Welcome to E-XFL.COM

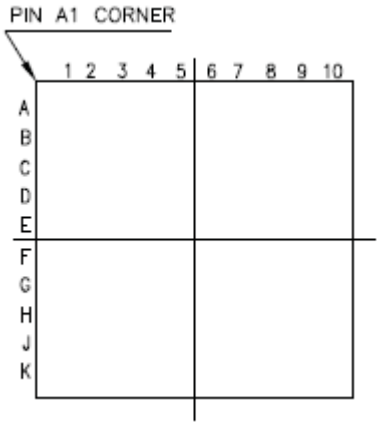
What is "Embedded - Microcontrollers"?

"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	CANbus, Ethernet, I²C, IrDA, LINbus, MMC/SD/SDIO, QSPI, SPI, SSC, 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)	3V ~ 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-TFBGA
Supplier Device Package	100-TFBGA (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsamv71n19b-cb

Figure 6-6. 100-ball VFBGA Package Outline



6.4 100-lead Package Pinout

Table 6-2. 100-lead Package Pinout

LQFP Pin	VFBGA Ball	TFBGA Ball	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		PIO Peripheral D		Reset State
					Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
72	D8	D8	VDDIO	GPIO_AD	PA0	I/O	WKUP0(1)	I	PWMC0_PWMH0	O	TIOA0	I/O	A17/BA1	O	I2SC0_MCK	–	PIO, I, PU, ST
70	C10	C10	VDDIO	GPIO_AD	PA1	I/O	WKUP1(1)	I	PWMC0_PWML0	O	TIOB0	I/O	A18	O	I2SC0_CK	–	PIO, I, PU, ST
66	D10	D10	VDDIO	GPIO	PA2	I/O	WKUP2(1)	I	PWMC0_PWMH1	O	–	–	DATRG	I	–	–	PIO, I, PU, ST
64	F9	F9	VDDIO	GPIO_AD	PA3	I/O	PIODC0(2)	I	TWD0	I/O	LONCOL1	I	PCK2	O	–	–	PIO, I, PU, ST
55	H10	H10	VDDIO	GPIO	PA4	I/O	WKUP3/ PIODC1(3)	I	TWCK0	O	TCLK0	I	UTXD1	O	–	–	PIO, I, PU, ST
52	H9	H9	VDDIO	GPIO_AD	PA5	I/O	WKUP4/ PIODC2(3)	I	PWMC1_PWML3	O	ISI_D4	I	URXD1	I	–	–	PIO, I, PU, ST
24	J2	J2	VDDIO	CLOCK	PA7	I/O	XIN32(4)	I	–	–	PWMC0_PWMH3	–	–	–	–	–	PIO, HiZ
25	K2	K2	VDDIO	CLOCK	PA8	I/O	XOUT32(4)	O	PWMC1_PWMH3	O	AFE0_ADTRG	I	–	–	–	–	PIO, HiZ
54	J9	J9	VDDIO	GPIO_AD	PA9	I/O	WKUP6/ PIODC3(3)	I	URXD0	I	ISI_D3	I	PWMC0_PWMFI0	I	–	–	PIO, I, PU, ST
46	K9	K9	VDDIO	GPIO_AD	PA10	I/O	PIODC4(2)	I	UTXD0	O	PWMC0_PWMEXTRG0	I	RD	I	–	–	PIO, I, PU, ST
44	J8	J8	VDDIO	GPIO_AD	PA11	I/O	WKUP7/ PIODC5(3)	I	QCS	O	PWMC0_PWMH0	O	PWMC1_PWML0	O	–	–	PIO, I, PU, ST
48	K10	K10	VDDIO	GPIO_AD	PA12	I/O	PIODC6(2)	I	QIO1	I/O	PWMC0_PWMH1	O	PWMC1_PWMH0	O	–	–	PIO, I, PU, ST
27	G5	G5	VDDIO	GPIO_AD	PA13	I/O	PIODC7(2)	I	QIO0	I/O	PWMC0_PWMH2	O	PWMC1_PWML1	O	–	–	PIO, I, PU, ST
34	H6	H6	VDDIO	GPIO_CLK	PA14	I/O	WKUP8/ PIODCEN1(3)	I	QSCK	O	PWMC0_PWMH3	O	PWMC1_PWMH1	O	–	–	PIO, I, PU, ST
33	J6	J6	VDDIO	GPIO_AD	PA15	I/O	–	I	D14	I/O	TIOA1	I/O	PWMC0_PWML3	O	I2SC0_WS	–	PIO, I, PU, ST
30	J5	J5	VDDIO	GPIO_AD	PA16	I/O	–	I	D15	I/O	TIOB1	I/O	PWMC0_PWML2	O	I2SC0_DI	–	PIO, I, PU, ST
16	G1	G1	VDDIO	GPIO_AD	PA17	I/O	AFE0_AD6(5)	I	QIO2	I/O	PCK1	O	PWMC0_PWMH3	O	–	–	PIO, I, PU, ST
15	G2	G2	VDDIO	GPIO_AD	PA18	I/O	AFE0_AD7(5)	I	PWMC1_PWMEXTRG1	I	PCK2	O	A14	O	–	–	PIO, I, PU, ST

20. USB Transmitter Macrocell Interface (UTMI)

20.1 Description

The USB Transmitter Macrocell Interface (UTMI) registers manage specific aspects of the integrated USB transmitter macrocell functionality not controlled in USB sections.

20.2 Embedded Characteristics

- 32-bit UTMI Registers Control Product-specific Behavior

32.6 Register Summary

Offset	Name	Bit Pos.								
0x00	PIO_PER	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x04	PIO_PDR	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x08	PIO_PSR	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0C ... 0x0F	Reserved									
0x10	PIO_OER	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x14	PIO_ODR	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x18	PIO_OSR	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x1C ... 0x1F	Reserved									
0x20	PIO_IFER	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x24	PIO_IFDR	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x28	PIO_IFSR	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x2C	Reserved									

Table 34-3. SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns

CPU Address Line																										
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
					Bk[1:0]		Row[11:0]												Column[7:0]								M0
				Bk[1:0]			Row[11:0]												Column[8:0]								M0
			Bk[1:0]				Row[11:0]												Column[9:0]								M0
		Bk[1:0]					Row[11:0]												Column[10:0]								M0

Note: M0 is the byte address inside a 16-bit half-word and Bk[1] = BA1, Bk[0] = BA0.

Table 34-4. SDRAM Configuration Mapping: 8K Rows, 256/512/1024/2048 Columns

CPU Address Line																										
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
				Bk[1:0]		Row[12:0]													Column[7:0]								M0
			Bk[1:0]			Row[12:0]													Column[8:0]								M0
		Bk[1:0]				Row[12:0]													Column[9:0]								M0
	Bk[1:0]					Row[12:0]													Column[10:0]								M0

Note: M0 is the byte address inside a 16-bit half-word and Bk[1] = BA1, Bk[0] = BA0.

34.5 Product Dependencies

34.5.1 SDRAM Device Initialization

The initialization sequence is generated by software. The sequence to initialize SDRAM devices is the following:

1. Set the SDRAM features in the SDRAMC_CR: asynchronous timings (TRC, TRAS, etc.), number of columns, number of rows, CAS latency and data bus width. Set UNAL bit in SDRAMC_CFR1.
2. For mobile SDRAM, configure temperature-compensated self-refresh (TCSR), drive strength (DS) and partial array self-refresh (PASR) in the Low Power register (SDRAMC_LPR).
3. Select the SDRAM memory device type in the Memory Device register (SDRAMC_MDR).
4. A pause of at least 200 µs must be observed before a signal toggle.
5. A NOP command is issued to the SDRAM devices. The application must write a 1 to the MODE field in the Mode register (SDRAMC_MR) (see **Note**). Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM address.
6. An All Banks Precharge command is issued to the SDRAM. The application must write a 2 to the MODE field in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM address.
7. Eight autorefresh (CBR) cycles are provided. The application must set the MODE field to 4 in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM location eight times.

Bit 16 – RWALL Read/Write Allowed
For an OUT pipe, this bit is set when the current bank is not full, i.e., the software can write further data into the FIFO.

For an IN pipe, this bit is set when the current bank is not empty, i.e., the software can read further data from the FIFO.

This bit is cleared otherwise.

This bit is also cleared when the RXSTALLDI or the PERRI bit = 1.

Bits 15:14 – CURRBK[1:0] Current Bank
For a non-control pipe, this field indicates the number of the current bank.

This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll it as an interrupt bit.

Value	Name	Description
0	BANK0	Current bank is bank0
1	BANK1	Current bank is bank1
2	BANK2	Current bank is bank2
3	Reserved	

Bits 13:12 – NBUSYBK[1:0] Number of Busy Banks
This field indicates the number of busy banks.

For an OUT pipe, this field indicates the number of busy banks, filled by the user, ready for an OUT transfer. When all banks are busy, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.

For an IN pipe, this field indicates the number of busy banks filled by IN transaction from the device. When all banks are free, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.

Value	Name	Description
0	0_BUSY	0 busy bank (all banks free)
1	1_BUSY	1 busy bank
2	2_BUSY	2 busy banks
3	3_BUSY	3 busy banks

Bits 9:8 – DTSEQ[1:0] Data Toggle Sequence
This field indicates the data PID of the current bank.

For an OUT pipe, this field indicates the data toggle of the next packet that is to be sent.

For an IN pipe, this field indicates the data toggle of the received packet stored in the current bank.

Value	Name	Description
0	DATA0	Data0 toggle sequence
1	DATA1	Data1 toggle sequence
2	Reserved	
3	Reserved	

Bit 7 – SHORTPACKETI Short Packet Interrupt

42.6.5.3 Read Memory Transfer

The user can access the data of the serial memory by sending an instruction with QSPI_IFR.DATAEN = 1 and QSPI_IFR.TFRTYP = 1.

In this mode, the QSPI is able to read data at random address into the serial Flash memory, allowing the CPU to execute code directly from it (XIP execute-in-place).

In order to fetch data, the user must first configure the instruction frame by writing the QSPI_IFR. Then data can be read at any address in the QSPI address space mapping. The address of the system bus read accesses match the address of the data inside the serial Flash memory.

When Fetch mode is enabled, several instruction frames can be sent before writing QSPI_CR.LASTXFR. Each time the system bus read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the corresponding address.

42.6.5.4 Continuous Read Mode

The QSPI is compatible with the Continuous Read mode which is implemented in some serial Flash memories.

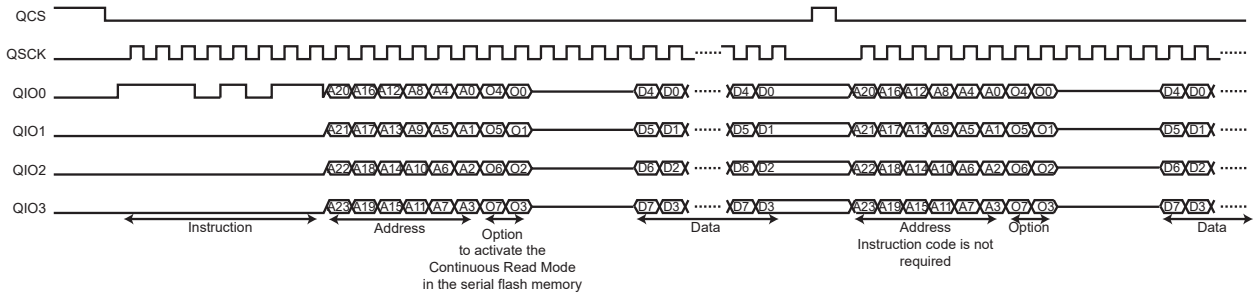
In Continuous Read mode, the instruction overhead is reduced by excluding the instruction code from the instruction frame. When the Continuous Read mode is activated in a serial Flash memory by a specific option code, the instruction code is stored in the memory. For the next instruction frames, the instruction code is not required as the memory uses the stored one.

In the QSPI, Continuous Read mode is used when reading data from the memory (QSPI_IFR.TFRTYP = 1). The addresses of the system bus read accesses are often nonsequential and this leads to many instruction frames that have the same instruction code. By disabling the send of the instruction code, the Continuous Read mode reduces the access time of the data.

To be functional, this mode must be enabled in both the QSPI and the serial Flash memory. The Continuous Read mode is enabled in the QSPI by writing CRM to ‘1’ in the QSPI_IFR (TFRTYP must equal 1). The Continuous Read mode is enabled in the serial Flash memory by sending a specific option code.

⚠ CAUTION If the Continuous Read mode is not supported by the serial Flash memory or disabled, CRM bit must not be written to ‘1’, otherwise data read out of the serial Flash memory is unpredictable.

Figure 42-10. Continuous Read Mode



42.6.5.5 Instruction Frame Transmission Examples

All waveforms in the following examples describe SPI transfers in SPI Clock mode 0 (QSPI_SCR.CPOL = 0 and QSPI_SCR.CPHA = 0; see section Serial Clock Phase and Polarity).

All system bus accesses described below refer to the system bus address phase. System bus wait cycles and system bus data phases are not shown.

Figure 43-7. Master Read with One Data Byte

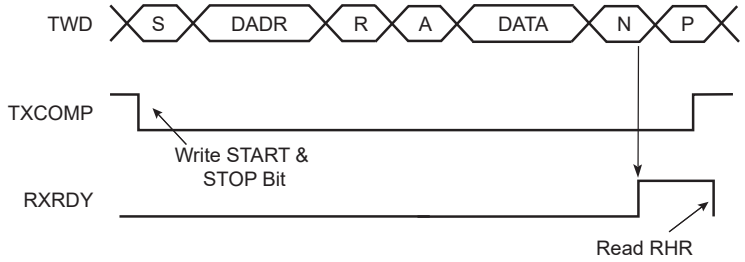


Figure 43-8. Master Read with Multiple Data Bytes

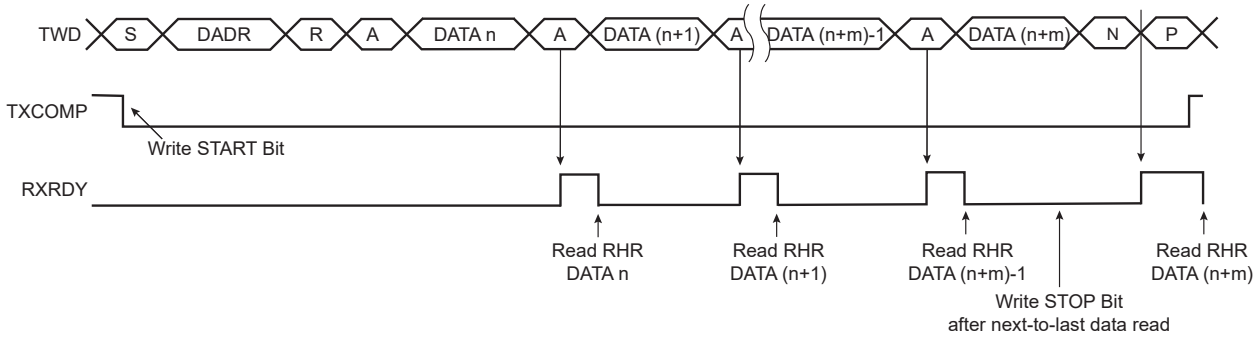
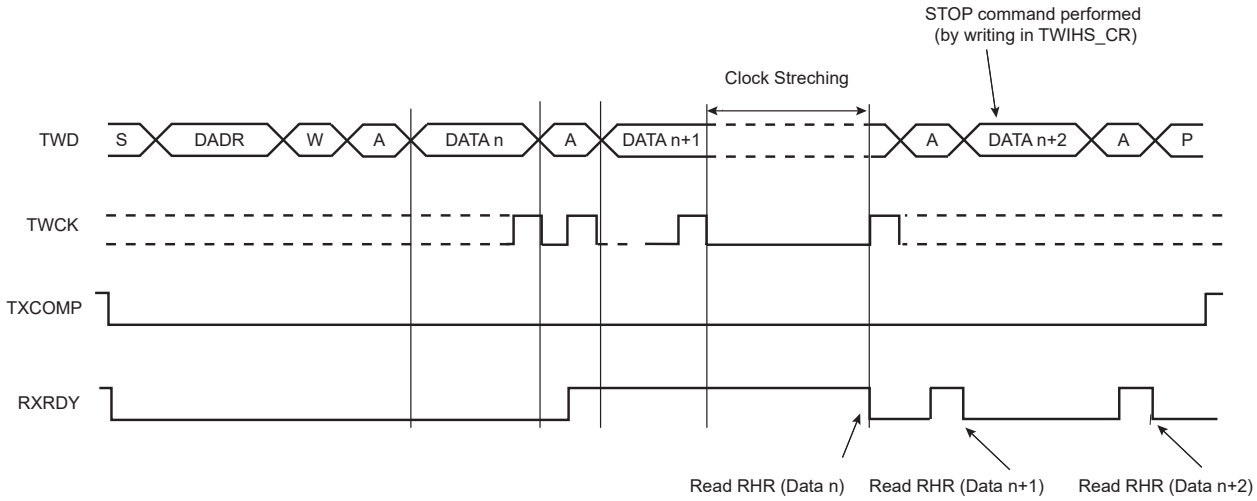


Figure 43-9. Master Read Clock Stretching with Multiple Data Bytes



RXRDY is used as receive ready for the DMA receive channel.

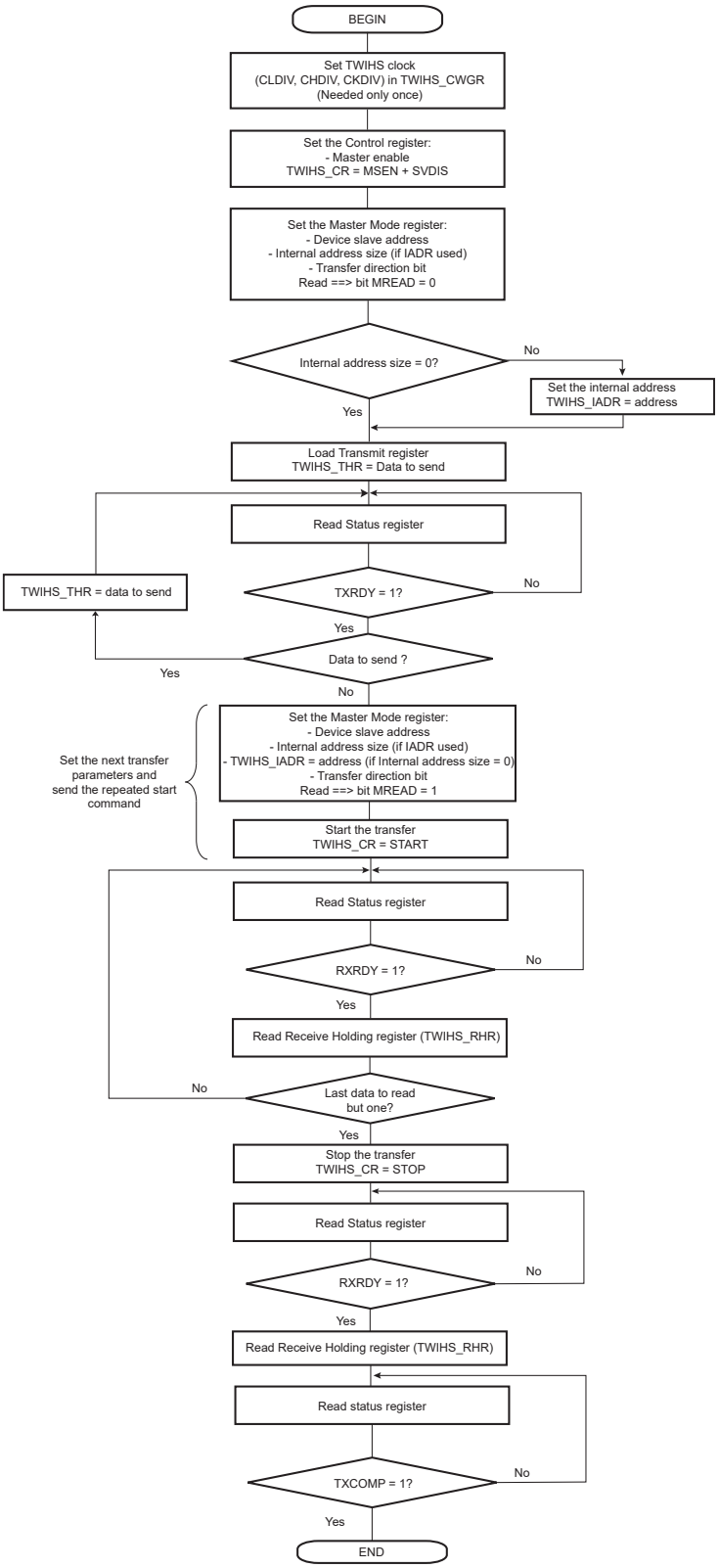
43.6.3.5 Internal Address

The TWIHS can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

43.6.3.5.1 7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g. within a memory page location in a serial memory. When performing read operations with an internal address, the TWIHS performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second START condition (after sending the IADR) is sometimes called “repeated start” (Sr) in I^2C fully-compatible devices. See Master Read with One-, Two- or Three-Byte Internal Address and One Data Byte.

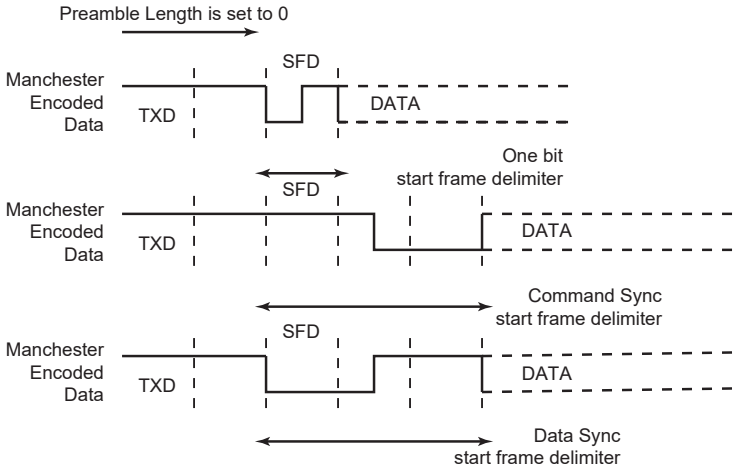
Figure 43-19. TWIHS Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)



45.8 Register Summary

Offset	Name	Bit Pos.								
0x00	I2SC_CR	7:0	SWRST		TXDIS	TXEN	CKDIS	CKEN	RXDIS	RXEN
		15:8								
		23:16								
		31:24								
0x04	I2SC_MR	7:0	FORMAT[1:0]			DATALENGTH[2:0]				MODE
		15:8		TXSAME	TXDMA	TXMONO		RXLOOP	RXDMA	RXMONO
		23:16			IMCKDIV[5:0]					
		31:24	IWS	IMCKMODE	IMCKFS[5:0]					
0x08	I2SC_SR	7:0		TXUR	TXRDY	TXEN		RXOR	RXRDY	RXEN
		15:8							RXORCH[1:0]	
		23:16			TXURCH[1:0]					
		31:24								
0x0C	I2SC_SCR	7:0		TXUR				RXOR		
		15:8							RXORCH[1:0]	
		23:16			TXURCH[1:0]					
		31:24								
0x10	I2SC_SSR	7:0		TXUR				RXOR		
		15:8							RXORCH[1:0]	
		23:16			TXURCH[1:0]					
		31:24								
0x14	I2SC_IER	7:0		TXUR	TXRDY			RXOR	RXRDY	
		15:8								
		23:16								
		31:24								
0x18	I2SC_IDR	7:0		TXUR	TXRDY			RXOR	RXRDY	
		15:8								
		23:16								
		31:24								
0x1C	I2SC_IMR	7:0		TXUR	TXRDY			RXOR	RXRDY	
		15:8								
		23:16								
		31:24								
0x20	I2SC_RHR	7:0	RHR[7:0]							
		15:8	RHR[15:8]							
		23:16	RHR[23:16]							
		31:24	RHR[31:24]							
0x24	I2SC_THR	7:0	THR[7:0]							
		15:8	THR[15:8]							
		23:16	THR[23:16]							
		31:24	THR[31:24]							

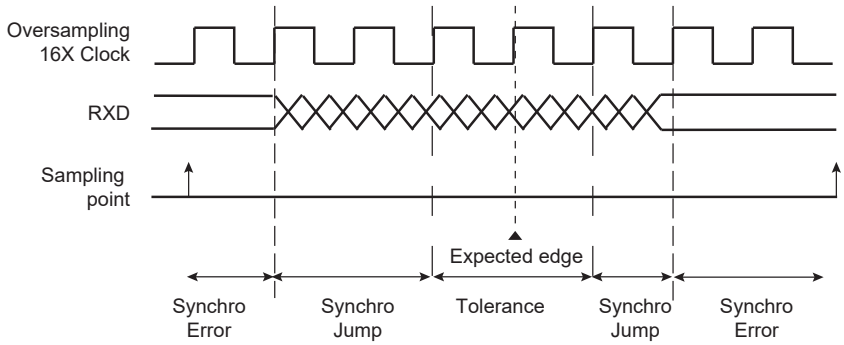
Figure 46-9. Start Frame Delimiter



46.6.3.2.1 Drift Compensation

Drift compensation is available only in 16X Oversampling mode. A hardware recovery system allows a larger clock drift. To enable the hardware system, USART_MAN.DRIFT must be written to ‘1’. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective action is taken. If the RXD event is between 4 and 2 clock cycles before the expected edge, then the current period is shortened by one clock cycle. If the RXD event is between 2 and 3 clock cycles after the expected edge, then the current period is lengthened by one clock cycle. These intervals are considered to be drift and so corrective actions are automatically taken.

Figure 46-10. Bit Resynchronization



46.6.3.3 Asynchronous Receiver

If the USART is programmed in Asynchronous operating mode (SYNC = 0), the receiver oversamples the RXD input line. The oversampling is either 16 or 8 times the baud rate clock, depending on the value of US_MR.OVER.

The receiver samples the RXD line. If the line is sampled during one-half of a bit time to 0, a start bit is detected and data, parity and stop bits are successively sampled on the bit rate clock.

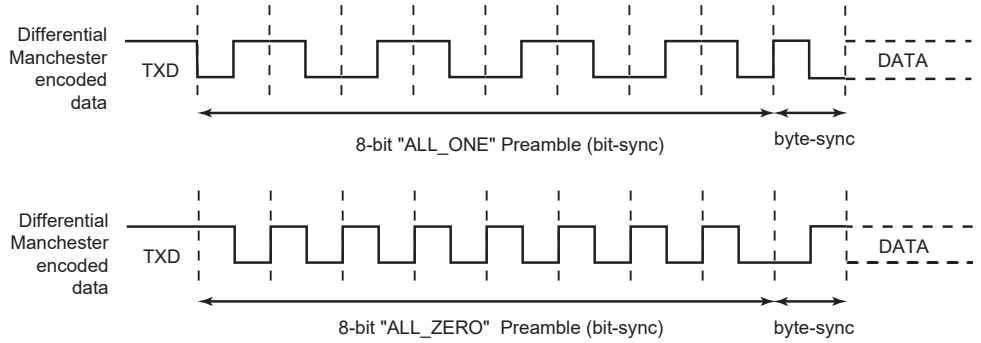
If the oversampling is 16 (OVER = 0), a start is detected at the eighth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 16 oversampling clock cycles. If the oversampling is 8 (OVER = 1), a start bit is detected at the fourth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 8 oversampling clock cycles.

The number of data bits, first bit sent and Parity mode are selected by the same fields and bits as the transmitter, i.e., respectively CHRL, MODE9, MSBF and PAR. For the synchronization mechanism only,

The LON implementation allows two different preamble patterns ALL_ONE and ALL_ZERO which can be configured via US_MAN.TX_PL. The following figure illustrates and defines the valid patterns.

Other preamble patterns are not supported.

Figure 46-57. Preamble Patterns



46.6.10.5.3 Preamble Reception

LON received frames begin with a preamble of variable length. The receiving algorithm does not check the preamble length, although a minimum of length of 4 bits is required for the receiving algorithm to consider the received preamble as valid.

As is the case with LON preamble transmission, two preamble patterns (ALL_ONE and ALL_ZERO) are allowed and can be configured through US_MAN.RX_PL. Figure 46-57 illustrates and defines the valid patterns.

Other preamble patterns are not supported.

46.6.10.5.4 Header Transmission

Each LON frame, after sending the preamble, starts with the frame header also called L2HDR according to the CEA-709 specification. This header consist of the priority bit, the alternative path bit and the backlog increment. It is the first data to be sent.

In LON mode the transmitting algorithm starts when the US_LONL2HDR register is written (it is the first data to send).

46.6.10.5.5 Header Reception

Each LON frame, after receiving the preamble, receives the frame header also called L2HDR according to the CEA-709 specification. This header consists of the priority bit, the alternative path bit, and the backlog increment.

The frame header is the first received data and the RXRDY bit rises as soon as the frame header as been received and stored in the Receive Holding register (US_RHR).

46.6.10.5.6 Data

Data are sent/received serially after the preamble transmission/reception. Data can be either sent/ received MSB first or LSB first depending on US_MR.MSBF.

46.6.10.5.7 CRC

The two last bytes of LON frames are dedicated to CRC.

When transmitting, the CRC of the frame is automatically generated and sent when expected.

When receiving frames the CRC is automatically checked and a LCRCE flag is set in US_CSR if the calculated CRC do not match the received one. Note that the two received CRC bytes are seen as two additional data from the user point of view.

46.7.41 USART LON Beta1 Rx Register

Name: US_LONB1RX
Offset: 0x0078
Reset: 0x0
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

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

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	BETA1RX[23:16]							
Access								
Reset	0	0	0	0	0	0	0	0

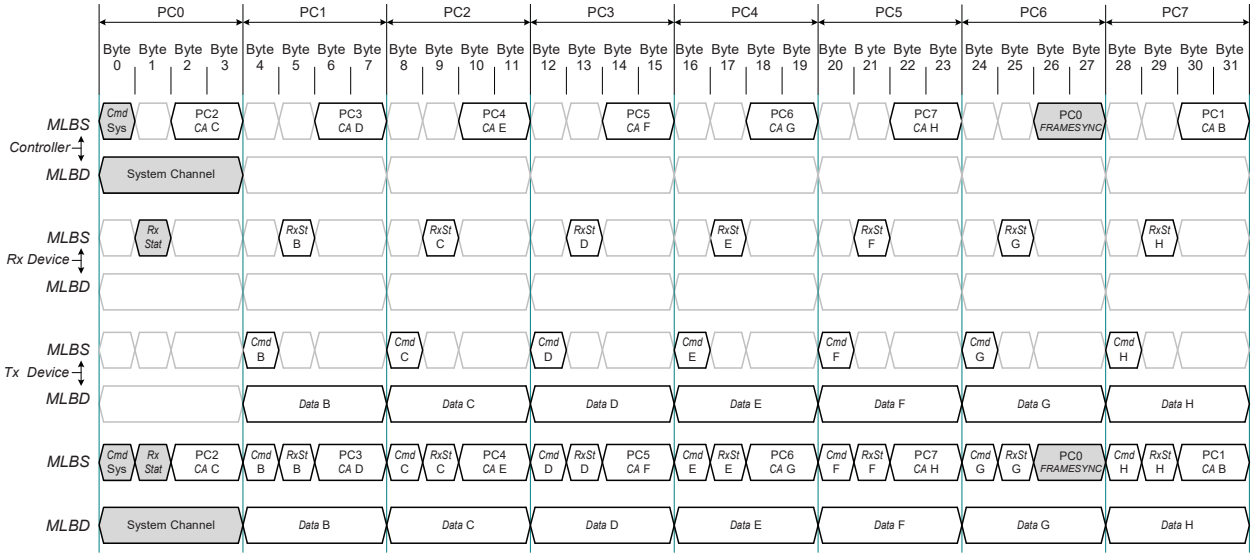
Bit	15	14	13	12	11	10	9	8
	BETA1RX[15:8]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BETA1RX[7:0]							
Access								
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – BETA1RX[23:0] LON Beta1 Length after Reception

Value	Description
1–16777215	LON beta1 length after reception in t_{bit}.

Figure 48-5. 3-pin MediaLB 256Fs Interface Example



48.6.1.7 Initialization

At power up, the MediaLB Controller might output a MLBReset command in the System Channel (all System commands are optional). Upon reception of the MLBReset command, all MediaLB Devices will cancel any current transmissions or receptions and clear their buffers.

Two scenarios are supported to configure MediaLB Devices and ChannelAddresses:

- Static pre-configured before startup. The system implementor decides which ChannelAddresses are to be used for every communication path on MediaLB. This static MediaLB configuration can be communicated by the EHC to the Controller through pre-defined power-up logical channels or through a secondary port.
- Dynamically at run-time. Dynamic configuration allows the board designer to support multiple build options where the EHC can query to find out if a particular Device is present or not on a particular board. The EHC instructs the Controller to scan for a particular DeviceAddress in the System Channel. The Controller uses the MLBScan command to look for a Device. The Controller then notifies the EHC whether the Device is present or not. If the Device is present, then the EHC can instruct the Controller to set the ChannelAddresses for the Device found. The EHC sends messages to the Controller to set each Indices/Logical channel, and waits the appropriate amount of time between each message as specified in the Devices documentation. When that particular Device is configured, the EHC can instruct the Controller to scan for the next Device.

Since the MediaLB Controller is the interface between the MediaLB Devices and the MOST Network, the Controller provides the MLBC signal and will also continue to operate even when the MOST Network is unlocked. When no activity exists on MediaLB, the Controller can shut off the MLBC placing MediaLB in a low-power state. The ChannelAddress assignments are not affected in low-power state; therefore, the same communication paths exists once MLBC is restarted.

MediaLB Devices are synchronously slaved to the MediaLB Controller through the MLBC signal. Since the Controller is synchronized to the MOST Network, the MLBC signal provides Network synchronization to all MediaLB Devices. Once the Controller starts up MLBC, all MediaLB Devices must synchronize to the MediaLB frame before communication can commence. When not frame-locked, Devices must search for the FRAMESYNC pattern, which defines a byte and physical channel boundary. Additionally, the start of the MediaLB frame (PC0) occurs one quadlet after FRAMESYNC is present on the bus. Even when a

Bit 1 – ISOC_BUFO Isochronous Rx Buffer Overflow Enable

Value	Description
1	A buffer overflow on an isochronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. This occurs only when isochronous flow control is disabled.

Bit 0 – ISOC_PE Isochronous Rx Protocol Error Enable

Value	Description
1	A ProtocolError detected on an isochronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.

51.7.22 PWM Output Selection Set Update Register

Name: PWM_OSSUPD
Offset: 0x54
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OSSUPL3	OSSUPL2	OSSUPL1	OSSUPL0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OSSUPH3	OSSUPH2	OSSUPH1	OSSUPH0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – OSSUPLx Output Selection Set for PWML output of the channel x

Value	Description
0	No effect.
1	Output override value OOVLx selected as PWML output of channel x at the beginning of the next channel x PWM period.

Bits 0, 1, 2, 3 – OSSUPHx Output Selection Set for PWMH output of the channel x

Value	Description
0	No effect.
1	Output override value OOVHx selected as PWMH output of channel x at the beginning of the next channel x PWM period.

Figure 53-7. Interpolator Spectral Mask for OSR = 8

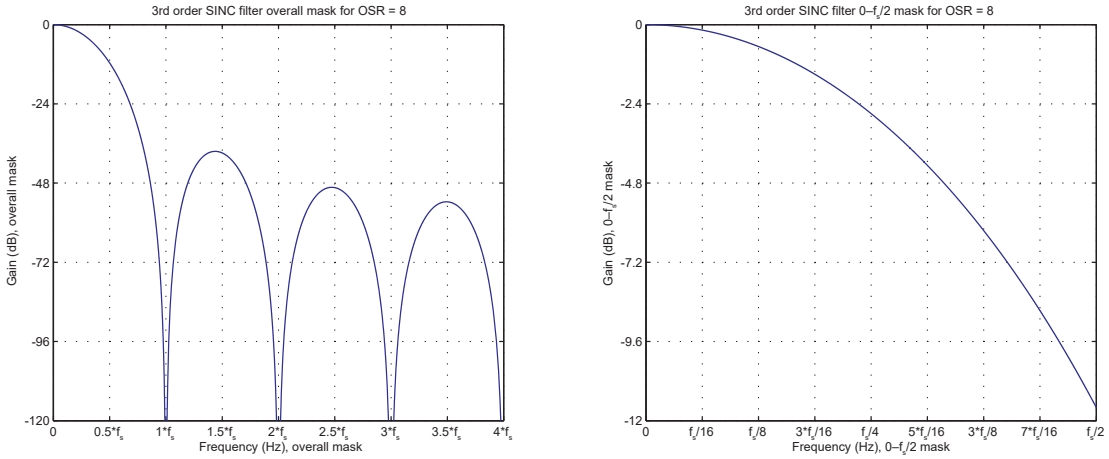


Figure 53-8. Interpolator Spectral Mask for OSR = 16

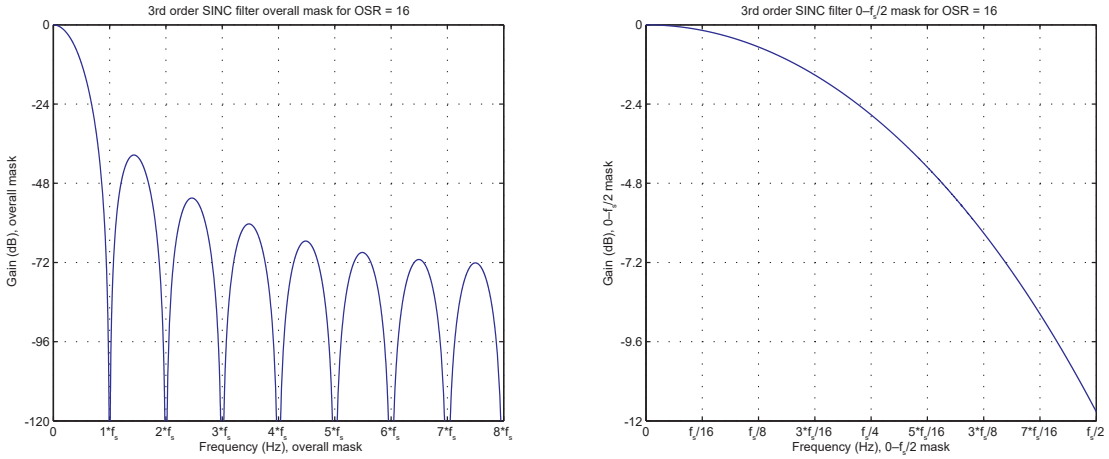
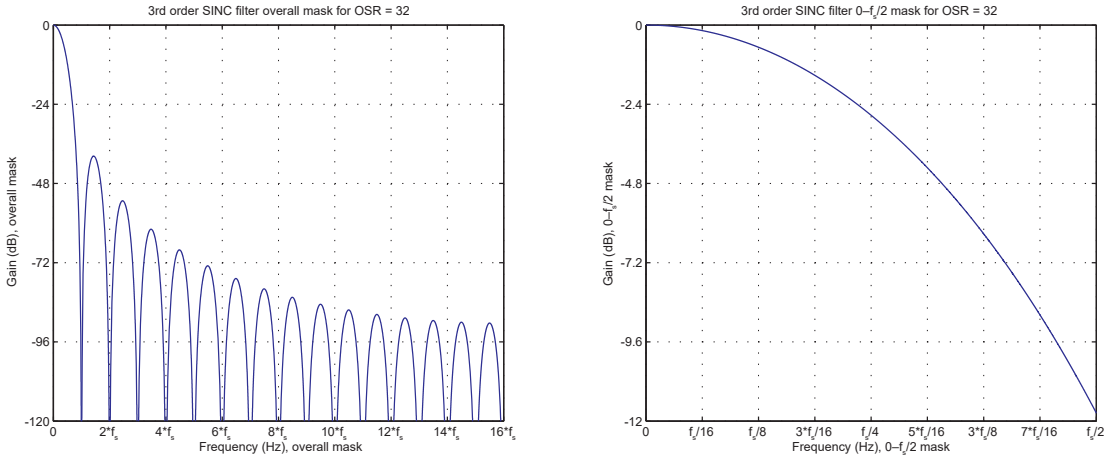


Figure 53-9. Interpolator Spectral Mask for OSR = 32



53.6.5 Conversion FIFO

Each channel embeds a four half-word FIFO to handle the data to be converted.

53.7.14 DACC Write Protection Status Register

Name: DACC_WPSR
Offset: 0xE8
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
	WPVSRC[7:0]							
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
								WPVS
Access								R
Reset								0

Bits 15:8 – WPVSRC[7:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the DACC_WPSR.
1	A write protection violation has occurred since the last read of the DACC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

57.5.12 AES Plaintext/Ciphertext Length Register

Name: AES_CLENR
Offset: 0x74
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	CLEN[31:24]							
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	23	22	21	20	19	18	17	16
	CLEN[23:16]							
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	15	14	13	12	11	10	9	8
	CLEN[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
	CLEN[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 31:0 – CLEN[31:0] Plaintext/Ciphertext Length
Length in bytes of the plaintext/ciphertext (C) data that is to be processed.

Note: The maximum byte length of the C portion of a message is limited to the 32-bit counter length.

2. 10 bits LSB relative to V_{REFP} scale, LSB = V_{VREFP} / 210 = 2.93 mV, with V_{VREFP} = 3V.

58.9 Analog Comparator Characteristics

Table 58-41. Analog Comparator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IR}	Input Voltage Range	–	GND + 0.2	–	VDDIN - 0.2	V
V_{IO}	Input Offset Voltage	Comparator only	–	–	10	mV
I_{VDDIN}	Current Consumption (VDDIN)	Low-power option (ACC_ACR.ISEL = 0)	–	20	–	µA
		High-speed option (ACC_ACR.ISEL = 1)	–	120	–	
V_{hys}	Hysteresis	ACC_ACR.HYST = 1 or 2 ACC_ACR.ISEL = 0	–	20	–	mV
		ACC_ACR.HYST = 3 ACC_ACR.ISEL = 0	–	40	–	
		ACC_ACR.HYST = 1 or 2 ACC_ACR.ISEL = 1	–	25	–	mV
		ACC_ACR.HYST = 3 ACC_ACR.ISEL = 1	–	45	–	
t_S	Settling Time	Overdrive > 100 mV (ACC_ACR.ISEL = 0)	–	–	1.5	µs
		Overdrive > 100 mV (ACC_ACR.ISEL = 1)	–	–	0.15	

58.10 Temperature Sensor

The temperature sensor is connected to channel 11 of the AFE0.

The temperature sensor provides an output voltage (V_{TEMP}) that is proportional to absolute temperature (PTAT).

Improvement of the raw performance of the temperature sensor acquisition can be achieved by performing a single temperature point calibration to remove the initial inaccuracies (V_{TEMP} and ADC offsets).

Table 58-42. Temperature Sensor Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{TEMP}	Output Voltage via AD11	T_A = 25°C	0.64	0.72	0.8	V
dV_{TEMP}/dT	Temperature Sensitivity (Slope Voltage versus Temperature)	–	2.06	2.33	2.60	mV/°C

Signal Name	Recommended Pin Connection	Description
		WARNING Power up and power down sequences given in the "Power Considerations" chapter must be respected.
VDDIO	Decoupling/filtering capacitors (100 nF) (1, 2)	Powers the Peripheral I/O lines (Input/Output Buffers), backup part, 1 Kbytes of Backup SRAM, 32 kHz crystal oscillator, oscillator pads Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. Supply ripple must not exceed 30 mVrms for 10 kHz to 10 MHz range. **WARNING** VDDIN and VDDIO must have the same level and must always be higher than VDDCORE. **WARNING** Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected.
VDDUTMII	Decoupling capacitor (100 nF) (1) (2)	Powers the USB transceiver interface. Must be connected to VDDIO. For USB operations, VDDUTMII and VDDIO voltage ranges must be from 3.0V to 3.6V. Must always be connected even if the USB is not used. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. Supply ripple must not exceed 20 mVrms for 10 kHz to 10 MHz range.
VDDPLLUSB	Decoupling/filtering RLC circuit (1)	Powers the UTMI PLL and the 3 to 20 MHz oscillator. For USB operations, VDDPLLUSB should be between 3.0V and 3.6V. The VDDPLLUSB power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLLUSB power supply routing, decoupling and also on bypass capacitors. Supply ripple must not exceed 10 mVrms for 10 kHz to 10 MHz range.
VDDOUT	Left unconnected	Voltage Regulator Output
VDDCORE	Decoupling capacitor (100 nF) (1) (2)	Powers the core, embedded memories and peripherals.