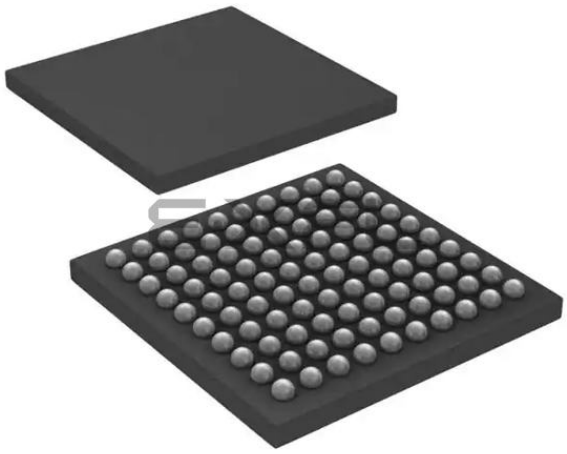


Welcome to E-XFL.COM

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®-M7
Core Size	32-Bit Single-Core
Speed	300MHz
Connectivity	I²C, IrDA, LINbus, MMC/SD/SDIO, QSPI, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	75
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256K x 8
Voltage - Supply (Vcc/Vdd)	1.08V ~ 3.6V
Data Converters	A/D 10x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	100-VFBGA
Supplier Device Package	100-VFBGA (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsams70n19a-cfnt

Bit 1 – WKUPS WKUP Wakeup Status (cleared on read)

Value	Description
0	(NO): No wakeup due to the assertion of the WKUP pins has occurred since the last read of SUPC_SR.
1	(PRESENT): At least one wakeup due to the assertion of the WKUP pins has occurred since the last read of SUPC_SR.

31.10.3 Asynchronous Partial Wakeup in Active Mode

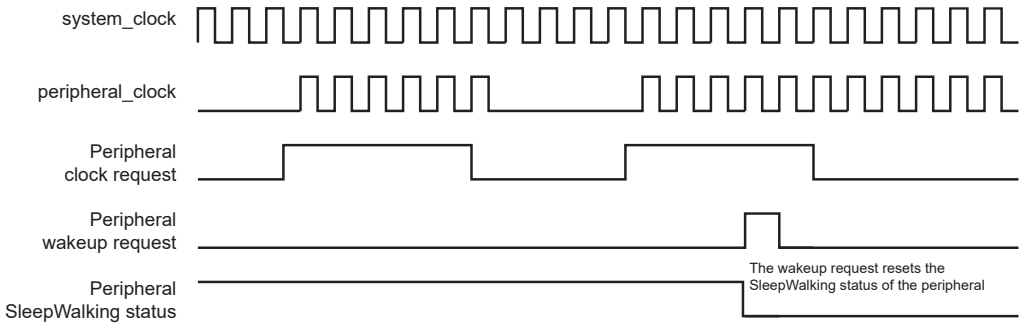
When the system is in Active mode, peripherals enabled for asynchronous partial wakeup have their respective clocks stopped until the peripherals request a clock. When a peripheral requests the clock, the PMC provides the clock without processor intervention.

The triggering of the peripheral clock request depends on conditions which can be configured for each peripheral. If these conditions are met, the peripheral asserts a request to the PMC. The PMC disables the Asynchronous Partial Wakeup mode of the peripheral and provides the clock to the peripheral until the user instructs the PMC to re-enable partial wakeup on the peripheral. This is done by setting PMC_SLPWK_ER.PIDx.

If the conditions are not met, the peripheral clears the clock request and the PMC stops the peripheral clock until the clock request is reasserted by the peripheral.

Note: Configuring Asynchronous Partial Wakeup mode requires the same registers as SleepWalking mode.

Figure 31-3. Asynchronous Partial Wakeup in Active Mode



31.10.3.1 Configuration Procedure

Before configuring the asynchronous partial wakeup function of a peripheral, check that the PIDx bit in PMC_PCSR is set. This ensures that the peripheral clock is enabled.

The steps to enable the asynchronous partial wakeup function of a peripheral are the following:

1. Check that the corresponding PIDx bit in the PMC SleepWalking Activity Status register (PMC_SLPWK_ASR) is set to ‘0’. This ensures that the peripheral has no activity in progress.
2. Enable the asynchronous partial wakeup function of the peripheral by writing a ‘1’ to the corresponding PIDx bit in the PMC_SLPWK_ER.
3. Check that the corresponding PIDx bit in PMC_SLPWK_ASR is set to ‘0’. This ensures that no activity has started during the enable phase.

If an activity has started during the enable phase, the asynchronous partial wakeup function must be immediately disabled by writing a ‘1’ to the PIDx bit in the PMC SleepWalking Disable register (PMC_SLPWK_DR). Wait for the end of peripheral activity before reinitializing the procedure.

31.11 Free-running Processor Clock

The free-running Processor clock (FCLK) used for sampling interrupts and clocking debug blocks ensures that interrupts can be sampled, and sleep events can be traced, while the processor is sleeping.

Offset	Register	Name	Access	Reset
0x0028	PLLA Register	CKGR_PLLAR	Read/Write	0x0000_3F00
0x002C	Reserved	–	–	–
0x0030	Master Clock Register	PMC_MCKR	Read/Write	0x0000_0001
0x0034	Reserved	–	–	–
0x0038	USB Clock Register	PMC_USB	Read/Write	0x0000_0000
0x003C	Reserved	–	–	–
0x0040+chid*0x04	Programmable Clock Register	PMC_PCK	Read/Write	0x0000_0000
0x005C	Reserved	–	–	–
0x0060	Interrupt Enable Register	PMC_IER	Write-only	–
0x0064	Interrupt Disable Register	PMC_IDR	Write-only	–
0x0068	Status Register	PMC_SR	Read-only	0x0003_0008
0x006C	Interrupt Mask Register	PMC_IMR	Read-only	0x0000_0000
0x0070	Fast Startup Mode Register	PMC_FSMR	Read/Write	0x0000_0000
0x0074	Fast Startup Polarity Register	PMC_FSPR	Read/Write	0x0000_0000
0x0078	Fault Output Clear Register	PMC_FOCR	Write-only	–
0x007C–0x00E0	Reserved	–	–	–
0x00E4	Write Protection Mode Register	PMC_WPMR	Read/Write	0x0
0x00E8	Write Protection Status Register	PMC_WPSR	Read-only	0x0
0x00EC–0x00FC	Reserved	–	–	–
0x0100	Peripheral Clock Enable Register 1	PMC_PCER1	Write-only	–
0x0104	Peripheral Clock Disable Register 1	PMC_PCDR1	Write-only	–
0x0108	Peripheral Clock Status Register 1	PMC_PCSR1	Read-only	0x0000_0000
0x010C	Peripheral Control Register	PMC_PCR	Read/Write	0x0000_0000
0x0110	Oscillator Calibration Register	PMC_OCR	Read/Write	(See **Note 2**)
0x0114	SleepWalking Enable Register 0	PMC_SLPWK_ER0	Write-only	–
0x0118	SleepWalking Disable Register 0	PMC_SLPWK_DR0	Write-only	–
0x011C	SleepWalking Status Register 0	PMC_SLPWK_SR0	Read-only	0x00000000
0x0120	SleepWalking Activity Status Register 0	PMC_SLPWK_ASR0	Read-only	0x00000000

32.6.1.21 PIO Pull-Up Disable Register

Name: PIO_PUDR
Offset: 0x0060
Property: Write-only

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

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access								
Reset								

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P PIO Pull-Up Disable

Value	Description
0	No effect.
1	Disables the pullup resistor on the I/O line.

Name	Function	Type	Active Level
NWAIT	External Wait Signal	Input	Low
SMC			
NCS0–EBI_NCS3	Chip Select Lines	Output	Low
NWR0–NWR1	Write Signals	Output	Low
NRD	Read Signal	Output	Low
NWE	Write Enable	Output	Low
NBS0–NBS1	Byte Mask Signals	Output	Low
EBI for NAND Flash Support			
NANDCS	NAND Flash Chip Select Line	Output	Low
NANDOE	NAND Flash Output Enable	Output	Low
NANDWE	NAND Flash Write Enable	Output	Low
SDRAM Controller			
SDCK (see **Note**)	SDR-SDRAM Clock	Output	
SDCKE	SDR-SDRAM Clock Enable	Output	High
SDCS	SDR-SDRAM Controller Chip Select Line	Output	Low
BA0–1	Bank Select	Output	
SDWE	SDR-SDRAM Write Enable	Output	Low
RAS - CAS	Row and Column Signal	Output	Low
SDA10	SDRAM Address 10 Line	Output	

Note: SDCK is the MCK clock for EBI, SDRAM Controller and SMC interfaces.

The connection of some signals through the MUX logic is not direct and depends on the Memory Controller in use at the moment.

The following table details the connections between the two Memory Controllers and the EBI pins.

Table 33-2. EBI Pins and Memory Controllers I/O Lines Connections

EBIx Pins	SDRAM I/O Lines	SMC I/O Lines
NWR1/NBS1	NBS1	NWR1
A0/NBS0	NBS0	SMC_A0
A1	Not Supported	SMC_A1
A[11:2]	SDRAMC_A[9:0]	SMC_A[11:2]
SDA10	SDRAMC_A10	Not Supported
A12	Not Supported	SMC_A12
A[15:13]	SDRAMC_A[13:11]	SMC_A[15:13]

34.7.7 SDRAMC Interrupt Mask Register

Name: SDRAMC_IMR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

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
								RES
Access								R
Reset								0

Bit 0 – RES Refresh Error Interrupt Mask

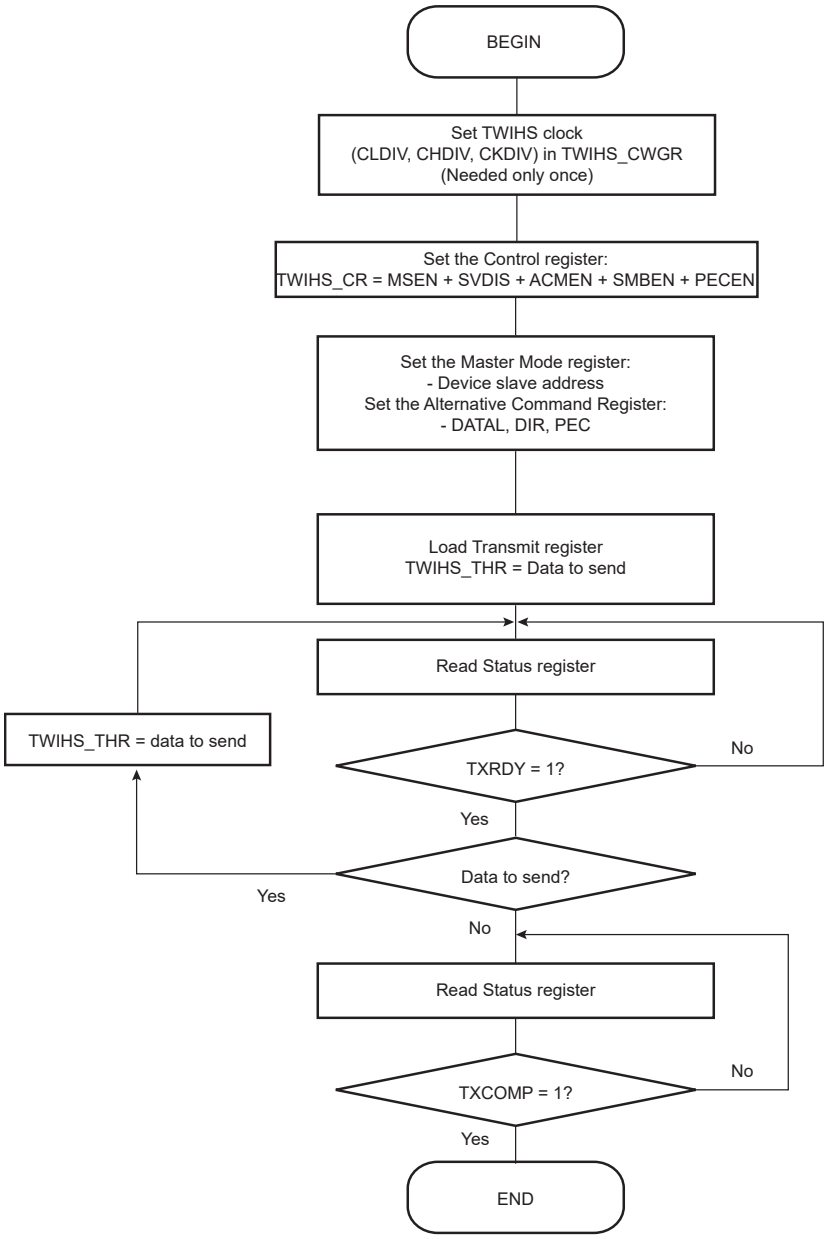
Value	Description
0	The refresh error interrupt is disabled.
1	The refresh error interrupt is enabled.

Offset	Name	Bit Pos.								
0x037C	XDMAC_CDS_MSP12	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0380	XDMAC_CSUS12	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0384	XDMAC_CDUS12	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0388 ... 0x038F	Reserved									
0x0390	XDMAC_CIE13	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0394	XDMAC_CID13	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0398	XDMAC_CIM13	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x039C	XDMAC_CIS13	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x03A0	XDMAC_CSA13	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x03A4	XDMAC_CDA13	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x03A8	XDMAC_CNDA13	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x03AC	XDMAC_CNDC13	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								

Offset	Name	Bit Pos.								
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x04AC	XDMAC_CNDC17	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x04B0	XDMAC_CUBC17	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x04B4	XDMAC_CBC17	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x04B8	XDMAC_CC17	7:0	MEMSET	SWREQ		DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						
0x04BC	XDMAC_CDS_MSP17	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x04C0	XDMAC_CSUS17	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x04C4	XDMAC_CDUS17	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x04C8 ... 0x04CF	Reserved									
0x04D0	XDMAC_CIE18	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x04D4	XDMAC_CID18	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x04D8	XDMAC_CIM18	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								

Offset	Name	Bit Pos.								
0x0754	USBHS_HSTDMAADDRESSx	7:0	BUFF_ADD[7:0]							
		15:8	BUFF_ADD[15:8]							
		23:16	BUFF_ADD[23:16]							
		31:24	BUFF_ADD[31:24]							
0x0758	USBHS_HSTDMACONTROLx	7:0	BURST_LCK	DESC_LD_IT	END_BUFFIT	END_TR_IT	END_B_EN	END_TR_EN	LDNXT_DSC	CHANN_ENB
		15:8								
		23:16	BUFF_LENGTH[7:0]							
		31:24	BUFF_LENGTH[15:8]							
0x075C	USBHS_HSTDMASTATUSx	7:0		DESC_LDST	END_BF_ST	END_TR_ST			CHANN_ACT	CHANN_ENB
		15:8								
		23:16	BUFF_COUNT[7:0]							
		31:24	BUFF_COUNT[15:8]							
0x0760	USBHS_HSTDMANXTDSC7	7:0	NXT_DSC_ADD[7:0]							
		15:8	NXT_DSC_ADD[15:8]							
		23:16	NXT_DSC_ADD[23:16]							
		31:24	NXT_DSC_ADD[31:24]							
0x0764	USBHS_HSTDMAADDRESSx	7:0	BUFF_ADD[7:0]							
		15:8	BUFF_ADD[15:8]							
		23:16	BUFF_ADD[23:16]							
		31:24	BUFF_ADD[31:24]							
0x0768	USBHS_HSTDMACONTROLx	7:0	BURST_LCK	DESC_LD_IT	END_BUFFIT	END_TR_IT	END_B_EN	END_TR_EN	LDNXT_DSC	CHANN_ENB
		15:8								
		23:16	BUFF_LENGTH[7:0]							
		31:24	BUFF_LENGTH[15:8]							
0x076C	USBHS_HSTDMASTATUSx	7:0		DESC_LDST	END_BF_ST	END_TR_ST			CHANN_ACT	CHANN_ENB
		15:8								
		23:16	BUFF_COUNT[7:0]							
		31:24	BUFF_COUNT[15:8]							
0x0770 ... 0x07FF	Reserved									
0x0800	USBHS_CTRL	7:0				RDERRE				
		15:8	USBE	FRZCLK						VBUSHWC
		23:16								
		31:24							UIMOD	UID
0x0804	USBHS_SR	7:0				RDERRI				
		15:8		CLKUSABLE	SPEED[1:0]					
		23:16								
		31:24								
0x0808	USBHS_SCR	7:0				RDERRIC				
		15:8								
		23:16								
		31:24								
0x080C	USBHS_SFR	7:0				RDERRIS				
		15:8							VBUSRQS	

Figure 43-20. TWIHS Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC



46.7.4 USART Mode Register (SPI_MODE)

Name: US_MR (SPI_MODE)
Offset: 0x0004
Reset: 0x0
Property: Read/Write

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

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

– Applicable if USART operates in SPI mode (USART_MODE = 0xE or 0xF):

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
				WRDBT		CLKO		CPOL
Access								
Reset				0		0		0

Bit	15	14	13	12	11	10	9	8
								CPHA
Access								
Reset								0

Bit	7	6	5	4	3	2	1	0
	CHRL[1:0]		USCLKS[1:0]		USART_MODE[3:0]			
Access								
Reset	0	0	0	0	0	0	0	0

Bit 20 – WRDBT Wait Read Data Before Transfer

Value	Description
0	The character transmission starts as soon as a character is written into US_THR (assuming TXRDY was set).
1	The character transmission starts when a character is written and only if RXRDY flag is cleared (Receive Holding Register has been read).

Bit 18 – CLKO Clock Output Select

Value	Description
0	The USART does not drive the SCK pin.
1	The USART drives the SCK pin if USCLKS does not select the external clock SCK.

Bit 16 – CPOL SPI Clock Polarity
Applicable if USART operates in SPI mode (Slave or Master, USART_MODE = 0xE or 0xF):

CPOL is used to determine the inactive state value of the serial clock (SCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

Table 48-8. MediaLB Channel Address to Logical Channel Mapping

Channel Address	Logical Channel
0x0002	1
0x0004	2
0x0006	3
....	....
0x007C	62
0x007E	63
0x01FE	0(1)

Note: 1. Logical Channel 0 is the System Channel and is reserved.

48.6.3.2 Host Bus Interface Block

The Host Bus Interface (HBI) block provides a 16-bit parallel slave port that provides an external Host Controller (HC) with access to all MOST channels and data types.

Up to 64 independent HBI channels are available to the HC, each configurable for either transmitting or receiving a particular application data type (synchronous, isochronous, asynchronous, or control). The HBI block provides source and sink access to the full network data bandwidth.

HBI Physical Addresses

To access a particular HBI DMA channel, hardware must first translate the HBI channel address to a channel allocation table (CAT) physical address. This physical address is then used to retrieve the channel label (CL), which in turn retrieves the channel descriptor.

See the following table for more information on the mapping between the HBI channel address and physical address.

Table 48-9. HBI Channel Address to Physical Address Mapping

HBI Channel	CAT Address	CAT Offset
0x0	0x88	000
0x1	0x88	001
0x2	0x88	010
0x3	0x88	011
0x4	0x88	100
0x5	0x88	101
0x6	0x88	110
0x7	0x88	111
0x8	0x89	000
...	...	...

4. Program the CAT for the inbound DMA
    4.1. For Tx channels (to MediaLB) HBI is the inbound DMA
    4.2. For Rx channels (from MediaLB) MediaLB is the inbound DMA
    4.3. Set the channel direction: RNW = 0
    4.4. Set the channel type:
    CT[2:0] = 010 (asynchronous), 001 (control), 011 (isochronous), or 000 (synchronous)
    4.5. Set the connection label: CL[5:0] = N
    4.6. If CT[2:0] = 000 (synchronous), set the mute bit (MT = 1)
    4.7. Set the channel enable: CE = 1
    4.8. Set all other bits of the CAT to ‘0’
5. Program the CAT for the outbound DMA
    5.1. For Tx channels (to MediaLB) MediaLB is the outbound DMA
    5.2. For Rx channels (from MediaLB) HBI is the outbound DMA
    5.3. Set the channel direction: RNW = 1
    5.4. Set the channel type: CT[2:0] = 010 (asynchronous), 001 (control), 011 (isochronous), or 000 (synchronous)
    5.5. Set the channel label: CL[5:0] = N
    5.6. If CT[2:0] = 000 (synchronous), set the mute bit (MT = 1)
    5.7. Set the channel enable: CE = 1
    5.8. Set all other bits of the CAT to ‘0’
6. Repeat steps 2–5 to initialize all logical channels

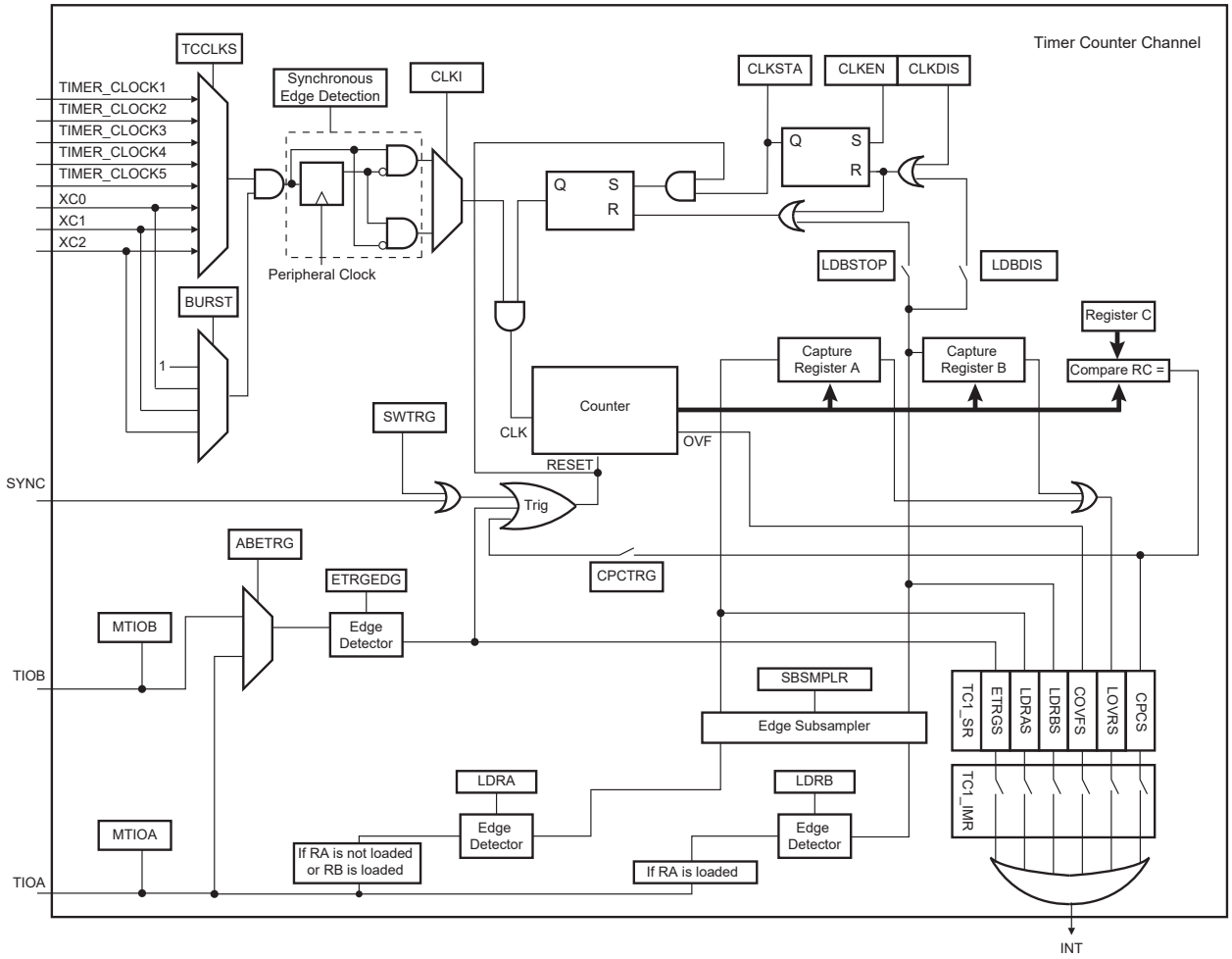
Program the AHB Block DMAs

The ADT resides in the external CTR and is programmed indirectly via APB reads and writes to the MIF.

1. Initialize all bits of the ADT to ‘0’
2. Select a logical channel: N = 0–63
3. Program the AHB block ping page for channel N
    3.1. Set the 32-bit base address (BA1)
    3.2. Set the 11-bit buffer depth (BD1): BD1 = buffer depth in bytes - 1
        3.2.1. For synchronous channels: (BD1 + 1) = n x frames per sub-buffer (m) x bytes-per-frame (bpf)
        3.2.2. For isochronous channels: (BD1 + 1) mod (BS + 1) = 0
        3.2.3. For asynchronous channels: 5 ≤ (BD1 + 1) ≤ 4096 (max packet length)
        3.2.4. For control channels: 5 ≤ (BD1 + 1) ≤ 4096 (max packet length)
    3.3. For asynchronous and control Tx channels set the packet start bit (PS1) iff the page contains the start of the packet
    3.4. Clear the page done bit (DNE1)
    3.5. Clear the error bit (ERR1)
    3.6. Set the page ready bit (RDY1)
4. Program the AHB block pong page for channel N
    4.1. Set the 32-bit base address (BA2)
    4.2. Set the 11-bit buffer depth (BD2): BD2 = buffer depth in bytes - 1
        4.2.1. For synchronous channels: (BD2 +1) = n x frames per sub-buffer (m) x bytes-per-frame (bpf)

Offset	Name	Bit Pos.								
... 0x7F										
0x80	MLB_HCTL	7:0							RST1	RST0
		15:8	EN							
		23:16								
		31:24								
0x84 ... 0x87	Reserved									
0x88	MLB_HCMR0	7:0	CHM: Bitwise Channel Mask Bit [31[7:0]							
		15:8	CHM: Bitwise Channel Mask Bit [31[15:8]							
		23:16	CHM: Bitwise Channel Mask Bit [31[23:16]							
		31:24	CHM: Bitwise Channel Mask Bit [31[31:24]							
0x8C	MLB_HCMR1	7:0	CHM: Bitwise Channel Mask Bit [63[7:0]							
		15:8	CHM: Bitwise Channel Mask Bit [63[15:8]							
		23:16	CHM: Bitwise Channel Mask Bit [63[23:16]							
		31:24	CHM: Bitwise Channel Mask Bit [63[31:24]							
0x90	MLB_HCER0	7:0	CERR: Bitwise Channel Error Bit [31[7:0]							
		15:8	CERR: Bitwise Channel Error Bit [31[15:8]							
		23:16	CERR: Bitwise Channel Error Bit [31[23:16]							
		31:24	CERR: Bitwise Channel Error Bit [31[31:24]							
0x94	MLB_HCER1	7:0	CERR: Bitwise Channel Error Bit [63[7:0]							
		15:8	CERR: Bitwise Channel Error Bit [63[15:8]							
		23:16	CERR: Bitwise Channel Error Bit [63[23:16]							
		31:24	CERR: Bitwise Channel Error Bit [63[31:24]							
0x98	MLB_HCBR0	7:0	CHB: Bitwise Channel Busy Bit [31[7:0]							
		15:8	CHB: Bitwise Channel Busy Bit [31[15:8]							
		23:16	CHB: Bitwise Channel Busy Bit [31[23:16]							
		31:24	CHB: Bitwise Channel Busy Bit [31[31:24]							
0x9C	MLB_HCBR1	7:0	CHB: Bitwise Channel Busy Bit [63[7:0]							
		15:8	CHB: Bitwise Channel Busy Bit [63[15:8]							
		23:16	CHB: Bitwise Channel Busy Bit [63[23:16]							
		31:24	CHB: Bitwise Channel Busy Bit [63[31:24]							
0xA0 ... 0xBF	Reserved									
0xC0	MLB_MDAT0	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0xC4	MLB_MDAT1	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	DATA[23:16]							
		31:24	DATA[31:24]							
0xC8	MLB_MDAT2	7:0	DATA[7:0]							
		15:8	DATA[15:8]							

Figure 50-6. Capture Mode



50.6.11 Waveform Mode

Waveform mode is entered by setting the TC_CMRx.WAVE bit.

In Waveform mode, the TC channel generates one or two PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOAx is configured as an output and TIOBx is defined as an output if it is not used as an external event (EEVT parameter in TC_CMR).

Waveform Mode shows the configuration of the TC channel when programmed in Waveform operating mode.

50.6.12 Waveform Selection

Depending on the WAVSEL parameter in TC_CMR, the behavior of TC_CV varies.

With any selection, TC_RA, TC_RB and TC_RC can all be used as compare registers.

RA Compare is used to control the TIOAx output, RB Compare is used to control the TIOBx output (if correctly configured) and RC Compare is used to control TIOAx and/or TIOBx outputs.

Offset	Name	Bit Pos.								
		23:16								
		31:24								
0xC4	TC_BMR	7:0			TC2XC2S[1:0]		TC1XC1S[1:0]		TC0XC0S[1:0]	
		15:8	INVIDX	INVB	INVA	EDGPHA	QDTRANS	SPEEDEN	POSEN	QDEN
		23:16	MAXFILT[3:0]					AUTOC	IDXPHB	SWAP
		31:24			MAXCMP[3:0]				MAXFILT[5:4]	
0xC8	TC_QIER	7:0					MPE	QERR	DIRCHG	IDX
		15:8								
		23:16								
		31:24								
0xCC ... 0xCF	Reserved									
0xD0	TC_QIMR	7:0					MPE	QERR	DIRCHG	IDX
		15:8								
		23:16								
		31:24								
0xD4	TC_QISR	7:0					MPE	QERR	DIRCHG	IDX
		15:8								DIR
		23:16								
		31:24								
0xD8	TC_FMR	7:0							ENCF1	ENCF0
		15:8								
		23:16								
		31:24								
0xDC ... 0xE3	Reserved									
0xE4	TC_WPMR	7:0								WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

51.7.11 PWM Sync Channels Update Control Register

Name: PWM_SCUC
Offset: 0x28
Reset: 0x00000000
Property: Read/Write

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
								UPDULOCK
Access								R/W
Reset								0

Bit 0 – UPDULOCK Synchronous Channels Update Unlock
This bit is automatically reset when the update is done.

Value	Description
0	No effect
1	If the UPDM field is set to ‘0’ in PWM Sync Channels Mode Register, writing the UPDULOCK bit to ‘1’ triggers the update of the period value, the duty-cycle and the dead-time values of synchronous channels at the beginning of the next PWM period. If the field UPDM is set to ‘1’ or ‘2’, writing the UPDULOCK bit to ‘1’ triggers only the update of the period value and of the dead-time values of synchronous channels.

Moreover, when a DMA channel is connected to the AFEC, a resolution lower than 16 bits sets the transfer request size to 16 bits.

Note: If ADTRG is asynchronous to the AFEC peripheral clock, the internal resynchronization introduces a jitter of 1 peripheral clock. This jitter may reduce the resolution of the converted signal. Refer to the formula below, where f_{IN} is the frequency of the analog signal to convert and t_J is the half-period of 1 peripheral clock.

$$\mathrm{SNR} = 20 \times \log 10\left(\frac{1}{2\pi f_{\mathrm{IN}} t_J}\right)$$

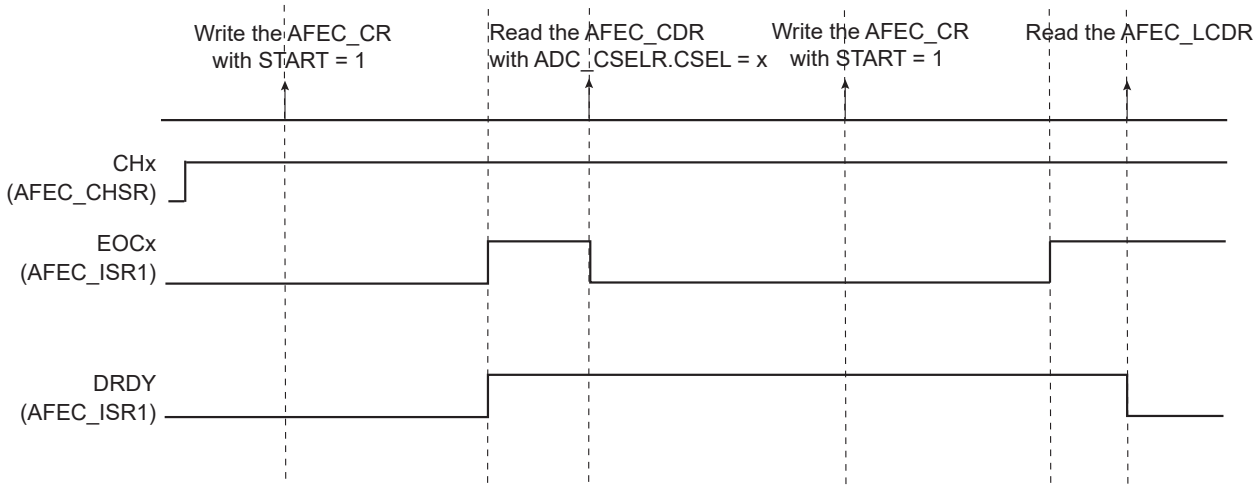
52.6.4 Conversion Results

When a conversion is completed, the resulting 12-bit digital value is stored in an internal register (one register for each channel) that can be read by means of the Channel Data Register (AFEC_CDR) and the Last Converted Data Register (AFEC_LCDR). By setting the bit TAG in the AFEC_EMR, the AFEC_LCDR presents the channel number associated with the last converted data in the CHNB field.

The bits EOCx, where 'x' corresponds to the value programmed in the CSEL bit of AFEC_CSELR, and DRDY in the Interrupt Status Register (AFEC_ISR) are set. In the case of a connected DMA channel, DRDY rising triggers a data transfer request. In any case, either EOCx or DRDY can trigger an interrupt.

Reading the AFEC_CDR clears the EOCx bit. Reading AFEC_LCDR clears the DRDY bit and the EOCx bit corresponding to the last converted channel.

Figure 52-4. EOCx and DRDY Flag Behavior



If AFEC_CDR is not read before further incoming data is converted, the corresponding OVREx flag is set in the Overrun Status Register (AFEC_OVER).

New data converted when DRDY is high sets the GOVRE bit in AFEC_ISR.

The OVREx flag is automatically cleared when AFEC_OVER is read, and the GOVRE flag is automatically cleared when AFEC_ISR is read.

53.7.8 DACC Interrupt Enable Register

Name: DACC_IER
Offset: 0x24
Reset: –
Property: Write-only

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

0: No effect.

1: Enables the corresponding interrupt.

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
			EOC1	EOC0			TXRDY1	TXRDY0
Access			W	W			W	W
Reset			0	–			0	–

Bits 4, 5 – EOCx End of Conversion Interrupt Enable of channel x

Bits 0, 1 – TXRDYx Transmit Ready Interrupt Enable of channel x

For the other resolution defined by RES, the code C_i is extended to the corresponding resolution.

The table below is a computation example for the above formula, where V_{VREFP} = 3V:

Table 58-33. Input Voltage Values in Single-ended Mode

C_i		Gain		
Signed	**Nonsigned**	**1**	**2**	**4**
-2048	0	0	0.75	1.125
0	2047	1.5	1.5	1.5
2047	4095	3	2.25	1.875

58.8.4.3 Example of LSB Computation

The LSB is relative to the analog scale V_{VREFP}.

The term LSB expresses the quantization step in volts, also used for one AFE code variation.

- Single-ended (SE) (ex: V_{VREFP} = 3.0V)
  - Gain = 1, LSB = (3.0V / 4096) = 732 µV
  - Gain = 2, LSB = (1.5V / 4096) = 366 µV
  - Gain = 4, LSB = (750 mV / 4096) = 183 µV
- Differential (DIFF) (ex: V_{VREFP} = 3.0V)
  - Gain = 0.5, LSB = (6.0V / 4096) = 1465 µV
  - Gain = 1, LSB = (3.0V / 4096) = 732 µV
  - Gain = 2, LSB = (1.5V / 4096) = 366 µV

The data include the AFE performances, as the PGA and AFE core cannot be separated. The temperature and voltage dependency are given as separate parameters.

58.8.4.4 Gain and Offset Errors

For:

- a given gain error: E_G (%)
- a given ideal code (C_i)
- a given offset error: E_O (LSB of 12 bits)

in 12-bit mode, the actual code (C_A) is calculated using the following formula

$$C_A = \left(1 + \frac{E_G}{100}\right) \times (C_i - 2047) + 2047 + E_O$$

For higher resolutions, the code can be extended to the corresponding resolution defined by RES.

58.8.4.4.1 Differential Mode

In Differential mode, the offset is defined when the differential input voltage is zero.