt from the software.

- **INTID: Interrupt ID**

Interrupt ID of the interrupt to trigger, in the range 0–239. For example, a value of 0x03 specifies interrupt IRQ3.

The DWT contains counters for the items that follow:

- Clock cycle (CYCCNT)
- Folded instructions
- Load Store Unit (LSU) operations
- Sleep Cycles
- CPI (all instruction cycles except for the first cycle)
- Interrupt overhead

13.5.6 ITM (Instrumentation Trace Macrocell)

The ITM is an application driven trace source that supports printf style debugging to trace Operating System (OS) and application events, and emits diagnostic system information. The ITM emits trace information as packets which can be generated by three different sources with several priority levels:

- **Software trace**: Software can write directly to ITM stimulus registers. This can be done thanks to the "printf" function. For more information, refer to Section 13.5.6.1 "How to Configure the ITM".
- **Hardware trace**: The ITM emits packets generated by the DWT.
- **Time stamping**: Timestamps are emitted relative to packets. The ITM contains a 21-bit counter to generate the timestamp.

13.5.6.1 How to Configure the ITM

The following example describes how to output trace data in asynchronous trace mode.

- Configure the TPIU for asynchronous trace mode (refer to Section 13.5.6.3 "5.4.3. How to Configure the TPIU")
- Enable the write accesses into the ITM registers by writing "0xC5ACCE55" into the Lock Access Register (Address: 0xE0000FB0)
- Write 0x00010015 into the Trace Control Register:
  - Enable ITM
  - Enable Synchronization packets
  - Enable SWO behavior
  - Fix the ATB ID to 1
- Write 0x1 into the Trace Enable Register:
  - Enable the Stimulus port 0
- Write 0x1 into the Trace Privilege Register:
  - Stimulus port 0 only accessed in privileged mode (Clearing a bit in this register will result in the corresponding stimulus port being accessible in user mode.)
- Write into the Stimulus port 0 register: TPIU (Trace Port Interface Unit)

  The TPIU acts as a bridge between the on-chip trace data and the Instruction Trace Macrocell (ITM).

  The TPIU formats and transmits trace data off-chip at frequencies asynchronous to the core.

13.5.6.2 Asynchronous Mode

The TPIU is configured in asynchronous mode, trace data are output using the single TRACESWO pin. The TRACESWO signal is multiplexed with the TDO signal of the JTAG Debug Port. As a consequence, asynchronous trace mode is only available when the Serial Wire Debug mode is selected since TDO signal is used in JTAG debug mode.

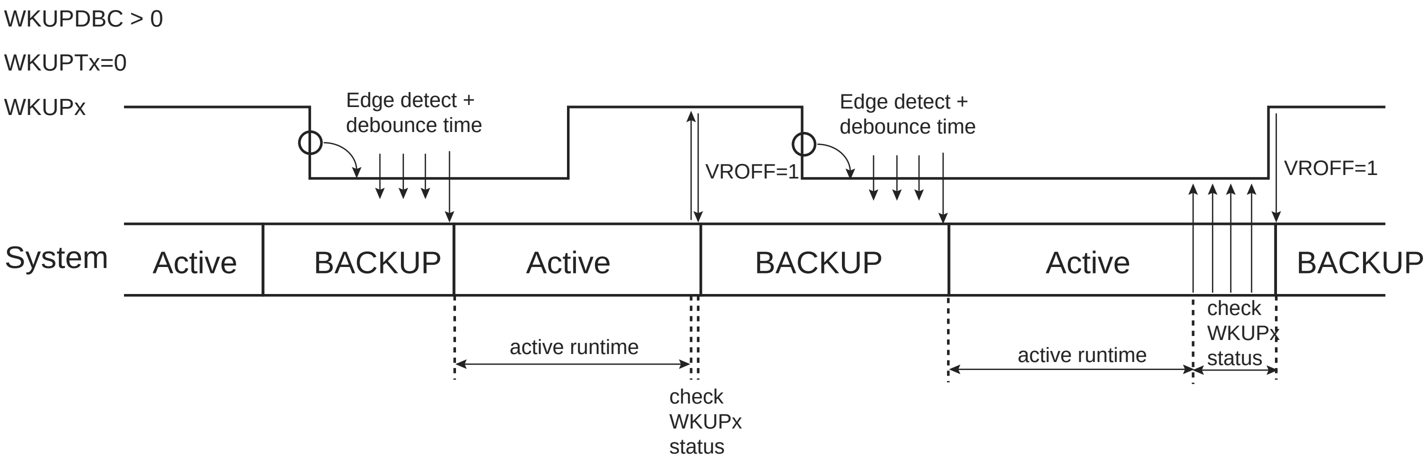
Two encoding formats are available for the single pin output:

- Manchester encoded stream. This is the reset value.
- NRZ_based UART byte structure

If an enabled WKUP pin is asserted for a duration longer than the debouncing period, a wake-up of the core power supply is started and the signals, WKUP0 to WKUPx as shown in Figure 18-4 "Wake-up Sources", are latched in SUPC_SR. This allows the user to identify the source of the wake-up. However, if a new wake-up condition occurs, the primary information is lost. No new wake-up can be detected since the primary wake-up condition has disappeared.

Before instructing the system to enter Backup mode, if the field WKUPDBC > 0, it must be checked that none of the WKUPx pins that are enabled for a wake-up (exit from Backup mode) holds an active polarity. This is checked by reading the pin status in the PIO Controller. If WKUPENx=1 and the pin WKUPx holds an active polarity, the system must not be instructed to enter Backup mode.

Figure 18-5. Entering and Exiting Backup Mode with a WKUP Pin



18.4.7.2 Low-power Tamper Detection and Anti-Tampering

Low-power debouncer inputs (WKUP0, WKUP1) can be used for tamper detection. If the tamper sensor is biased through a resistor and constantly driven by the power supply, this leads to power consumption as long as the tamper detection switch is in its active state. To prevent power consumption when the switch is in active state, the tamper sensor circuitry must be intermittently powered, and thus a specific waveform must be applied to the sensor circuitry.

The waveform is generated using RTCOUTx in all modes including Backup mode. Refer to the RTC section for waveform generation.

Separate debouncers are embedded, one for WKUP0 input, one for WKUP1 input.

The WKUP0 and/or WKUP1 inputs perform a system wake-up upon tamper detection. This is enabled by setting the LPDBCEN0/1 bit in the SUPC_WUMR.

WKUP0 and/or WKUP1 inputs can also be used when VDDCORE is powered to detect a tamper.

When the bit LPDBCENx is written to 1, WKUPx pins must not be configured to act as a debouncing source for the WKUPDBC counter (WKUPENx must be cleared in SUPC_WUIR).

Low-power tamper detection or debounce requires RTC output (RTCOUTx) to be configured to generate a duty cycle programmable pulse (i.e., OUT0 = 0x7 in RTC_MR) in order to create the sampling points of both debouncers. The sampling point is the falling edge of the RTCOUTx waveform.

Figure 18-6 shows an example of an application where two tamper switches are used. RTCOUTx powers the external pull-up used by the tamper sensor circuitry.

29.17.9 PMC Clock Generator PLLA Register

Name: CKGR_PLLAR

Address: 0x400E0428

Access: Read/Write

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

Possible limitations on PLLA input frequencies and multiplier factors should be checked before using the PMC.

Warning: Bit 29 must always be set to 1 when programming the CKGR_PLLAR.

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

- **DIVA:** PLLA Front_End Divider

0: Divider output is stuck at 0 and PLLA is disabled.

1: Divider is bypassed (divide by 1) PLLA is enabled

2–255: Clock is divided by DIVA

- **PLLACOUNT: PLLA Counter**

Specifies the number of Slow Clock cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

- **MULA: PLLA Multiplier**

0: The PLLA is deactivated (PLLA also disabled if DIVA = 0).

7 up to 62 = The PLLA Clock frequency is the PLLA input frequency multiplied by MULA + 1.

Unlisted values are forbidden.

- **ONE: Must Be Set to 1**

Bit 29 must always be set to 1 when programming the CKGR_PLLAR.

29.17.17PMC Interrupt Mask Register

Name: PMC_IMR

Address: 0x400E046C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
–	–	–	–	–	CFDEV	MOSCRCS	MOSCSELS
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
–	–	–	–	MCKRDY	LOCKB	LOCKA	MOSCXTS

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

0: No effect.

1: Enables the corresponding interrupt.

- **MOSCXTS: Main Crystal Oscillator Status Interrupt Mask**

- **LOCKA: PLLA Lock Interrupt Mask**

- **LOCKB: PLLB Lock Interrupt Mask**

- **MCKRDY: Master Clock Ready Interrupt Mask**

- **PCKRDYx: Programmable Clock Ready x Interrupt Mask**

- **MOSCSELS: Main Oscillator Selection Status Interrupt Mask**

- **MOSCRCS: Main On-Chip RC Status Interrupt Mask**

- **CFDEV: Clock Failure Detector Event Interrupt Mask**

31.6.12 PIO Output Data Status Register

Name: PIO_ODSR

Address: 0x400E0E38 (PIOA), 0x400E1038 (PIOB), 0x400E1238 (PIOC)

Access: Read-only or Read/Write

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: Output Data Status**

0: The data to be driven on the I/O line is 0.

1: The data to be driven on the I/O line is 1.

31.6.16 PIO Interrupt Mask Register

Name: PIO_IMR

Address: 0x400E0E48 (PIOA), 0x400E1048 (PIOB), 0x400E1248 (PIOC)

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: Input Change Interrupt Mask**

0: Input change interrupt is disabled on the I/O line.

1: Input change interrupt is enabled on the I/O line.

31.6.27 PIO Input Filter Slow Clock Enable Register

Name: PIO_IFSCER

Address: 0x400E0E84 (PIOA), 0x400E1084 (PIOB), 0x400E1284 (PIOC)

Access: Write-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: Slow Clock Debouncing Filtering Select**

0: No effect.

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

31.6.45 PIO Lock Status Register

Name: PIO_LOCKSR

Address: 0x400E0EE0 (PIOA), 0x400E10E0 (PIOB), 0x400E12E0 (PIOC)

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: Lock Status**

0: The I/O line is not locked.

1: The I/O line is locked.

32.9 Synchronous Serial Controller (SSC) User Interface

Table 32-5. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Control Register	SSC_CR	Write-only	–
0x4	Clock Mode Register	SSC_CMR	Read/Write	0x0
0x8–0xC	Reserved	–	–	–
0x10	Receive Clock Mode Register	SSC_RCMR	Read/Write	0x0
0x14	Receive Frame Mode Register	SSC_RFMR	Read/Write	0x0
0x18	Transmit Clock Mode Register	SSC_TCMR	Read/Write	0x0
0x1C	Transmit Frame Mode Register	SSC_TFMR	Read/Write	0x0
0x20	Receive Holding Register	SSC_RHR	Read-only	0x0
0x24	Transmit Holding Register	SSC_THR	Write-only	–
0x28–0x2C	Reserved	–	–	–
0x30	Receive Sync. Holding Register	SSC_RSHR	Read-only	0x0
0x34	Transmit Sync. Holding Register	SSC_TSHR	Read/Write	0x0
0x38	Receive Compare 0 Register	SSC_RC0R	Read/Write	0x0
0x3C	Receive Compare 1 Register	SSC_RC1R	Read/Write	0x0
0x40	Status Register	SSC_SR	Read-only	0x000000CC
0x44	Interrupt Enable Register	SSC_IER	Write-only	–
0x48	Interrupt Disable Register	SSC_IDR	Write-only	–
0x4C	Interrupt Mask Register	SSC_IMR	Read-only	0x0
0x50–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	SSC_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	SSC_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–
0x100–0x128	Reserved for PDC registers	–	–	–

33.5 Signal Description

Table 33-1. Signal Description

Pin Name	Pin Description	Type	
		Master	Slave
MISO	Master In Slave Out	Input	Output
MOSI	Master Out Slave In	Output	Input
SPCK	Serial Clock	Output	Input
NPCS1–NPCS3	Peripheral Chip Selects	Output	Unused
NPCS0/NSS	Peripheral Chip Select/Slave Select	Output	Input

33.6 Product Dependencies

33.6.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the SPI pins to their peripheral functions.

Table 33-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
SPI	MISO	PA12	A
SPI	MOSI	PA13	A
SPI	NPCS0	PA11	A
SPI	NPCS1	PA9	B
SPI	NPCS1	PA31	A
SPI	NPCS1	PB14	A
SPI	NPCS1	PC4	B
SPI	NPCS2	PA10	B
SPI	NPCS2	PA30	B
SPI	NPCS2	PB2	B
SPI	NPCS3	PA3	B
SPI	NPCS3	PA5	B
SPI	NPCS3	PA22	B
SPI	SPCK	PA14	A

33.6.2 Power Management

The SPI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SPI clock.

36.7.9 USART Interrupt Mask Register

Name: US_IMR

Address: 0x40024010 (0), 0x40028010 (1)

Access: Read-only

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

23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	DCDIC	DSRIC	RIIC

15	14	13	12	11	10	9	8
–	–	NACK	RXBUFF	TXBUFE	ITER	TXEMPTY	TIMEOUT

7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	TXRDY	RXRDY

For SPI specific configuration, see Section 36.7.10 ”USART Interrupt Mask Register (SPI_MODE)”.

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**

- **RXBRK: Receiver Break Interrupt Mask**

- **ENDRX: End of Receive Buffer Interrupt Mask (available in all USART modes of operation)**

- **ENDTX: End of Transmit Buffer Interrupt Mask (available in all USART modes of operation)**

- **OVRE: Overrun Error Interrupt Mask**

- **FRAME: Framing Error Interrupt Mask**

- **PARE: Parity Error Interrupt Mask**

- **TIMEOUT: Time-out Interrupt Mask**

- **TXEMPTY: TXEMPTY Interrupt Mask**

- **ITER: Max Number of Repetitions Reached Interrupt Mask**

- **TXBUFE: Transmit Buffer Empty Interrupt Mask (available in all USART modes of operation)**

- **RXBUFF: Receive Buffer Full Interrupt Mask (available in all USART modes of operation)**

- **NACK: Non Acknowledge Interrupt Mask**

- **RIIC: Ring Indicator Input Change Mask**

- To prevent any unexpected activation of the Fault Protection on the channel x, the bit FPEx[y] can be set to '1' only if the FPOLy bit has been previously configured to its final value.

If a comparison unit is enabled (see Section 39.6.3 "PWM Comparison Units") and if a fault is triggered in the channel 0, then the comparison cannot match.

As soon as the fault protection is triggered on a channel, an interrupt (different from the interrupt generated at the end of the PWM period) can be generated but only if it is enabled and not masked. The interrupt is reset by reading the interrupt status register, even if the fault which has caused the trigger of the fault protection is kept active.

39.6.2.7 Synchronous Channels

Some channels can be linked together as synchronous channels. They have the same source clock, the same period, the same alignment and are started together. In this way, their counters are synchronized together.

The synchronous channels are defined by the SYNCx bits in the PWM Sync Channels Mode Register (PWM_SCM). Only one group of synchronous channels is allowed.

When a channel is defined as a synchronous channel, the channel 0 is also automatically defined as a synchronous channel. This is because the channel 0 counter configuration is used by all the synchronous channels.

If a channel x is defined as a synchronous channel, the fields/bits for the channel 0 are used instead of those of channel x:

- CPRE in PWM_CMR0 instead of CPRE in PWM_CMRx (same source clock)
- CPRD in PWM_CPRD0 instead of CPRD in PWM_CPRDx (same period)
- CALG in PWM_CMR0 instead of CALG in PWM_CMRx (same alignment)

Modifying the fields CPRE, CPRD and CALG of for channels with index greater than 0 has no effect on output waveforms.

Because counters of synchronous channels must start at the same time, they are all enabled together by enabling the channel 0 (by the CHID0 bit in PWM_ENA register). In the same way, they are all disabled together by disabling channel 0 (by the CHID0 bit in PWM_DIS register). However, a synchronous channel x different from channel 0 can be enabled or disabled independently from others (by the CHIDx bit in PWM_ENA and PWM_DIS registers).

Defining a channel as a synchronous channel while it is an asynchronous channel (by writing the bit SYNCx to '1' while it was at '0') is allowed only if the channel is disabled at this time (CHIDx = 0 in PWM_SR). In the same way, defining a channel as an asynchronous channel while it is a synchronous channel (by writing the SYNCx bit to '0' while it was '1') is allowed only if the channel is disabled at this time.

The UPDM field (Update Mode) in the PWM_SCM register selects one of the three methods to update the registers of the synchronous channels:

- Method 1 (UPDM = 0): The period value, the duty-cycle values and the dead-time values must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).The update is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM Sync Channels Update Control Register (PWM_SCUC) is set to '1' (see "Method 1: Manual write of duty-cycle values and manual trigger of the update").
- Method 2 (UPDM = 1): The period value, the duty-cycle values, the dead-time values and the update period value must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPD). The update of the period value and of the dead-time values is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM_SCUC register is set to '1'. The update of the duty-cycle values and the update period value is triggered automatically after an update period defined by the field UPR in the PWM Sync Channels Update Period Register (PWM_SCUP) (see "Method 2: Manual write of duty-cycle values and automatic trigger of the update").

39.7.1 PWM Clock Register

Name: PWM_CLK

Address: 0x40020000

Access: Read/Write

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

23	22	21	20	19	18	17	16
DIVB							

15	14	13	12	11	10	9	8
–	–	–	–	PREA			

7	6	5	4	3	2	1	0
DIVA							

This register can only be written if bits WPSWS0 and WPHWS0 are cleared in the PWM Write Protection Status Register.

- **DIVA: CLKA Divide Factor**

Value	Name	Description
0	CLKA_POFF	CLKA clock is turned off
1	PREA	CLKA clock is clock selected by PREA
2–255	PREA_DIV	CLKA clock is clock selected by PREA divided by DIVA factor

- **DIVB: CLKB Divide Factor**

Value	Name	Description
0	CLKB_POFF	CLKB clock is turned off
1	PREB	CLKB clock is clock selected by PREB
2–255	PREB_DIV	CLKB clock is clock selected by PREB divided by DIVB factor

- **PREA: CLKA Source Clock Selection**

Value	Name	Description
0	CLK	Peripheral clock
1	CLK_DIV2	Peripheral clock/2
2	CLK_DIV4	Peripheral clock/4
3	CLK_DIV8	Peripheral clock/8
4	CLK_DIV16	Peripheral clock/16
5	CLK_DIV32	Peripheral clock/32
6	CLK_DIV64	Peripheral clock/64
7	CLK_DIV128	Peripheral clock/128
8	CLK_DIV256	Peripheral clock/256

42.7.16 ADC Channel Gain Register

Name: ADC_CGR

Address: 0x40038048

Access: Read/Write

31	30	29	28	27	26	25	24
GAIN15		GAIN14		GAIN13		GAIN12	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
GAIN11		GAIN10		GAIN9		GAIN8	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
GAIN7		GAIN6		GAIN5		GAIN4	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GAIN3		GAIN2		GAIN1		GAIN0	

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

- **GAINx: Gain for Channel x**

Gain applied on input of analog-to-digital converter.

Value	Name	Description
0	SE1_DIFF0_5	Single-ended gain = 1 (ADC_COR.DIFFx = 0), differential gain = 0.5 (ADC_COR.DIFFx = 1)
1	SE1_DIFF1	Single-ended gain = 1 (ADC_COR.DIFFx = 0), differential gain = 1 (ADC_COR.DIFFx = 1)
2	SE2_DIFF2	Single-ended gain = 2 (ADC_COR.DIFFx = 0), differential gain = 2 (ADC_COR.DIFFx = 1)
3	SE4_DIFF2	Single-ended gain = 4 (ADC_COR.DIFFx = 0), differential gain = 2 (ADC_COR.DIFFx = 1)

The DIFFx bit in this table is described in Section 42.7.17 “ADC Channel Offset Register”.

43.7.10 DACC Interrupt Status Register

Name: DACC_ISR

Address: 0x4003C030

Access: Read-only

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

- **TXRDY: Transmit Ready Interrupt Flag**

0: DACC is not ready to accept new conversion requests.

1: DACC is ready to accept new conversion requests.

- **EOC: End of Conversion Interrupt Flag**

0: No conversion has been performed since the last DACC_ISR read.

1: At least one conversion has been performed since the last DACC_ISR read.

- **ENDTX: End of DMA Interrupt Flag**

0: The Transmit Counter register has not reached 0 since the last write in DACC_TCR or DACC_TNCR.

1: The Transmit Counter register has reached 0 since the last write in DACC _TCR or DACC_TNCR.

- **TXBUFE: Transmit Buffer Empty**

0: The Transmit Counter register has not reached 0 since the last write in DACC_TCR or DACC_TNCR.

1: The Transmit Counter register has reached 0 since the last write in DACC _TCR or DACC_TNCR.