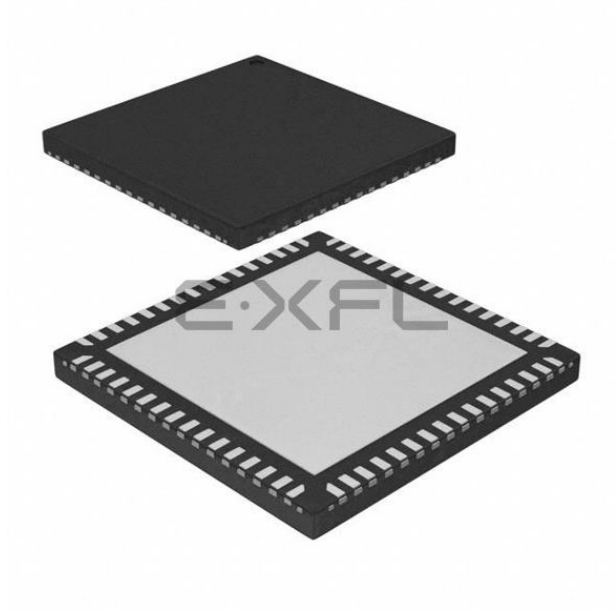


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

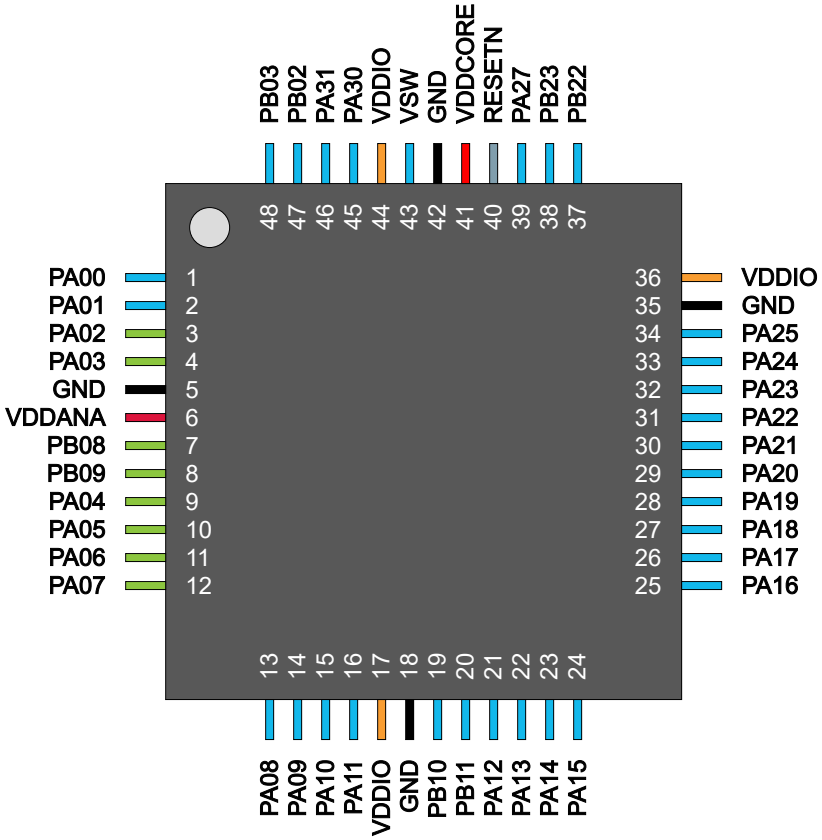
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	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	256KB (256K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K 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-VFQFN Exposed Pad
Supplier Device Package	64-QFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsamd51j18a-mu

4. Pinout

4.1 Pin Count 48 (G)



11.10.5 Cache Lock per Way

Name: LCKWAY
Offset: 0x10
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
					LCKWAY[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 3:0 – LCKWAY[3:0] Lockdown Way Register
This field selects which way is locked.

Offset	Name	Bit Pos.								
		15:8	BKUP[15:8]							
		23:16	BKUP[23:16]							
		31:24	BKUP[31:24]							
0x90	BKUP4	7:0	BKUP[7:0]							
		15:8	BKUP[15:8]							
		23:16	BKUP[23:16]							
		31:24	BKUP[31:24]							
0x94	BKUP5	7:0	BKUP[7:0]							
		15:8	BKUP[15:8]							
		23:16	BKUP[23:16]							
		31:24	BKUP[31:24]							
0x98	BKUP6	7:0	BKUP[7:0]							
		15:8	BKUP[15:8]							
		23:16	BKUP[23:16]							
		31:24	BKUP[31:24]							
0x9C	BKUP7	7:0	BKUP[7:0]							
		15:8	BKUP[15:8]							
		23:16	BKUP[23:16]							
		31:24	BKUP[31:24]							

21.8 Register Description - Mode 0 - 32-Bit Counter

This Register Description section is valid if the RTC is in COUNT32 mode (CTRLA.MODE=0).

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

Bit 10 – SRCINC Source Address Increment Enable
Writing a '0' to this bit will disable the source address incrementation. The address will be kept fixed during the data transfer.

Writing a '1' to this bit will enable the source address incrementation. By default, the source address is incremented by 1. If the STEPSEL bit is set, flexible step-size settings are available in the STEPSIZE register.

Value	Description
0	The Source Address Increment is disabled.
1	The Source Address Increment is enabled.

Bits 9:8 – BEATSIZE[1:0] Beat Size
These bits define the size of one beat. A beat is the size of one data transfer bus access, and the setting apply to both read and write accesses.

Value	Name	Description
0x0	BYTE	8-bit bus transfer
0x1	HWORD	16-bit bus transfer
0x2	WORD	32-bit bus transfer
other		Reserved

Bits 4:3 – BLOCKACT[1:0] Block Action
These bits define what actions the DMAC should take after a block transfer has completed.

BLOCKACT[1:0]	Name	Description
0x0	NOACT	Channel will be disabled if it is the last block transfer in the transaction
0x1	INT	Channel will be disabled if it is the last block transfer in the transaction and block interrupt
0x2	SUSPEND	Channel suspend operation is completed
0x3	BOTH	Both channel suspend operation and block interrupt

Bits 2:1 – EVOSEL[1:0] Event Output Selection
These bits define the event output selection.

EVOSEL[1:0]	Name	Description
0x0	DISABLE	Event generation disabled
0x1	BLOCK	Event strobe when block transfer complete
0x2		Reserved
0x3	BEAT	Event strobe when beat transfer complete

Bit 0 – VALID Descriptor Valid
Writing a '0' to this bit in the Descriptor or Write-Back memory will suspend the DMA channel operation when fetching the corresponding descriptor.

The bit is automatically cleared in the Write-Back memory section when channel is aborted, when an error is detected during the block transfer, or when the block transfer is completed.

Offset	Name	Bit Pos.								
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0124	TBFT511	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0128	TBFT1023	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x012C	TBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0130	GTBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0134	TUR	7:0	TXUNR[7:0]							
		15:8							TXUNR[9:8]	
		23:16								
		31:24								
0x0138	SCF	7:0	SCOL[7:0]							
		15:8	SCOL[15:8]							
		23:16							SCOL[17:16]	
		31:24								
0x013C	MCF	7:0	MCOL[7:0]							
		15:8	MCOL[15:8]							
		23:16							MCOL[17:16]	
		31:24								
0x0140	EC	7:0	XCOL[7:0]							
		15:8							XCOL[9:8]	
		23:16								
		31:24								
0x0144	LC	7:0	LCOL[7:0]							
		15:8							LCOL[9:8]	
		23:16								
		31:24								
0x0148	DTF	7:0	DEFT[7:0]							
		15:8	DEFT[15:8]							
		23:16							DEFT[17:16]	
		31:24								
0x014C	CSE	7:0	CSR[7:0]							
		15:8							CSR[9:8]	
		23:16								
		31:24								

24.9.67 GMAC 256 to 511 Byte Frames Received Register

Name: TBFR511
Offset: 0x174
Reset: 0x00000000
Property: Read-Only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
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
	NFRX[23:16]							
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
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 256 to 511 Byte Frames Received without Error
This bit fields counts the number of 256 to 511 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

27.7.3 Interrupt Enable Clear

Name: INTENCLR
Offset: 0x08
Reset: 0x00
Property: PAC Write-Protection

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

Bit	7	6	5	4	3	2	1	0
								ERR
Access								RW
Reset								0

Bit 0 – ERR Peripheral Access Error Interrupt Disable
This bit indicates that the Peripheral Access Error Interrupt is disabled and an interrupt request will be generated when one of the interrupt flag registers bits (INTFLAGAHB, INTFLAGn) is set:

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the Peripheral Access Error interrupt Enable bit and disables the corresponding interrupt request.

Value	Description
0	Peripheral Access Error interrupt is disabled.
1	Peripheral Access Error interrupt is enabled.

Value	Description
0	TC2 peripheral is not write protected
1	TC2 peripheral is write protected

Bit 12 – TCC1 TCC1 APB Protect Enable

Value	Description
0	TCC1 peripheral is not write protected
1	TCC1 peripheral is write protected

Bit 11 – TCC0 TCC0 APB Protect Enable

Value	Description
0	TCC0 peripheral is not write protected
1	TCC0 peripheral is write protected

Bit 10 – SERCOM3 SERCOM3 APB Protect Enable

Value	Description
0	SERCOM3 peripheral is not write protected
1	SERCOM3 peripheral is write protected

Bit 9 – SERCOM2 SERCOM2 APB Protect Enable

Value	Description
0	SERCOM2 peripheral is not write protected
1	SERCOM2 peripheral is write protected

Bit 7 – EVSYS EVSYS APB Protect Enable

Value	Description
0	EVSYS peripheral is not write protected
1	EVSYS peripheral is write protected

Bit 5 – DMAC DMAC APB Protect Enable

Value	Description
0	DMAC peripheral is not write protected
1	DMAC peripheral is write protected

Bit 4 – PORT PORT APB Protect Enable

Value	Description
0	PORT peripheral is not write protected
1	PORT peripheral is write protected

Bit 3 – CMCC CMCC APB Protect Enable

Value	Description
0	CMCC peripheral is not write protected
1	CMCC peripheral is write protected

Value	Description
0	The ERROR interrupt is disabled.
1	The ERROR interrupt is enabled.

Bit 2 – TXC Transmission Complete Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' will set the corresponding interrupt request.

Value	Description
0	The TXC interrupt is disabled.
1	The TXC interrupt is enabled.

Bit 1 – DRE Transmit Data Register Empty Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' will set the corresponding interrupt request.

Value	Description
0	The DRE interrupt is disabled.
1	The DRE interrupt is enabled.

Bit 0 – RXC Receive Data Register Full Interrupt Enable
Writing a '0' to this bit has no effect.

Writing a '1' will set the corresponding interrupt request.

Value	Description
0	The RXC interrupt is disabled.
1	The RXC interrupt is enabled.

A Dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (refer to table below). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) • Element Size to the Tx Buffer Start Address TXBC.TBSA.

Table 39-7. Tx Buffer / FIFO / Queue Element Size

TXESC.TBDS[2:0]	Data Field [bytes]	Element Size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

39.6.6.3 Tx FIFO

Tx FIFO operation is configured by programming TXBC.TFQM to '0'. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the Get Index TXFQS.TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The CAN calculates the Tx FIFO Free Level TXFQS.TFFL as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index TXFQS.TFQPI. An Add Request increments the Put Index to the next free Tx FIFO element. When the Put Index reaches the Get Index, Tx FIFO Full (TXFQS.TFQF = '1') is signaled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a '1' to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO's Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via TXBAR. The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is canceled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (refer to Table 39-7). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/ Queue Put Index TXFQS.TFQPI (0…31) • Element Size to the Tx Buffer Start Address TXBC.TBSA.

39.6.6.4 Tx Queue

Tx Queue operation is configured by programming TXBC.TFQM to '1'. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that

register (CCCR.CSR = 1). Once all pending transactions are completed and the idle bus state is detected, the CAN will automatically set the Clock Stop Acknowledge bit (CCCR.CSA = 1). The CAN then reverts back to its initial state (CCCR.INIT = 1), blocking further transfers, and it is now safe for CLK_CANx_APB and GCLK_CANx to be switched off and the system may go to standby.

To leave low power mode, CLK_CANx_APB and GCLK_CANx must be active before writing CCCR.CSR to '0'. The CAN will acknowledge this by resetting CCCR.CSA = 0. Afterwards, the application can restart CAN communication by resetting bit CCCR.INIT.

39.6.10 Synchronization

Due to the asynchronicity between the main clock domain (CLK_CAN_APB) and the peripheral clock domain (GCLK_CAN) some registers are synchronized when written. When a write-synchronized register is written, the read back value will not be updated until the register has completed synchronization.

The following bits and registers are write-synchronized:

l Initialization bit in CC Control register (CCCR.INIT)

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 25 – BOE Bus_Off Status Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 24 – EWE Error Warning Status Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 23 – EPE Error Passive Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 22 – ELOE Error Logging Overflow Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 21 – BEUE Bit Error Uncorrected Interrupt Enable.

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 20 – BECE Bit Error Corrected Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 19 – DRXE Message stored to Dedicated Rx Buffer Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

Bit 18 – TOOE Timeout Occurred Interrupt Enable

Value	Description
0	Interrupt disabled.
1	Interrupt enabled.

```
            }
else // Manage the error
```

43.3.4.2.6 Status Returned Values

Table 43-8. ClearFlags Service Return Codes

Returned Status	Importance	Meaning
PUKCL_OK	–	Service functioned correctly.

43.3.4.3 Swap

43.3.4.3.1 Purpose

This service performs swapping of two buffers.

43.3.4.3.2 How to Use the Service

43.3.4.3.3 Description

This service swaps two buffers, X and Y, of the same size in memory.

The service name for this operation is `Swap`.

43.3.4.3.4 Parameters Definition

This service can easily be accessed through the use of the `PUKCL_Swap()` macro.

Table 43-9. Swap Service Parameters

Parameter	Type	Direction	Location	Data Length	Before Executing the Service	After Executing the Service
nu1XBase	nu1	I	Crypto RAM	u2Length	Base of the number X	Base of X filled with Y
nu1YBase	nu1	I	Crypto RAM	u2Length	Base of the number Y	Base of Y filled with X
u2XLength	u2	I	–	–	Length of X and Y	Length of X and Y

43.3.4.3.5 Code Example

```
_PARAM PUKCLParam;
PPUKCL_PARAM pvPUKCLParam = &PUKCLParam;

// Initialize parameters
PUKCL_Swap(nu1XBase) = <Base of the X number>;
PUKCL_Swap(nu1YBase) = <Base of the Y number>;
PUKCL_Swap(u2XLength) = <Length of the numbers>;

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

43.3.4.3.6 Constraints

The following conditions must be avoided to ensure that the service works correctly:

- nu1XBase or nu1YBase are not aligned on 32-bit boundaries
- u2XLength is either <4, > 0xffc, or not a 32-bit length
- {nu1XBase, u2XLength} or {nu1YBase, u2XLength} do not entirely lie in PUKCCRAM

- Both PUKCL_EXPMOD_REGULARRSA and PUKCL_EXPMOD_FASTRSA are specified.
- {nu1PrecompBase,<PrecompLength>} overlaps with either: {nu1NBase, u2NLength + 4}, {nu1CnsBase, u2NLength + 12} {nu1RndBase, u2NLength + 12} or {nu1ExpBase, u2ExpLength + 4}
- {nu1RndBase,3*u2NLength + 24} overlaps with either: {nu1NBase, u2NLength + 4},{nu1CnsBase, u2NLength + 12} {nu1XBase, u2NLength + 12} or {nu1ExpBase, u2ExpLength + 4}
- {nu1NBase, u2NLength + 4} overlaps {nu1CnsBase, u2NLength +12}

43.3.5.3.9 Status Returned Values

Table 43-61. PrimeGen Service Return Codes

Returned Status	Importance	Meaning
PUKCL_NUMBER_IS_PRIME	Information	The generated or tested number has been detected as probably prime.
PUKCL_NUMBER_IS_NOT_PRIME	Information	The generated or tested number has been detected as composite.

43.3.5.4 Modular Exponentiation (With CRT)

43.3.5.4.1 Purpose

The purpose of this service is to perform the Modular Exponentiation with the Chinese Remainders Theorem (CRT). This service processes integers in GF(p) only.

The options available for this service are:

- Fast implementation
- Regular implementation
- Exponent is located in Crypto RAM or not
- Exponent window size

43.3.5.4.2 How to Use the Service

43.3.5.4.3 Description

This service processes a Modular Exponentiation with the Chinese Remainder Theorem:

$R = X^D mod(N)$ *with* $N = P * Q$

Important: For this service, be sure to follow the directives given for the RSA implementation on the chip you use.

This service requires that the modulus N is the product of two co-primes P and Q and that the decryption exponents D is co-prime with the product ((P-1)*(Q-1)).

The Input data are P, Q, EP, EQ, Rvalue, and X. P and Q are the co-primes so that N = P*Q.

X is the number to exponentiate.

EP, EQ and Rval are calculated as follows:

$EP = Dmod(P - 1)$ $EQ = Dmod(Q - 1)$ $Rval = P^{-1}mod(Q)$

In some cases, the decryption exponent D may not be available and the encryption exponent E may be available instead. The possibilities to calculate the parameters are:

43.3.6.2.4 Parameters Definition

Table 43-68. ZpEccAddFast 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
nu1Workspace	nu1	I	Crypto RAM	5*u2ModLength + 32	–	Corrupted workspace

43.3.6.2.5 Code Example

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

PUKCL (u2Option) = 0;

PUKCL _ZpEccAdd(nu1ModBase) = <Base of the ram location of P>;
PUKCL _ZpEccAdd(nu1CnsBase) = <Base of the ram location of Cns>;
PUKCL _ZpEccAdd(u2ModLength) = <Byte length of P>;
PUKCL _ZpEccAdd(nu1PointABase) = <Base of the ram location of the A point>;
PUKCL _ZpEccAdd(nu1PointBBase) = <Base of the ram location of the B point>;
PUKCL _ZpEccAdd(nu1Workspace) = <Base of the ram location of the workspace>;
...

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

43.3.6.2.6 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
- {nu1ModBase, u2ModLength + 4}, {nu1CnsBase, u2ModLength + 8}, {nu1PointABase, 3*u2ModLength + 12}, {nu1PointBBase, 3*u2ModLength + 12}, {nu1Workspace, <WorkspaceLength>} are not in Crypto RAM
- u2ModLength is either: < 12, > 0xffc or not a 32-bit length

43.3.6.4.4 Parameters Definition

Table 43-72. ZpEccDblFastService

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 modulus P	Length of modulus P
nu1ABase	u2	I	Crypto RAM	u2ModLength + 4	Parameter a of the elliptic curve	Parameter a of the elliptic curve
nu1PointABase	nu1	I/O	Crypto RAM	3*u2ModLength + 12	Input point A (projective coordinates)	Resulting point C (projective coordinates)
nu1Workspace	nu1	I	Crypto RAM	4*u2ModLength + 28	–	Corrupted workspace

43.3.6.4.5 Code Example

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


PUKCL (u2Option) = 0;

PUKCL _ZpEccDbl(nu1ModBase) = <Base of the ram location of P>;
PUKCL _ZpEccDbl(u2ModLength) = <Byte length of P>;
PUKCL _ZpEccDbl(nu1CnsBase) = <Base of the ram location of Cns>;
PUKCL _ZpEccDbl(nu1PointABase) = <Base of the ram location of the A point>;
PUKCL _ZpEccDbl(nu1ABase) = <Base of the a parameter of the elliptic curve>;
PUKCL _ZpEccDbl(nu1Workspace) = <Base of the ram location of the workspace>;
...

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

43.3.6.4.6 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, nu1ABase, nu1Workspace are not aligned on 32-bit boundaries
- {nu1ModBase, u2ModLength + 4}, {nu1CnsBase, u2ModLength + 8}, {nu1PointABase, 3*u2ModLength+ 12}, {nu1ABase, u2ModLength + 4}, {nu1Workspace, <WorkspaceLength>} are not in Crypto RAM
- u2ModLength is either: < 12, > 0xffc or not a 32-bit length

47.6.9.5 Dithering Mode

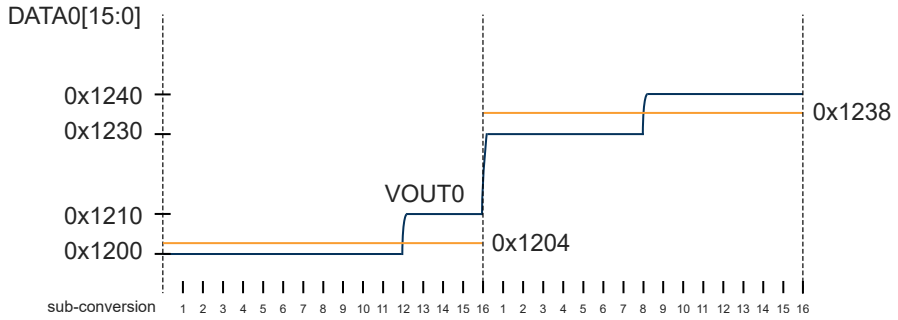
Dithering is enabled by setting DACCTRLx.DITHER to 1. In dithering mode, DATAx is a 16-bit unsigned value where DATAx[15:4] is the 12-bit data converted by DAC and DATAx[3:0] represent the dither bits, used to minimize the quantization error.

The principle is to make 16 sub-conversions of the DATAx[15:4] value or the (DATAx[15:4] + 1) value, so that by averaging those two values, the conversion result of the 16-bit value (DATAx[15:0]) s accurate.

To operate, the STARTx event must be configured to generate 16 events for each DATAx[15:0] conversion, and DATABUFx must be loaded every 16 DAC conversions. EMPTYx event and DMA request are therefore generated every 16 DATABUFx to DATAx transfer. STATUS.EOCx still reports end of each sub-conversions.

Following timing diagram shows examples with DATA0[15:0] = 0x1204 followed by DATA0[15:0] = 0x1238.

Figure 47-5. DAC Conversions in Dithering Mode



47.6.9.6 Interpolation Mode

The DAC provides interpolation that allows for oversampling ratios (OSR) of 2x, 4x, 8x, 16x or 32x. Interpolation mode is selected by writing a non-zero value to the Oversampling Ratio bits in the DACx Control register (DACCTRLx.OSR).

The data is sampled once over OSR trigger events and then recomputed at the trigger sample rate using a third-order SINC filter.

The figures below show the spectral mask of the SINC filter depending on the selected OSR. f_S is the sampling frequency of the input signal which corresponds to the trigger frequency divided by OSR.

The Filter usage bit DACCTRLx.FEXT determines whether the filter is integrated to the corresponding DAC or used as a standalone filter driven by DMA. If DACCTRLx.FEXT=0, the DAC takes the filter output while the value of RESULTx is reading zero. Conversely, If DACCTRLx.FEXT=1, the DAC value remains zero, and the value of RESULTx register reflects the filter output.

Figure 47-6. Interpolator Spectral Mask for 2x OSR

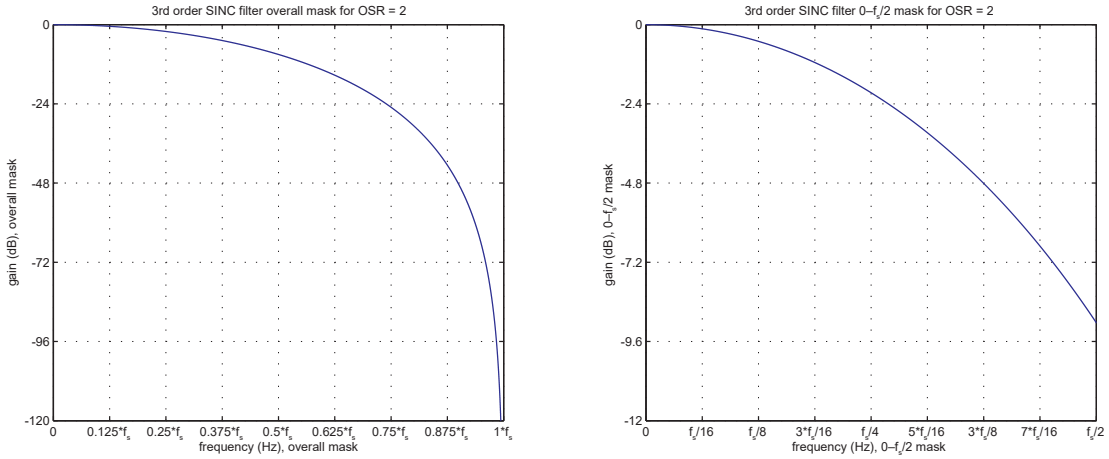


Figure 47-7. Interpolator Spectral Mask for 4x OSR

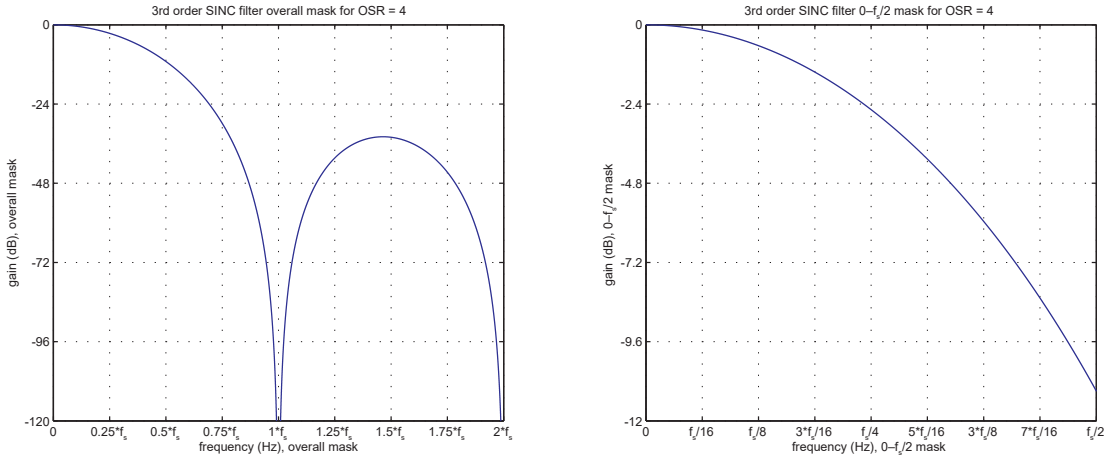
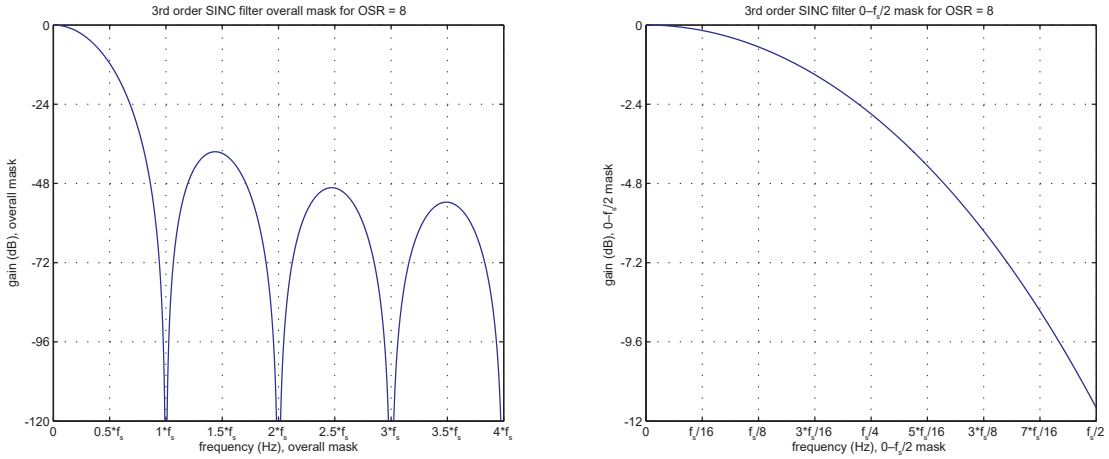


Figure 47-8. Interpolator Spectral Mask for 8x OSR



Bit 0 – DIR Counter Direction
This bit is used to change the direction of the counter.

Writing a '0' to this bit has no effect.

Writing a '1' to this bit will clear the bit and make the counter count up.

Value	Description
0	The timer/counter is counting up (incrementing).
1	The timer/counter is counting down (decrementing).

59. Revision History

Table 59-1. Rev. B - 4/2018

Section Name or Type	Change Description
Features	Updated CAN FD reference. Added 120-ball TFBGA package.
Configuration Summary	Added 120-ball TFBGA to the family feature tables.
Ordering Information	Updated the notes for devices in WLCSP packages. Updated Package Type, adding CT = TFBGA.
Pinout	Added the 120-ball TFBGA package pinout diagram.
Multiplexed Signals	Added 120-ball TFBGA and updated Note 3 (see Table 6-1.
OSC32KCTRL - 32 kHz Oscillators Controller	Added the EN1K and EN32K bits to the OSCULP32K register (see 29.8.9 OSCULP32K).
SERCOM - Serial Communication Interface	Added Fractional Baud information to the Baud Rate Equations (see Table 33-2).
QSPI - Quad Serial Peripheral Interface	Added equations to the BAUD register (see 37.8.3 BAUD).
CAN - Control Area Network	Updated the Overview. Updated ISO 11898 references throughout the chapter.
Public Key Cryptography Controller (PUKCC)	Added the Public Key Cryptography Library (PUKCL) Application Programmer Interface (API) section.
TCC - Timer/Counter for Control Applications	Updated the number of TCC instances to 5 (4:0).
54. Electrical Characteristics at 85°C	(1) Improved SPI maximum speed information in Table 54-52. (2). Added example for QSPI maximum frequency examples Table 54-54.
Packaging Information	Added the 120-ball TFBGA package (see 55.3.6 120-ball TFBGA).

Table 59-2. Rev. A - 07/2017

This is the initial release of the document.