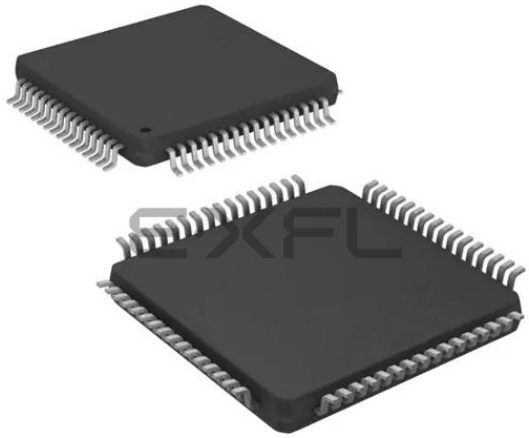


Welcome to E-XFL.COM

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®-M4F
Core Size	32-Bit Single-Core
Speed	120MHz
Connectivity	CANbus, EBI/EMI, I²C, IrDA, LINbus, MMC/SD, QSPI, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM
Number of I/O	51
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	192K x 8
Voltage - Supply (Vcc/Vdd)	1.71V ~ 3.63V
Data Converters	A/D 24x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsame51j19a-au

10.1.2 Integrated Configurable Debug

The Cortex-M4 processor implements a complete hardware debug solution. This provides high system visibility of the processor and memory through a 2-pin *Serial Wire Debug* (SWD) port that is ideal for microcontrollers and other small package devices.

For system trace the processor integrates an *Instrumentation Trace Macrocell* (ITM) alongside data watchpoints and a profiling unit. The *Embedded Trace Macrocell* (ETM) delivers unrivaled instruction trace capture in an area far smaller than traditional trace units, enabling many low cost MCUs to implement full instruction trace for the first time.

To enable simple and cost-effective profiling of the system events these generate, a stream of software-generated messages, data trace, and profiling information is exported over three different ways:

- Output off chip using the TPIU, through a single pin, called *Serial Wire Viewer* (SWV). Limited to ITM system trace
- Output off chip using the TPIU, through a 4-bit pin interface. Bandwidth is limited
- Internally stored in RAM, using the CoreSight ETB. Bandwidth is then optimal but capacity is limited

The *Flash Patch and Breakpoint Unit* (FPB) provides up to 8 hardware breakpoint comparators that debuggers can use. The comparators in the FPB also provide remap functions of up to 8 words in the program code in the CODE memory region. This enables applications stored on a non-erasable, ROM-based microcontroller to be patched if a small programmable memory, for example flash, is available in the device. During initialization, the application in ROM detects, from the programmable memory, whether a patch is required. If a patch is required, the application programs the FPB to remap a number of addresses. When those addresses are accessed, the accesses are redirected to a remap table specified in the FPB configuration, which means the program in the non-modifiable ROM can be patched.

10.1.3 Cortex-M4 Processor Features and Configuration

- Thumb® instruction set combines high code density with 32-bit performance
- IEEE 754-compliant single-precision Floating Point Unit (FPU)
- Integrated sleep modes for low power consumption
- Fast code execution permits slower processor clock or increases Sleep mode time
- Hardware division and fast digital-signal-processing orientated multiply accumulate
- Saturating arithmetic for signal processing
- Deterministic, high-performance interrupt handling for time-critical applications
- Memory Protection Unit (MPU) for safety-critical applications
- Extensive debug and trace capabilities: Serial Wire Debug and Serial Wire Trace reduce the number of pins required for debugging, tracing, and code profiling.

Features	Cortex-M4 Options	SAM D5x/E5x Configuration
Interrupts	1 to 240	138
Number of priority bits	3 to 8	3 = eight levels of priority
Data endianness	Little-endian or big-endian	Little-endian
SysTick Timer calibration value		0x80000000
MPU	Present or Not present	Present

This bit is never cleared.

Figure 21-7. Edge Detection with Synchronous Stability Debouncing

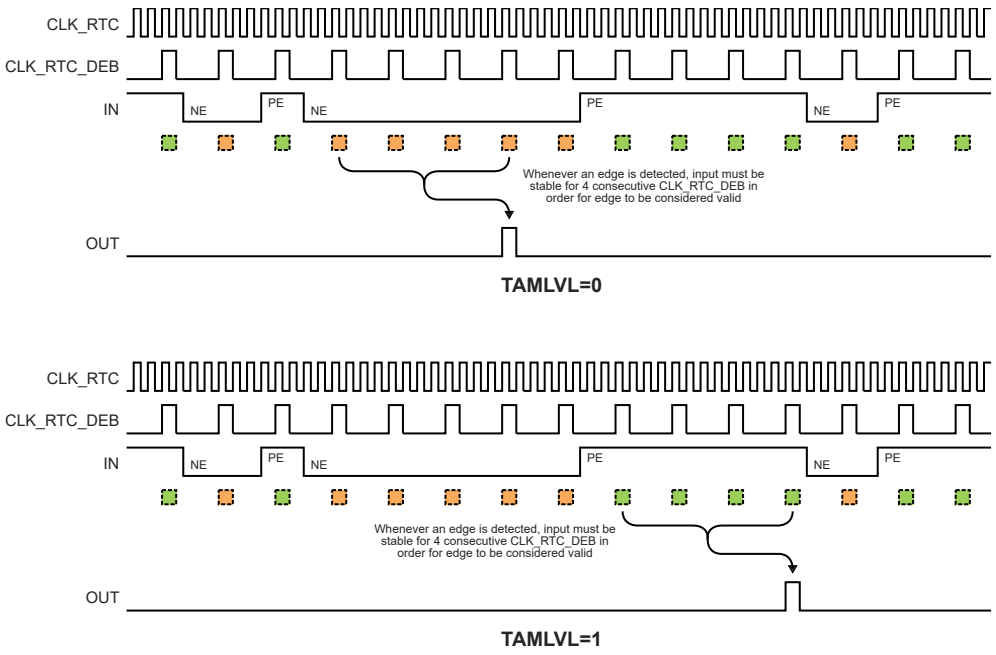
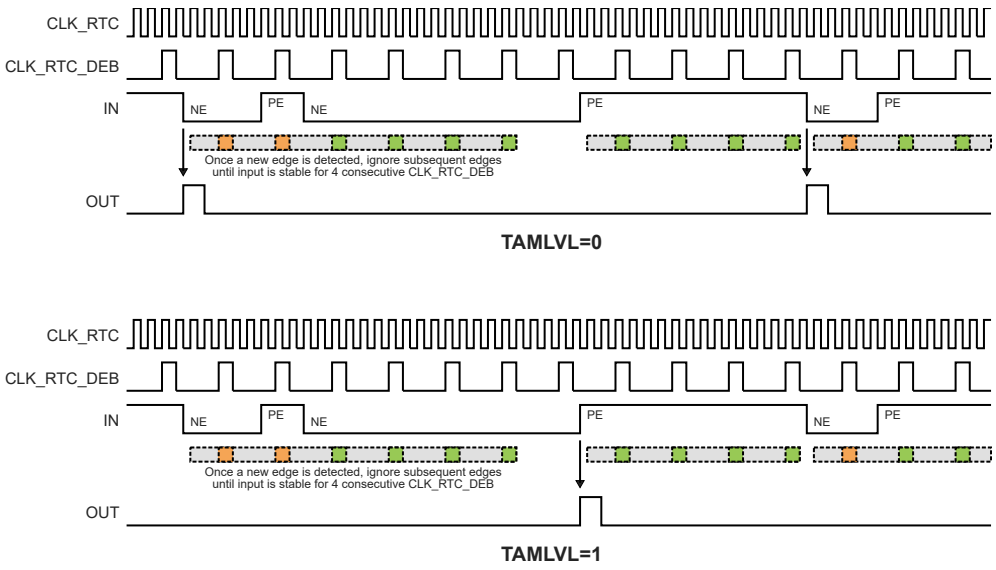


Figure 21-8. Edge Detection with Asynchronous Stability Debouncing



21.7 Register Summary - Mode 0 - 32-Bit Counter

Offset	Name	Bit Pos.								
0x00	CTRLA	7:0	MATCHCLR				MODE[1:0]		ENABLE	SWRST
		15:8	COUNTSYNC	GPTRST			PRESCALER[3:0]			
0x02	CTRLB	7:0	DMAEN	RTCOUT	DEBASYNC	DEBMAJ			GP2EN	GP0EN
		15:8		ACTF[2:0]				DEBF[2:0]		
0x04	EVCTRL	7:0	PEREOn[7:0]							
		15:8	OVFEO	TAMPEREO					CMPEOn[1:0]	
		23:16								TAMPEVEI
		31:24								
0x08	INTENCLR	7:0	PERn[7:0]							
		15:8	OVF	TAMPER					CMPn[1:0]	
0x0A	INTENSET	7:0	PERn[7:0]							
		15:8	OVF	TAMPER					CMPn[1:0]	
0x0C	INTFLAG	7:0	PERn[7:0]							
		15:8	OVF	TAMPER					CMPn[1:0]	
0x0E	DBGCTRL	7:0								DBGRUN
0x0F	Reserved									
0x10	SYNCBUSY	7:0		COMPn[1:0]			COUNT	FREQCORR	ENABLE	SWRST
		15:8	COUNTSYNC							
		23:16					GPn[3:0]			
		31:24								
0x14	FREQCORR	7:0	SIGN	VALUE[6:0]						
0x15 ... 0x17	Reserved									
0x18	COUNT	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16	COUNT[23:16]							
		31:24	COUNT[31:24]							
0x1C ... 0x1F	Reserved									
0x20	COMP0	7:0	COMP[7:0]							
		15:8	COMP[15:8]							
		23:16	COMP[23:16]							
		31:24	COMP[31:24]							
0x24	COMP1	7:0	COMP[7:0]							
		15:8	COMP[15:8]							
		23:16	COMP[23:16]							
		31:24	COMP[31:24]							
0x28 ... 0x3F	Reserved									
0x40	GP0	7:0	GP[7:0]							

22.8.13 Active Channel and Levels

Name: ACTIVE
Offset: 0x30
Reset: 0x00000000
Property: -

Bit	31	30	29	28	27	26	25	24
	BTCNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BTCNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ABUSY			ID[4:0]				
Access	R			R	R	R	R	R
Reset	0			0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					LVLEXx	LVLEXx	LVLEXx	LVLEXx
Access					R	R	R	R
Reset					0	0	0	0

Bits 31:16 – BTCNT[15:0] Active Channel Block Transfer Count
These bits hold the 16-bit block transfer count of the ongoing transfer. This value is stored in the active channel and written back in the corresponding Write-Back channel memory location when the arbiter grants a new channel access. The value is valid only when the active channel active busy flag (ABUSY) is set.

Bit 15 – ABUSY Active Channel Busy
This bit is cleared when the active transfer count is written back in the write-back memory section.

This bit is set when the next descriptor transfer count is read from the write-back memory section.

Bits 12:8 – ID[4:0] Active Channel ID
These bits hold the channel index currently stored in the active channel registers. The value is updated each time the arbiter grants a new channel transfer access request.

Bits 3,2,1,0 – LVLEXx Level x Channel Trigger Request Executing [x=3..0]
This bit is set when a level-x channel trigger request is executing or pending.

This bit is cleared when no request is pending or being executed.

31. EVSYS – Event System

24.6 Functional Description

24.6.1 Media Access Controller

The Transmit Block of the Media Access Controller (MAC) takes data from FIFO, adds preamble, checks and adds padding and frame check sequence (FCS). Both half duplex and full duplex Ethernet modes of operation are supported.

When operating in half duplex mode, the MAC Transmit Block generates data according to the Carrier Sense Multiple Access with Collision Detect (CSMA/CD) protocol. The start of transmission is deferred if Carrier Sense (CRS) is active. If Collision (COL) is detected during transmission, a jam sequence is asserted and the transmission is retried after a random back off. The CRS and COL signals have no effect in full duplex mode.

The Receive Block of the MAC checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC address checking block and FIFO. Software can configure the GMAC to receive jumbo frames of up to 10240 Bytes. It can optionally strip CRC (Cyclic Redundancy Check) from the received frame before transferring it to FIFO.

The Address Checker recognizes four specific 48-bit addresses, can recognize four different types of ID values, and contains a 64-bit Hash register for matching multicast and unicast addresses as required. It can recognize the broadcast address *all-'1'* (0xFFFFFFFFFFFF) and copy all frames. The MAC can also reject all frames that are not VLAN tagged, and recognize Wake on LAN events.

The MAC Receive Block supports offloading of IP, TCP and UDP checksum calculations (both IPv4 and IPv6 packet types supported), and can automatically discard bad checksum frames.

24.6.2 IEEE 1588 Time Stamp Unit

The IEEE 1588 time stamp unit (TSU) is implemented as a 94-bit timer.

- The 48 upper bits [93:46] of the timer count seconds and are accessible in the GMAC 1588 Timer Seconds High Register" (TSH) and GMAC 1588 Timer Seconds Low Register (TSL).
- The 30 lower bits [45:16] of the timer count nanoseconds and are accessible in the GMAC 1588 Timer Nanoseconds Register (TN).
- The lowest 16 bits [15:0] of the timer count sub-nanoseconds.

The 46 lower bits roll over when they have counted to 1s. The timer increments by a programmable period (to approximately 15.2fs resolution) with each MCK period and can also be adjusted in 1ns resolution (incremented or decremented) through APB register accesses.

24.6.3 AHB Direct Memory Access Interface

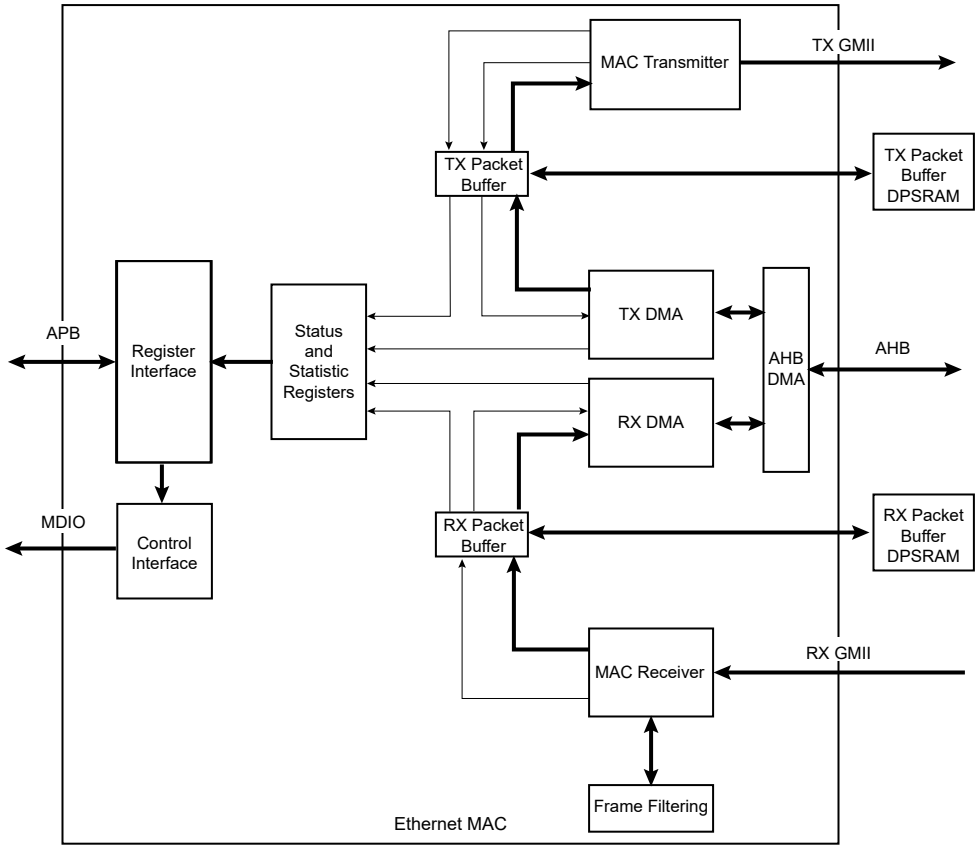
The GMAC DMA controller is connected to the MAC FIFO interface and provides a scatter-gather type capability for packet data storage.

The DMA implements packet buffering where dual-port memories are used to buffer multiple frames.

24.6.3.1 Packet Buffer DMA

- Easier to guarantee maximum line rate due to the ability to store multiple frames in the packet buffer, where the number of frames is limited by the amount of packet buffer memory and Ethernet frame size
- Full store and forward, or partial store and forward programmable options (partial store will cater for shorter latency requirements)

Figure 24-2. Data Paths with Packet Buffers Included



24.6.3.7 Transmit Packet Buffer

The transmitter packet buffer will continue attempting to fetch frame data from the AHB system memory until the packet buffer itself is full, at which point it will attempt to maintain its full level.

To accommodate the status and statistics associated with each frame, three words per packet (or two if the GMAC is configured in 64-bit data path mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the DPRAM is required in order to decouple the DMA interface of the buffer from the MAC interface, to update the MAC status/statistics and to generate interrupts in the order in which the packets that they represent were fetched from the AHB memory.

If any errors occur on the AHB while reading the transmit frame, the fetching of packet data from AHB memory is halted. The MAC transmitter will continue to fetch packet data, thereby emptying the packet buffer and allowing any good (non-erroneous) frames to be transmitted successfully. Once these have been fully transmitted, the status/statistics for the erroneous frame will be updated and software will be informed via an interrupt that an AHB error occurred. This way, the error is reported in the correct packet order.

The transmit packet buffer will only attempt to read more frame data from the AHB when space is available in the packet buffer memory. If space is not available it must wait until the a packet fetched by the MAC completes transmission and is subsequently removed from the packet buffer memory.
Note: If full store and forward mode is active and if a single frame is fetched that is too large for the packet buffer memory, the frame is flushed and the DMA halted with an error status. This is because a complete frame must be written into the packet buffer before transmission can begin, and therefore the

31.7.13 Event User m

Name: USERm
Offset: 0x0120 + m*0x01 [m=0..66]
Reset: 0x0
Property: PAC Write-Protection

Bit	7	6	5	4	3	2	1	0
	CHANNEL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – CHANNEL[7:0] Channel Event Selection
These bits select channel n to connect to the event user m.

Note: A value x of this bit field selects channel n = x-1.

Table 31-2. User Multiplexer Number m

USER*m*	User Multiplexer	Description	Path Type(1)
m = 0	RTC_TAMPER	RTC Tamper	A
m = 1..4	PORT_EV0..3	PORT Event 0..3	A
m = 5..12	DMAC_CH0..7	Channel 0..7	S, R
m = 13	-	Reserved	-
m = 14	CM4_TRACE_START	CM4 trace start	S, R
m = 15	CM4_TRACE_STOP	CM4 trace stop	S, R
m = 16	CM4_TRACE_TRIG	CM4 trace trigger	S, R
m = 17..18	TCC0 EV0..1	TCC0 EVx	A, S, R
m = 19..24	TCC0 MC0..5	TCC0 MCx	A, S, R
m = 25..26	TCC1 EV0..1	TCC1 EVx	A, S, R
m = 27..30	TCC1 MC0..3	TCC1 MCx	A, S, R
m = 31..32	TCC2 EV0..1	TCC2 EVx	A, S, R
m = 33..35	TCC2 MC0..2	TCC2 MCx	A, S, R
m = 36..37	TCC3 EV0..1	TCC3 EVx	A, S, R
m = 38..39	TCC3 MC0..1	TCC3 MCx	A, S, R
m = 40..41	TCC4 EV0..1	TCC4 EVx	A, S, R
m = 42..43	TCC4 MC0..1	TCC4 MCx	A, S, R
m = 44..51	TC0..7 EVU	TC0..7 EVU	A, S, R
m = 52..54	PDEC_EVU 0..2	PDEC EVU x	A, S, R

34.8.9 Status

Name: STATUS
Offset: 0x1A
Reset: 0x0000
Property: -

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	ITER	TXE	COLL	ISF	CTS	BUFOVF	FERR	PERR
Access	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – ITER Maximum Number of Repetitions Reached
This bit is set when the maximum number of NACK repetitions or retransmissions is met in ISO7816 T=0 mode.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

Bit 6 – TXE Transmitter Empty
When CTRLA.FORM is set to LIN master mode, this bit is set when any ongoing transmission is complete and TxDATA is empty.

When CTRLA.FORM is not set to LIN master mode, this bit will always read back as zero.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

Bit 5 – COLL Collision Detected
This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when collision detection is enabled (CTRLB.COLDEN) and a collision is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

Bit 4 – ISF Inconsistent Sync Field
This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when the frame format is set to auto-baud (CTRLA.FORM) and a sync field not equal to 0x55 is received.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

Bit 0 – BUSERR Bus Error

This bit indicates that an illegal bus condition has occurred on the bus, regardless of bus ownership. An illegal bus condition is detected if a protocol violating start, repeated start or stop is detected on the I^2C bus lines. A start condition directly followed by a stop condition is one example of a protocol violation. If a time-out occurs during a frame, this is also considered a protocol violation, and will set BUSERR.

If the I^2C master is the bus owner at the time a bus error occurs, STATUS.ARBLOST and INTFLAG.MB will be set in addition to BUSERR.

Writing the ADDR.ADDR register will automatically clear the BUSERR flag.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

This bit is not write-synchronized.

Table 39-4. Example Filter Configuration for Rx Buffers

Filter Element	SFID1[10:0] / EFID1[28:0]	SFID2[10:9] / EFID2[10:9]	SFID2[5:0] / EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register NDAT1, NDAT2 is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the CPU by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

Rx Buffer Handling

- Reset interrupt flag IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

39.6.5.4 Debug on CAN Support

Debug messages are stored into Rx Buffers. For debug handling three consecutive Rx buffers (e.g. #61, #62, #63) have to be used for storage of debug messages A, B, and C. The format is the same as for an Rx Buffer or an Rx FIFO element (see 39.9.2 Rx Buffer and FIFO Element).

Advantage: Fixed start address for the DMA transfers (relative to RXBC.RBSA), no additional configuration required.

For filtering of debug messages Standard / Extended Filter Elements with SFEC / EFEC = "111" have to be set up. Messages matching these filter elements are stored into the Rx Buffers addressed by SFID2 / EFID2[5:0].

After message C has been stored, the DMA request output is activated and the three messages can be read from the Message RAM under DMA control. The RAM words holding the debug messages will not be changed by the CAN while DMA request is activated. The behavior is similar to that of an Rx Buffers with its New Data flag set.

After the DMA has completed the DMA unit sets the DMA acknowledge. This resets DMA request. Now the CAN is prepared to receive the next set of debug messages.

Filtering for Debug Messages

Filtering for debug messages is done by configuring one Standard / Extended Message ID Filter Element for each of the three debug messages. To enable a filter element to filter for debug messages SFEC / EFEC has to be programmed to "111". In this case fields SFID1 / SFID2 and EFID1 / EFID2 have a different meaning (see 39.9.5 Standard Message ID Filter Element and 39.9.6 Extended Message ID Filter Element). While SFID2 / EFID2[10:9] controls the debug message handling state machine, SFID2 / EFID2[5:0] controls the location for storage of a received debug message.

39.8.11 Timeout Counter Configuration

Name: TOCC
Offset: 0x28
Reset: 0xFFFF0000
Property: Write-restricted

This register is write-restricted and only writable if bit fields CCCR.CCE = 1 and CCCR.INIT = 1.

Bit	31	30	29	28	27	26	25	24
	TOP[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	TOP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						TOS[1:0]		ETOC
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:16 – TOP[15:0] Timeout Period
Start value of the Timeout Counter (down-counter). Configures the Timeout Period.

Bits 2:1 – TOS[1:0] Timeout Select
When operating in Continuous mode, a write to TOCV presets the counter to the value configured by TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by TOCC.TOP. Down-counting is started when the first FIFO element is stored.

Value	Name	Description
0x0	CONT	Continuous operation.
0x1	TXEF	Timeout controlled by TX Event FIFO.
0x2	RXF0	Timeout controlled by Rx FIFO 0.
0x3	RXF1	Timeout controlled by Rx FIFO 1.

Bit 0 – ETOC Enable Timeout Counter

Value	Description
0	Timeout Counter disabled.
1	Timeout Counter enabled.

40.8.24 Host Control 2 Register: e.MMC

Name: HC2R
Offset: 0x3E
Reset: 0x0000
Property: -

Note: The content of the HC2R register is depending on the mode. This description is for e.MMC mode. For SD/SDIO mode, see 40.8.25 HC2R.

Bit	15	14	13	12	11	10	9	8
	PVALEN							
Access	R/W							
Reset	0							

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 15 – PVALEN Preset Value Enable
As the operating SDCLK frequency depends on the system implementation, it is difficult to determine these parameters in the standard host driver. When Preset Value Enable (PVALEN) is set to 1, automatic SDCLK frequency generationis performed without considering system-specific conditions. This bit enables the functions defined in PVR.

If this bit is written to 0, the Clock Generator Select bit (CCR.CLKGSEL) and the SDCLK Frequency Select bit (CCR.SDCLKFSEL) in the Clock Control Register (CCR) are selected by the user.

If this bit is set to 1, CCR.SDCLKFSEL and .CLKGSEL and HC2R.DRVSEL are set by the peripheral as specified in the Preset Value Register (PVR).

Value	Description
0	CCR.SDCLK, CCR.SDCLKFSEL controlled by the user.
1	Automatic selection by Preset Value is enabled.

43.3.6.3.4 Parameters Definition

Table 43-70. ZpEccAddSubFast Service Parameters

Parameter	Type	Direction	Location	Data Length	Before Executing the Service	After Executing the Service
nu1ModBase	nu1	I	Crypto RAM	u2ModLength + 4	Base of Modulus P	Base of Modulus P
nu1CnsBase	nu1	I	Crypto RAM	u2ModLength + 8	Base of Cns	Base of Cns
u2ModLength	u2	I	–	–	Length of modulo	Length of modulo
nu1PointABase	nu1	I/O	Crypto RAM	3*u2ModLength + 12	Input point A (projective coordinates)	Resulting point C (projective coordinates)
nu1PointBBase	nu1	I	Crypto RAM	3*u2ModLength + 12	Input point B (projective coordinates)	Input point B
u2Operator	u2	I	-	-	Addition or Subtraction	Addition or Subtraction
nu1Workspace	nu1	I	Crypto RAM	5*u2ModLength + 32	–	Corrupted workspace

43.3.6.3.5 Code Example

```
PUKCL_PARAM PUKCLParam;
PPUKCL_PARAM pvPUKCLParam = &PUKCLParam;


PUKCL (u2Option) = 0;

PUKCL _ZpEccAddSub(nu1ModBase) = <Base of the ram location of P>;
PUKCL _ZpEccAddSub(nu1CnsBase) = <Base of the ram location of Cns>;
PUKCL _ZpEccAddSub(u2ModLength) = <Byte length of P>;
PUKCL _ZpEccAddSub(nu1PointABase) = <Base of the ram location of the A point>;
PUKCL _ZpEccAddSub(nu1PointBBase) = <Base of the ram location of the B point>;
PUKCL _ZpEccAddSub(nu1Workspace) = <Base of the ram location of the workspace>;
PUKCL _ZpEccAddSub(u2Operator) = <Operation to perform (PUKCL_ZPECCADD or PUKCL_ZPECCSUB)>;
...

// vPUKCL_Process() is a macro command, which populates the service name
// and then calls the library...
vPUKCL_Process(ZpEccAddSubFast,&PUKCLParam);
if (PUKCL (u2Status) == PUKCL_OK)
        {
        ...
        }
else // Manage the error
```

43.3.6.3.6 Constraints

No overlapping between either input and output are allowed. The following conditions must be avoided to ensure that the service works correctly:

- nu1ModBase, nu1CnsBase, nu1PointABase, nu1PointBBase, nu1Workspace are not aligned on 32-bit boundaries

Parameter	Type	Direction	Location	Data Length	Before Executing the Service	After Executing the Service
u2ScalarLength	u2	I	–	–	Length of scalar (same length as the length of order)	Length of scalar
nu1PointABase (see **Note 2**)	nu1	I/O	Crypto RAM	3*u2ModLength + 12	Input point A (three coordinates (X,Y) affine and Z = 1)	Resulting signature (R,S,0)
nu1ABase	nu1	I	Crypto RAM	u2ModLength + 4	Parameter a of the elliptic curve	Unchanged
nu1Workspace	nu1	I	Crypto RAM	8*u2ModLength + 44	–	Corrupted workspace

Note:

1. The hash value calculus is defined by the ECDSA norm and depends on the elliptic curve domain parameters. To construct the input parameter, the 4 Most Significant Bytes must be set to zero.
2. The resulting signature format is different from the point A format (see Description above for information on the point A format).

43.3.6.11.5 Code Example

```
PUKCL_PARAM PUKCLParam;
PPUKCL_PARAM pvPUKCLParam = &PUKCLParam;

// ! The Random Number Generator must be initialized and started
// ! following the directives given for the RNG on the chip


PUKCL (u2Option) = 0;

// Depending on the option specified, not all fields should be filled PUKCL
_ZpEcDsaGenerate(nu1ModBase) = <Base of the ram location of P>; PUKCL
_ZpEcDsaGenerate(u2ModLength) = <Byte length of P>;
PUKCL _ZpEcDsaGenerate(nu1CnsBase) = <Base of the ram location of Cns>;
PUKCL _ZpEcDsaGenerate(nu1PointABase) = <Base of the A point>;
PUKCL _ZpEcDsaGenerate(nu1PrivateKey) = <Base of the Private Key>;
PUKCL _ZpEcDsaGenerate(nu1ScalarNumber) = <Base of the ScalarNumber>;
PUKCL _ZpEcDsaGenerate(nu1OrderPointBase) = <Base of the order of A point>;
PUKCL _ZpEcDsaGenerate(nu1ABase) = <Base of the a parameter of the curve>;
PUKCL _ZpEcDsaGenerate(nu1Workspace) = <Base of the workspace>;
PUKCL _ZpEcDsaGenerate(nu1HashBase) = <Base of the SHA resulting hash>;
PUKCL_ZpEcDsaGenerate(u2ScalarLength)    = < Length of ScalarNumber>;
...

// vPUKCL_Process() is a macro command, which populates the service name
// and then calls the library...
vPUKCL_Process(ZpEcDsaGenerateFast, pvPUKCLParam);
if (PUKCL (u2Status) == PUKCL_OK)
            {
            ...
            }
else // Manage the error
```

45.8.9 Window Monitor Lower Threshold

Name: WINLT
Offset: 0x0C
Reset: 0x0000
Property: PAC Write-Protection, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	WINLT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WINLT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – WINLT[15:0] Window Lower Threshold
If the window monitor is enabled, these bits define the lower threshold value.

47.8.8 Synchronization Busy

Name: SYNCBUSY
Offset: 0x08
Reset: 0x00000000
Property: -

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			DATABUF1	DATABUF0	DATA1	DATA0	ENABLE	SWRST
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – DATABUF1 Data Buffer DAC1
This bit is set when DATABUF1 register is written.

This bit is cleared when DATABUF1 synchronization is completed.

Value	Description
0	No ongoing synchronized access.
1	Synchronized access is ongoing.

Bit 4 – DATABUF0 Data Buffer DAC0
This bit is set when DATABUF0 register is written.

This bit is cleared when DATABUF0 synchronization is completed.

Value	Description
0	No ongoing synchronized access.
1	Synchronized access is ongoing.

Bit 3 – DATA1 Data DAC1
This bit is set when DATA1 register is written.

This bit is cleared when DATA1 synchronization is completed.

47.8.11 Data DAC0

Name: DATA0
Offset: 0x10
Reset: 0x0000
Property: PAC Write-Protection, Write-Synchronized

Bit	15	14	13	12	11	10	9	8
	DATA[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – DATA[15:0] DAC0 Data
DATA0 register contains the 12-bit value that is converted to a voltage by the DAC0. The adjustment of these 12 bits within the 16-bit register is controlled by DACCTRL0.LEFTADJ:

- DATA[11:0] when DACCTRL0.LEFTADJ=0.

- DATA[15:4] when DACCTRL0.LEFTADJ=1.

In dithering mode (whatever DACCTRL0.LEFTADJ value):

- DATA[15:4] are the 12-bit converted by DAC0.

- DATA[3:0] are the dither bits.

Bits 2:0 – WAVEGEN[2:0] Waveform Generation Operation
These bits select the waveform generation operation. The settings impact the top value and control if frequency or PWM waveform generation should be used. These bits are not synchronized.

Value	Name	Description						
		Operation	Top	Update	Waveform Output On Match	Waveform Output On Update	OVFIF/Event Up	Down
0x0	NFRQ	Normal Frequency	PER	TOP/Zero	Toggle	Stable	TOP	Zero
0x1	MFRQ	Match Frequency	CC0	TOP/Zero	Toggle	Stable	TOP	Zero
0x2	NPWM	Normal PWM	PER	TOP/Zero	Set	Clear	TOP	Zero
0x3	Reserved	-	-	-	-	-	-	-
0x4	DSCRITICAL	Dual-slope PWM	PER	Zero	~DIR	Stable	–	Zero
0x5	DSBOTTOM	Dual-slope PWM	PER	Zero	~DIR	Stable	–	Zero
0x6	DSBOTH	Dual-slope PWM	PER	TOP & Zero	~DIR	Stable	TOP	Zero
0x7	DSTOP	Dual-slope PWM	PER	Zero	~DIR	Stable	TOP	–

CTRLA.RESOLUTION	Bits [23:m]
0x0 - NONE	23:0
0x1 - DITH4	23:4
0x2 - DITH5	23:5
0x3 - DITH6	23:6 (depicted)

Bits 5:0 – DITHER[5:0] Dithering Cycle Number
These bits hold the number of extra cycles that are added on the PWM pulse width every 64 PWM frames.
Note: This bit field consists of the n LSB of the register. n is dependent on the value of the Resolution bits in the Control A register (CTRLA.RESOLUTION):

CTRLA.RESOLUTION	Bits [n:0]
0x0 - NONE	-
0x1 - DITH4	3:0
0x2 - DITH5	4:0
0x3 - DITH6	5:0 (depicted)