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/M4F
Core Size	32-Bit Dual-Core
Speed	120MHz
Connectivity	EBI/EMI, I²C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, LCD, POR, PWM, WDT
Number of I/O	52
Program Memory Size	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 6x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-LQFP
Supplier Device Package	100-LQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsam4cmp16cc-aur

8.1.5.2 Metrology/Coprocessor Core (Core 1) Boot Process

After reset, the Sub-system 1 is hold in reset and with no clock. It is up to the Master Application (Core 0 Application) running on the Core 0 to enable the Sub-system 1. Then the application code can be downloaded into the CM4P1 Boot memory (SRAM1), and CM4P0 can afterwards de-assert the CM4P1 reset line. The secondary processor (CM4P1) always identifies SRAM1 as “Boot memory”.

8.1.5.3 Sub-system 1 Startup Sequence

After the Core 0 is booted from Flash, the Core 0 application must perform the following steps:

1. Enable Core 1 System Clock (Bus and peripherals).
2. Enable Core 1 Clock.
3. Release Core 1 System Reset (Bus and peripherals).
4. Enable SRAM1 and SRAM2 Clock.
5. Copy Core 1 Application from Flash into SRAM1.
6. Release Core 1 Reset.

After Step 6, the Core 1 boots from SRAM1 memory.

Pseudo-code:

```
1- // Enable Coprocessor Bus Master Clock (PMC_SCER.CPBMCK).

2- // Enable Coprocessor Clocks (PMC_SCER.CPCK).
       // Set Coprocessor Clock Prescaler and Source (PMC_MCKR.CPPRES).
       // Choose coprocessor main clock source (PMC_MCKR.CPCSS).

3- // Release coprocessor peripheral reset (RSTC_CPMR.CPEREN).

4- // Enable Core 1 SRAM1 and SRAM2 Memories (PMC_PCER.PID42).

5- // Copy Core 1 application code from Flash into SRAM1.

6- // Release coprocessor reset (RSTC_CPMR.CPROCEN).
```

8.1.5.4 Sub-system 1 Start-up Time

Table 8-4 provides the start-up time of sub-system 1 in terms of the number of clock cycles for different CPU speeds. The figures in this table take into account the time to copy 16 Kbytes of code from Flash into SRAM1 using the ‘memcopy’ function from the standard C library and to release Core 1 reset signal. The start-up time of the device from power-up is not taken into account.

Table 8-4. Sub-system 1 Start-up Time

Core Clock (MHz)	Flash Wait State	Core Clock Cycles	Time
21	0	44122	2.1 ms
42	1	45158	1.07 ms
63	2	46203	735 µs
85	3	47242	55 µs
106	4	48284	455 µs
120	5	49329	411 µs

Table 12-5. Memory Region Shareability Policies

<table>
<tr><th>Address Range</th><th>Memory Region</th><th>Memory Type</th><th>Shareability</th></tr>
<tr><td>0x00000000–0x1FFFFFFF</td><td>Code</td><td>Normal[1]</td><td>–</td></tr>
<tr><td>0x20000000–0x3FFFFFFF</td><td>SRAM</td><td>Normal[1]</td><td>–</td></tr>
<tr><td>0x40000000–0x5FFFFFFF</td><td>Peripheral</td><td>Device[1]</td><td>–</td></tr>
<tr><td>0x60000000–0x7FFFFFFF</td><td rowspan="2">External RAM</td><td rowspan="2">Normal[1]</td><td rowspan="2">–</td></tr>
<tr><td>0x80000000–0x9FFFFFFF</td></tr>
<tr><td>0xA0000000–0xBFFFFFFF</td><td rowspan="2">External device</td><td rowspan="2">Device[1]</td><td>Shareable[1]</td></tr>
<tr><td>0xC0000000–0xDFFFFFFF</td><td>Non-shareable[1]</td></tr>
<tr><td>0xE0000000–0xE00FFFFF</td><td>Private Peripheral Bus</td><td>Strongly-ordered[1]</td><td>Shareable[1]</td></tr>
<tr><td>0xE0100000–0xFFFFFFFF</td><td>Vendor-specific device</td><td>Device[1]</td><td>–</td></tr>
</table>

Notes: 1. See “Memory Regions, Types and Attributes” for more information.

Instruction Prefetch and Branch Prediction

The Cortex-M4 processor:

- 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”.

12.6.11 Floating-point Instructions

The table below shows the floating-point instructions.

These instructions are only available if the FPU is included, and enabled, in the system. See “Enabling the FPU” for information about enabling the floating-point unit.

Table 12-27. Floating-point Instructions

Mnemonic	Description
VABS	Floating-point Absolute
VADD	Floating-point Add
VCMP	Compare two floating-point registers, or one floating-point register and zero
VCMPE	Compare two floating-point registers, or one floating-point register and zero with Invalid Operation check
VCVT	Convert between floating-point and integer
VCVT	Convert between floating-point and fixed point
VCVTR	Convert between floating-point and integer with rounding
VCVTB	Converts half-precision value to single-precision
VCVTT	Converts single-precision register to half-precision
VDIV	Floating-point Divide
VFMA	Floating-point Fused Multiply Accumulate
VFNMA	Floating-point Fused Negate Multiply Accumulate
VFMS	Floating-point Fused Multiply Subtract
VFNMS	Floating-point Fused Negate Multiply Subtract
VLDM	Load Multiple extension registers
VLDR	Loads an extension register from memory
VLMA	Floating-point Multiply Accumulate
VLMS	Floating-point Multiply Subtract
VMOV	Floating-point Move Immediate
VMOV	Floating-point Move Register
VMOV	Copy ARM core register to single precision
VMOV	Copy 2 ARM core registers to 2 single precision
VMOV	Copies between ARM core register to scalar
VMOV	Copies between Scalar to ARM core register
VMRS	Move to ARM core register from floating-point System Register
VMSR	Move to floating-point System Register from ARM Core register
VMUL	Multiply floating-point
VNEG	Floating-point negate
VNMLA	Floating-point multiply and add
VNMLS	Floating-point multiply and subtract
VNMUL	Floating-point multiply
VPOP	Pop extension registers

12.6.11.21 VMUL

Floating-point Multiply.

Syntax

```
VMUL{cond}.F32 {Sd,} Sn, Sm
```

where:

cond	is an optional condition code, see “Conditional Execution”.
Sd	is the destination floating-point value.
Sn, Sm	are the operand floating-point values.

Operation

This instruction:

1. Multiplies two floating-point values.
2. Places the results in the destination register.

Restrictions

There are no restrictions.

Condition Flags

These instructions do not change the flags.

12.6.11.27 VSTM

Floating-point Store Multiple.

Syntax

```
VSTM{mode}{cond}{.size} Rn{!}, list
```

where:

mode	is the addressing mode: - *IA* Increment After. The consecutive addresses start at the address specified in *Rn*. This is the default and can be omitted. - *DB* Decrement Before. The consecutive addresses end just before the address specified in *Rn*.
cond	is an optional condition code, see “Conditional Execution”.
size	is an optional data size specifier. If present, it must be equal to the size in bits, 32 or 64, of the registers in *list*.
Rn	is the base register. The SP can be used
!	is the function that causes the instruction to write a modified value back to *Rn*. Required if mode == DB.
list	is a list of the extension registers to be stored, as a list of consecutively numbered doubleword or singleword registers, separated by commas and surrounded by brackets.

Operation

This instruction:

- Stores multiple extension registers to consecutive memory locations using a base address from an ARM core register.

Restrictions

The restrictions are:

- list must contain at least one register.
  If it contains doubleword registers it must not contain more than 16 registers.
- Use of the PC as *Rn* is deprecated.

Condition Flags

These instructions do not change the flags.

- **HFRDY: Hard Fault Ready**

0: The priority did not permit to set the HardFault handler to the pending state when the floating-point stack frame was allocated.

1: The priority permitted to set the HardFault handler to the pending state when the floating-point stack frame was allocated.

- **THREAD: Thread Mode**

0: The mode was not the Thread Mode when the floating-point stack frame was allocated.

1: The mode was the Thread Mode when the floating-point stack frame was allocated.

- **USER: User Privilege Level**

0: The privilege level was not User when the floating-point stack frame was allocated.

1: The privilege level was User when the floating-point stack frame was allocated.

- **LSPACT: Lazy State Preservation Active**

0: The lazy state preservation is not active.

1: The lazy state preservation is active. The floating-point stack frame has been allocated but saving the state to it has been deferred.

22.5.2 EEFC Flash Command Register

Name: EEFC_FCR

Address: 0x400E0A04 (0), 0x400E0C04 (1)

Access: Write-only

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

- **FCMD: Flash Command**

Value	Name	Description
0x00	GETD	Get Flash descriptor
0x01	WP	Write page
0x02	WPL	Write page and lock
0x03	EWP	Erase page and write page
0x04	EWPL	Erase page and write page then lock
0x05	EA	Erase all
0x06	EPL	Erase plane
0x07	EPA	Erase pages
0x08	SLB	Set lock bit
0x09	CLB	Clear lock bit
0x0A	GLB	Get lock bit
0x0B	SGPB	Set GPNVM bit
0x0C	CGPB	Clear GPNVM bit
0x0D	GGPB	Get GPNVM bit
0x0E	STUI	Start read unique identifier
0x0F	SPUI	Stop read unique identifier
0x10	GCALB	Get CALIB bit
0x11	ES	Erase sector
0x12	WUS	Write user signature
0x13	EUS	Erase user signature
0x14	STUS	Start read user signature
0x15	SPUS	Stop read user signature

26.9.1 Bus Matrix Master Configuration Registers

Name: MATRIX_MCFGx [x=0..6]

Address: 0x400E0200 (0), 0x48010000 (1)

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
–	–	–	–	–	ULBT		

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

- **ULBT: Undefined Length Burst Type**

0: Unlimited Length Burst

No predicted end of burst is generated, therefore INCR bursts coming from this master can only be broken if the Slave Slot Cycle Limit is reached. If the Slot Cycle Limit is not reached, the burst is normally completed by the master, at the latest, on the next AHB 1 KByte address boundary, allowing up to 256-beat word bursts or 128-beat double-word bursts.

This value should not be used in the very particular case of a master capable of performing back-to-back undefined length bursts on a single slave, since this could indefinitely freeze the slave arbitration and thus prevent another master from accessing this slave.

1: Single Access

The undefined length burst is treated as a succession of single accesses, allowing re-arbitration at each beat of the INCR burst or bursts sequence.

2: 4-beat Burst

The undefined length burst or bursts sequence is split into 4-beat bursts or less, allowing re-arbitration every 4 beats.

3: 8-beat Burst

The undefined length burst or bursts sequence is split into 8-beat bursts or less, allowing re-arbitration every 8 beats.

4: 16-beat Burst

The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing re-arbitration every 16 beats.

5: 32-beat Burst

The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing re-arbitration every 32 beats.

6: 64-beat Burst

The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing re-arbitration every 64 beats.

7: 128-beat Burst

The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing re-arbitration every 128 beats.

Unless duly needed, the ULBT should be left at its default 0 value for power saving.

- 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.

- **BAT: Byte Access Type**

This field is used only if DBW defines a 16-bit data bus.

Value	Name	Description
0	BYTE_SELECT	Byte select access type: - Write operation is controlled using NCS, NWE, NBS0, NBS1. - Read operation is controlled using NCS, NRD, NBS0, NBS1.
1	BYTE_WRITE	Byte write access type: - Write operation is controlled using NCS, NWR0, NWR1. - Read operation is controlled using NCS and NRD.

- **DBW: Data Bus Width**

Value	Name	Description
0	8_BIT	8-bit Data Bus
1	16_BIT	16-bit Data Bus

- **TDF_CYCLES: Data Float Time**

This field gives the integer number of clock cycles required by the external device to release the data after the rising edge of the read controlling signal. The SMC always provide one full cycle of bus turnaround after the TDF_CYCLES period. The external bus cannot be used by another chip select during TDF_CYCLES + 1 cycles. From 0 up to 15 TDF_CYCLES can be set.

- **TDF_MODE: TDF Optimization**

0: TDF optimization is disabled.

– The number of TDF wait states is inserted before the next access begins.

1: TDF optimization is enabled.

– The number of TDF wait states is optimized using the setup period of the next read/write access.

- **PMEN: Page Mode Enabled**

0: Standard read is applied.

1: Asynchronous burst read in Page mode is applied on the corresponding chip select.

- **PS: Page Size**

If Page mode is enabled, this field indicates the size of the page in bytes.

Value	Name	Description
0	4_BYTE	4-byte page
1	8_BYTE	8-byte page
2	16_BYTE	16-byte page
3	32_BYTE	32-byte page

30.18.22PMC Write Protection Status Register

Name: PMC_WPSR

Address: 0x400E04E8

Access: Read-only

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

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the PMC_WPSR.

1: A write protection violation has occurred since the last read of the PMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

32.5.14 Register Write Protection

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

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

The WPVS bit is automatically cleared after reading the PIO_WPSR.

The following registers can be write-protected:

- PIO Enable Register
- PIO Disable Register
- PIO Output Enable Register
- PIO Output Disable Register
- PIO Input Filter Enable Register
- PIO Input Filter Disable Register
- PIO Multi-driver Enable Register
- PIO Multi-driver Disable Register
- PIO Pull-Up Disable Register
- PIO Pull-Up Enable Register
- PIO Peripheral ABCD Select Register 1
- PIO Peripheral ABCD Select Register 2
- PIO Output Write Enable Register
- PIO Output Write Disable Register
- PIO Pad Pull-Down Disable Register
- PIO Pad Pull-Down Enable Register

32.6.30 PIO Pad Pull-Down Disable Register

Name: PIO_PPDDR

Address: 0x400E0E90 (PIOA), 0x400E1090 (PIOB), 0x4800C090 (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

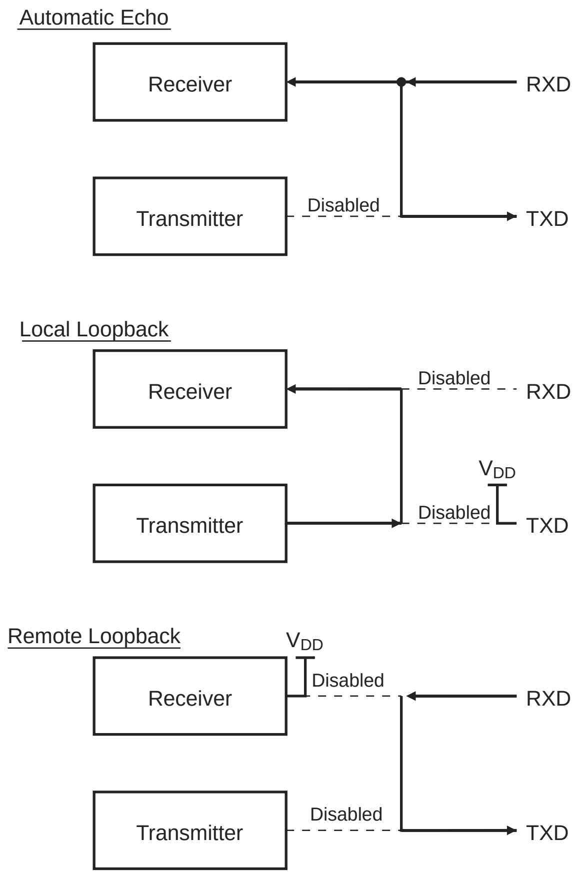
This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

- **P0–P31: Pull-Down Disable**

0: No effect.

1: Disables the pull-down resistor on the I/O line.

Figure 35-14. Test Modes



- **CHMODE: Channel Mode**

Value	Name	Description
0	NORMAL	Normal mode
1	AUTOMATIC	Automatic echo
2	LOCAL_LOOPBACK	Local loopback
3	REMOTE_LOOPBACK	Remote loopback

- **OPT_CLKDIV: Optical Link Clock Divider**

0 to 31: The optical modulation clock frequency is defined by PLLACK / (8 * (OPT_CLKDIV + 8)).

- **OPT_DUTY: Optical Link Modulation Clock Duty Cycle**

Value	Name	Description
0	DUTY_50	Modulation clock duty cycle Is 50%.
1	DUTY_43P75	Modulation clock duty cycle Is 43.75%.
2	DUTY_37P5	Modulation clock duty cycle Is 37.5%.
3	DUTY_31P25	Modulation clock duty cycle Is 31.75%.
4	DUTY_25	Modulation clock duty cycle Is 25%.
5	DUTY_18P75	Modulation clock duty cycle Is 18.75%.
6	DUTY_12P5	Modulation clock duty cycle Is 12.5%.
7	DUTY_6P25	Modulation clock duty cycle Is 6.25%.

- **OPT_CMPTH: Receive Path Comparator Threshold**

Value	Name	Description
0	VDDIO_DIV2	Comparator threshold is VDDIO/2 volts.
1	VDDIO_DIV2P5	Comparator threshold is VDDIO/2.5 volts.
2	VDDIO_DIV3P3	Comparator threshold is VDDIO/3.3 volts.
3	VDDIO_DIV5	Comparator threshold is VDDIO/5 volts.
4	VDDIO_DIV10	Comparator threshold is VDDIO/10 volts.

37.7.8 TC Register C

Name: TC_RCx [x=0..2]

Address: 0x4001001C (0)[0], 0x4001005C (0)[1], 0x4001009C (0)[2], 0x4001401C (1)[0], 0x4001405C (1)[1], 0x4001409C (1)[2]

Access: Read/Write

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

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

- **RC: Register C**

RC contains the Register C value in real time.

IMPORTANT: For 16-bit channels, RC field size is limited to register bits 15:0.

37.7.18 TC QDEC Interrupt Status Register

Name: TC_QISR

Address: 0x400100D4 (0), 0x400140D4 (1)

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
–	–	–	–	–	–	–	DIR

7	6	5	4	3	2	1	0
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: No Index input change since the last read of TC_QISR.

1: The IDX input has changed since the last read of TC_QISR.

- **DIRCHG: Direction Change**

0: No change on rotation direction since the last read of TC_QISR.

1: The rotation direction changed since the last read of TC_QISR.

- **QERR: Quadrature Error**

0: No quadrature error since the last read of TC_QISR.

1: A quadrature error occurred since the last read of TC_QISR.

- **DIR: Direction**

Returns an image of the actual rotation direction.

38.7.11 PWM Channel Period Register

Name: PWM_CPRD[0..3]

Address: 0x48008208 [0], 0x48008228 [1], 0x48008248 [2], 0x48008268 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
CPRD							

23	22	21	20	19	18	17	16
CPRD							

15	14	13	12	11	10	9	8
CPRD							

7	6	5	4	3	2	1	0
CPRD							

Only the first 16 bits (internal channel counter size) are significant.

- **CPRD: Channel Period**

If the waveform is left-aligned, then the output waveform period depends on the counter source clock and can be calculated:

– By using the Master Clock (MCK) divided by an X given prescaler value (with X being 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula will be:

$$\frac{(X \times CPRD)}{MCK}$$

– By using a Master Clock divided by one of both DIVA or DIVB divider, the formula becomes, respectively:

$$\frac{(CPRD \times DIVA)}{MCK} \; or \; \frac{(CPRD \times DIVB)}{MCK}$$

If the waveform is center-aligned, then the output waveform period depends on the counter source clock and can be calculated:

– By using the Master Clock (MCK) divided by an X given prescaler value (with X being 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula will be:

$$\frac{(2 \times X \times CPRD)}{MCK}$$

– By using a Master Clock divided by one of both DIVA or DIVB divider, the formula becomes, respectively:

$$\frac{(2 \times CPRD \times DIVA)}{MCK} \text{ or } \frac{(2 \times CPRD \times DIVB)}{MCK}$$

39.8.8 SLCDC Interrupt Mask Register

Name: SLCDC_IMR

Address: 0x4003C028

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
–	–	–	–	–	DIS	–	ENDFRAME

- **ENDFRAME: End of Frame Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **DIS: SLCDC Disable Completion Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

39.8.9 SLCDC Interrupt Status Register

Name: SLCDC_ISR

Address: 0x4003C02C

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
–	–	–	–	–	DIS	–	ENDFRAME

- **ENDFRAME: End of Frame Interrupt Status**

0: No End of Frame occurred since the last read.

1: End of Frame occurred since the last read.

- **DIS: SLCDC Disable Completion Interrupt Status**

0: The SLCDC is enabled.

1: The SLCDC is disabled.

40.7.12 ADC Temperature Sensor Mode Register

Name: ADC_TEMPMR

Address: 0x40038034

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
–	–	TEMPCMPMOD		–	–	–	TEMPON

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

- **TEMPON: Temperature Sensor ON**

0: The temperature sensor is not enabled.

1: The temperature sensor is enabled and the measurements are triggered.

- **TEMPCMPMOD: Temperature Comparison Mode**

Value	Name	Description
0	LOW	Generates an event when the converted data is lower than the low threshold of the window.
1	HIGH	Generates an event when the converted data is higher than the high threshold of the window.
2	IN	Generates an event when the converted data is in the comparison window.
3	OUT	Generates an event when the converted data is out of the comparison window.