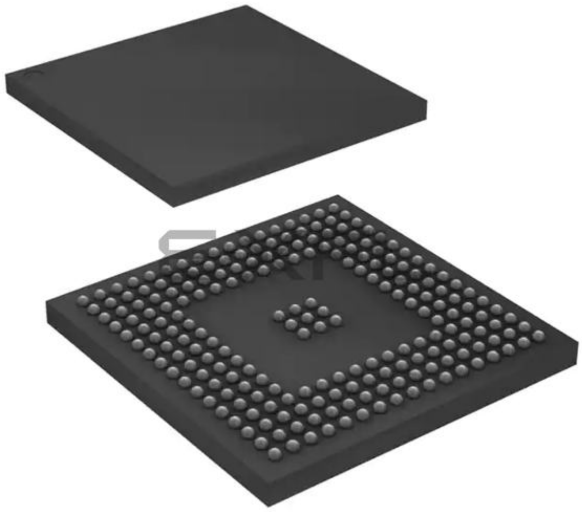


Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Active
Core Processor	ARM926EJ-S
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	400MHz
Co-Processors/DSP	-
RAM Controllers	LPDDR, LPDDR2, DDR2, DDR, SDR, SRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	10/100Mbps (2)
SATA	-
USB	USB 2.0 (3)
Voltage - I/O	1.8V, 2.5V, 3.0V, 3.3V
Operating Temperature	-40°C ~ 85°C (TA)
Security Features	-
Package / Case	217-LFBGA
Supplier Device Package	217-LFBGA (15x15)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at91sam9x25-cu

9.6.6 JTAG ID Code Register

Access: Read-only

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

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part Number is 0x5B2F

- **MANUFACTURER IDENTITY[11:1]**

Set to 0x01F.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

JTAG ID Code value is 0x05B2_F03F.

12.9.7 AIC Interrupt Mask Register

Name: AIC_IMR

Address: 0xFFFFF110

Access: Read-only

Reset: 0x0

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- **FIQ: Interrupt Mask**

0: Corresponding interrupt is disabled.

1: Corresponding interrupt is enabled.

- **SYS: Interrupt Mask**

0: Corresponding interrupt is disabled.

1: Corresponding interrupt is enabled.

- **PID2–PID31: Interrupt Mask**

0: Corresponding interrupt is disabled.

1: Corresponding interrupt is enabled.

- **WDDBGHLT: Watchdog Debug Halt**

0: The watchdog runs when the processor is in debug state.

1: The watchdog stops when the processor is in debug state.

- **WDIDLEHLT: Watchdog Idle Halt**

0: The watchdog runs when the system is in idle state.

1: The watchdog stops when the system is in idle state.

22.6.30 PIO Pad Pull-Down Disable Register

Name: PIO_PPDDR

Address: 0xFFFFF490 (PIOA), 0xFFFFF690 (PIOB), 0xFFFFF890 (PIOC), 0xFFFFFA90 (PIOD)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Pull-Down Disable**

0: No effect.

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

22.6.38 PIO Additional Interrupt Modes Mask Register

Name: PIO_AIMMR

Address: 0xFFFFF4B8 (PIOA), 0xFFFFF6B8 (PIOB), 0xFFFFF8B8 (PIOC), 0xFFFFFAB8 (PIOD)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
P23	P22	P21	P20	P19	P18	P17	P16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
P15	P14	P13	P12	P11	P10	P9	P8
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: IO Line Index**

Selects the IO event type triggering an interrupt.

0: The interrupt source is a both-edge detection event.

1: The interrupt source is described by the registers PIO_ELSR and PIO_FRLHSR.

22.6.49 PIO I/O Drive Register 1

Name: PIO_DRIVER1

Address: 0xFFFFF514 (PIOA), 0xFFFFF714 (PIOB), 0xFFFFF914 (PIOC), 0xFFFFFB14 (PIOD)

Access: Read/Write

31	30	29	28	27	26	25	24
LINE15		LINE14		LINE13		LINE12	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
LINE11		LINE10		LINE9		LINE8	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LINE7		LINE6		LINE5		LINE4	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINE3		LINE2		LINE1		LINE0	

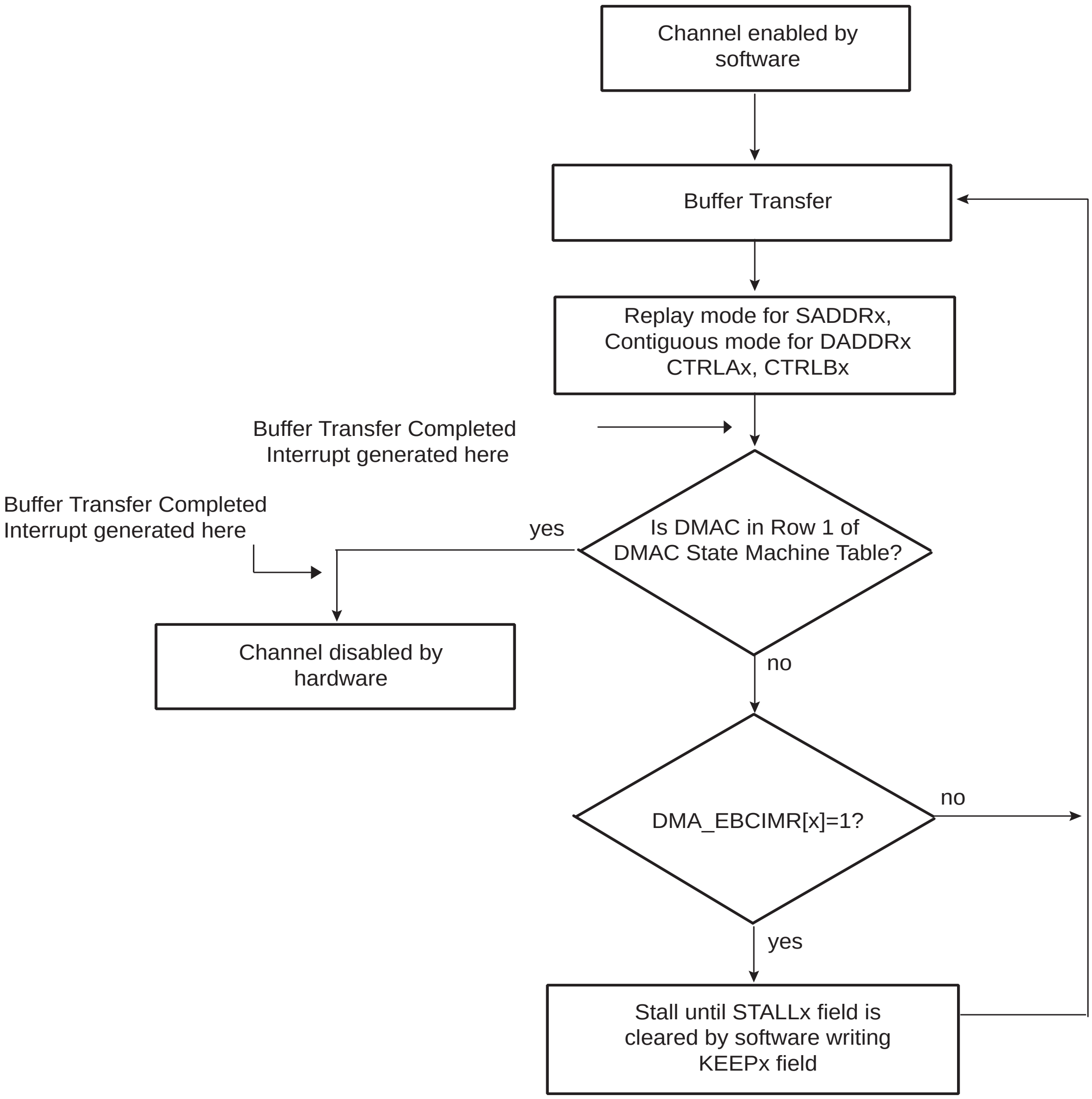
- **LINEx [x=0..15]: Drive of PIO Line x**

Value	Name	Description
0	HI_DRIVE	High drive
1	ME_DRIVE	Medium drive
2	LO_DRIVE	Low drive
3	–	Reserved

24.5.2.2 Round-Robin Arbitration

This algorithm is only used in the highest and lowest priority pools. It allows the Bus Matrix arbiters to properly dispatch requests from different masters to the same slave. If two or more master requests are active at the same time in the priority pool, they are serviced in a round-robin increasing master number order.

Figure 30-14. DMAC Transfer Replay Mode is Enabled for the Source and Contiguous Destination Address



Multi-buffer DMAC Transfer with Linked List for Source and Contiguous Destination Address (Row 2)

1. Read the DMAC_CHSR to choose a free (disabled) channel.
2. Set up the linked list in memory. Write the control information in the LLI.DMAC_CTRLAx and LLI.DMAC_CTRLBx register location of the buffer descriptor for each LLI in memory for channel x. For example, in the register, you can program the following:
   a. Set up the transfer type (memory or non-memory peripheral for source and destination) and flow control device by programming the FC field in DMAC_CTRLBx.
   b. Set up the transfer characteristics, such as:
      – i. Transfer width for the source in the SRC_WIDTH field.
      – ii. Transfer width for the destination in the DST_WIDTH field.
      – v. Incrementing/decrementing or fixed address for source in SRC_INCR field.
      – vi. Incrementing/decrementing or fixed address for destination DST_INCR field.
3. Write the starting destination address in DMAC_DADDRx for channel x.

Note: The values in the LLI.DMAC_DADDRx register location of each Linked List Item (LLI) in memory, although fetched during an LLI fetch, are not used.

30.8.2 DMAC Enable Register

Name: DMAC_EN

Address: 0xFFFFEC04 (0), 0xFFFFEE04 (1)

Access: Read/Write

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

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

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

- **ENABLE: General Enable of DMA**

0: DMA Controller is disabled.

1: DMA Controller is enabled.

31.7.20 UDPHS Endpoint Status Register (Isochronous Endpoint)

Name: UDPHS_EPTSTAx [x=0..6] (ISOENDPT)

Address: 0xF803C11C [0], 0xF803C13C [1], 0xF803C15C [2], 0xF803C17C [3], 0xF803C19C [4], 0xF803C1BC [5], 0xF803C1DC [6]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	BYTE_COUNT						

23	22	21	20	19	18	17	16
BYTE_COUNT				BUSY_BANK_STA		CURBK	

15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW

7	6	5	4	3	2	1	0
TOGGLESQ_STA		–	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x1 in "UDPHS Endpoint Configuration Register" .

- **TOGGLESQ_STA: Toggle Sequencing (cleared upon USB reset)**

Toggle Sequencing:

– **IN endpoint:** It indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.

– OUT endpoint:

These bits are set by hardware to indicate the PID data of the current bank:

Value	Name	Description
0	DATA0	DATA0
1	DATA1	DATA1
2	DATA2	Data2 (only for High Bandwidth Isochronous Endpoint)
3	MDATA	MData (only for High Bandwidth Isochronous Endpoint)

Notes: 1. In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).
2. These bits are updated for OUT transfer:
- A new data has been written into the current bank.
- The user has just cleared the Received OUT Data bit to switch to the next bank.
3. For High Bandwidth Isochronous Out endpoint, it is recommended to check the UDPHS_EPTSTAx/TXRDY_TRER bit to know if the toggle sequencing is correct or not.
4. This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_OVFLW: Overflow Error (cleared upon USB reset)**

This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

31.7.24 UDPHS DMA Channel Control Register

Name: UDPHS_DMACONTROLx [x = 0..5]

Address: 0xF803C308 [0], 0xF803C318 [1], 0xF803C328 [2], 0xF803C338 [3], 0xF803C348 [4], 0xF803C358 [5]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_LENGTH							

23	22	21	20	19	18	17	16
BUFF_LENGTH							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
BURST_LCK	DESC_LD_IT	END_BUFFIT	END_TR_IT	END_B_EN	END_TR_EN	LDNXT_DSC	CHANN_ENB

Note: Channel 0 is not used.

- **CHANN_ENB: (Channel Enable Command)**

0: DMA channel is disabled at and no transfer will occur upon request. This bit is also cleared by hardware when the channel source bus is disabled at end of buffer.

If the UDPHS_DMACONTROL register LDNXT_DSC bit has been cleared by descriptor loading, the firmware will have to set the corresponding CHANN_ENB bit to start the described transfer, if needed.

If the UDPHS_DMACONTROL register LDNXT_DSC bit is cleared, the channel is frozen and the channel registers may then be read and/or written reliably as soon as both UDPHS_DMASTATUS register CHANN_ENB and CHANN_ACT flags read as 0.

If a channel request is currently serviced when this bit is cleared, the DMA FIFO buffer is drained until it is empty, then the UDPHS_DMASTATUS register CHANN_ENB bit is cleared.

If the LDNXT_DSC bit is set at or after this bit clearing, then the currently loaded descriptor is skipped (no data transfer occurs) and the next descriptor is immediately loaded.

1: UDPHS_DMASTATUS register CHANN_ENB bit will be set, thus enabling DMA channel data transfer. Then any pending request will start the transfer. This may be used to start or resume any requested transfer.

- **LDNXT_DSC: Load Next Channel Transfer Descriptor Enable (Command)**

0: No channel register is loaded after the end of the channel transfer.

1: The channel controller loads the next descriptor after the end of the current transfer, i.e., when the UDPHS_DMASTATUS/CHANN_ENB bit is reset.

If the UDPHS_DMA CONTROL/CHANN_ENB bit is cleared, the next descriptor is immediately loaded upon transfer request.

DMA Channel Control Command Summary

LDNXT_DSC	CHANN_ENB	Description
0	0	Stop now
0	1	Run and stop at end of buffer
1	0	Load next descriptor now
1	1	Run and link at end of buffer

- **OPDCMD: Open Drain Command**

0 (PUSHPULL): Push pull command.

1 (OPENDRAIN): Open drain command.

- **MAXLAT: Max Latency for Command to Response**

0 (5): 5-cycle max latency.

1 (64): 64-cycle max latency.

- **TRCMD: Transfer Command**

Value	Name	Description
0	NO_DATA	No data transfer
1	START_DATA	Start data transfer
2	STOP_DATA	Stop data transfer
3	–	Reserved

- **TRDIR: Transfer Direction**

0 (WRITE): Write.

1 (READ): Read.

- **TRTYP: Transfer Type**

Value	Name	Description
0	SINGLE	MMC/SD Card Single Block
1	MULTIPLE	MMC/SD Card Multiple Block
2	STREAM	MMC Stream
4	BYTE	SDIO Byte
5	BLOCK	SDIO Block

- **IOSPCMD: SDIO Special Command**

Value	Name	Description
0	STD	Not an SDIO Special Command
1	SUSPEND	SDIO Suspend Command
2	RESUME	SDIO Resume Command

- **ATACS: ATA with Command Completion Signal**

0 (NORMAL): Normal operation mode.

1 (COMPLETION): This bit indicates that a completion signal is expected within a programmed amount of time (HSMCI_CSTOR).

- **BOOT_ACK: Boot Operation Acknowledge**

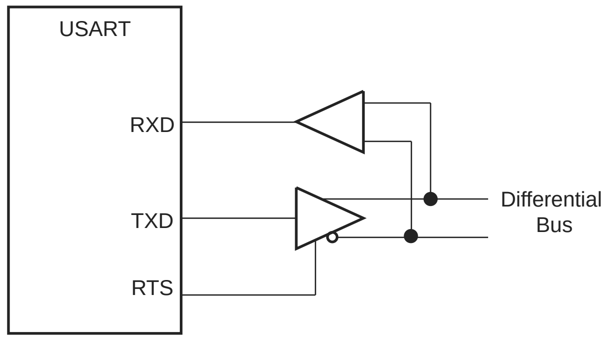
The master can choose to receive the boot acknowledge from the slave when a Boot Request command is issued. When set to one this field indicates that a Boot acknowledge is expected within a programmable amount of time defined with DTOMUL and DTOCYC fields located in the HSMCI_DTOR. If the acknowledge pattern is not received then an acknowledge timeout error is raised. If the acknowledge pattern is corrupted then an acknowledge pattern error is set.

- **ETRGS: External Trigger**

0: The External Trigger Interrupt is disabled.

1: The External Trigger Interrupt is enabled.

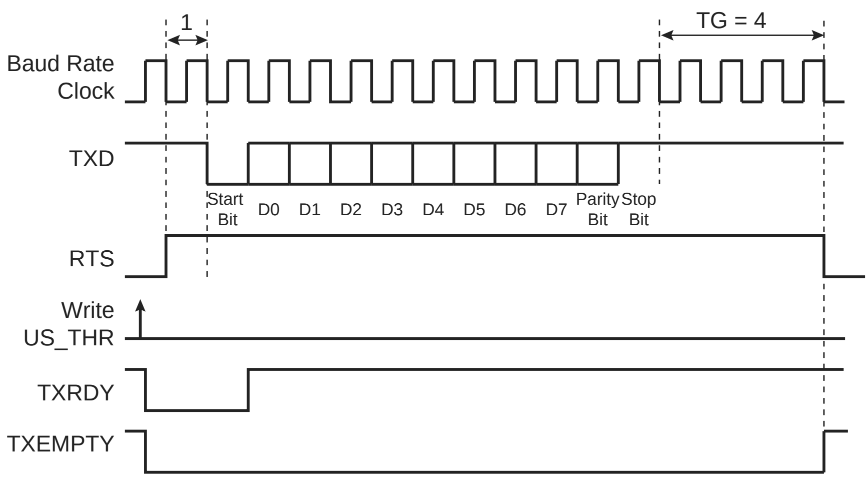
Figure 38-35. Typical Connection to a RS485 Bus



The USART is set in RS485 mode by writing the value 0x1 to the USART_MODE field in US_MR.

The RTS pin is at a level inverse to the TXEMPTY bit. Significantly, the RTS pin remains high when a timeguard is programmed so that the line can remain driven after the last character completion. Figure 38-36 gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

Figure 38-36. Example of RTS Drive with Timeguard



- **FCS: Force SPI Chip Select**

Applicable if USART operates in SPI master mode (USART_MODE = 0xE):

0: No effect.

1: Forces the Slave Select Line NSS (RTS pin) to 0, even if USART is not transmitting, in order to address SPI slave devices supporting the CSAAT mode (Chip Select Active After Transfer).

- **RCS: Release SPI Chip Select**

Applicable if USART operates in SPI master mode (USART_MODE = 0xE):

0: No effect.

1: Releases the Slave Select Line NSS (RTS pin).

38.7.17 USART Receive Holding Register

Name: US_RHR

Address: 0xF801C018 (0), 0xF8020018 (1), 0xF8024018 (2), 0xF8028018 (3)

Access: Read-only

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

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RXSYNH	–	–	–	–	–	–	RXCHR

7	6	5	4	3	2	1	0
RXCHR							

- **RXCHR: Received Character**

Last character received if RXRDY is set.

- **RXSYNH: Received Sync**

0: Last character received is a data.

1: Last character received is a command.

=> SJW = t_{SJW}/t_{CSC} - 1 = 3

Finally: CAN_BR = 0x00053255

CAN Bus Synchronization

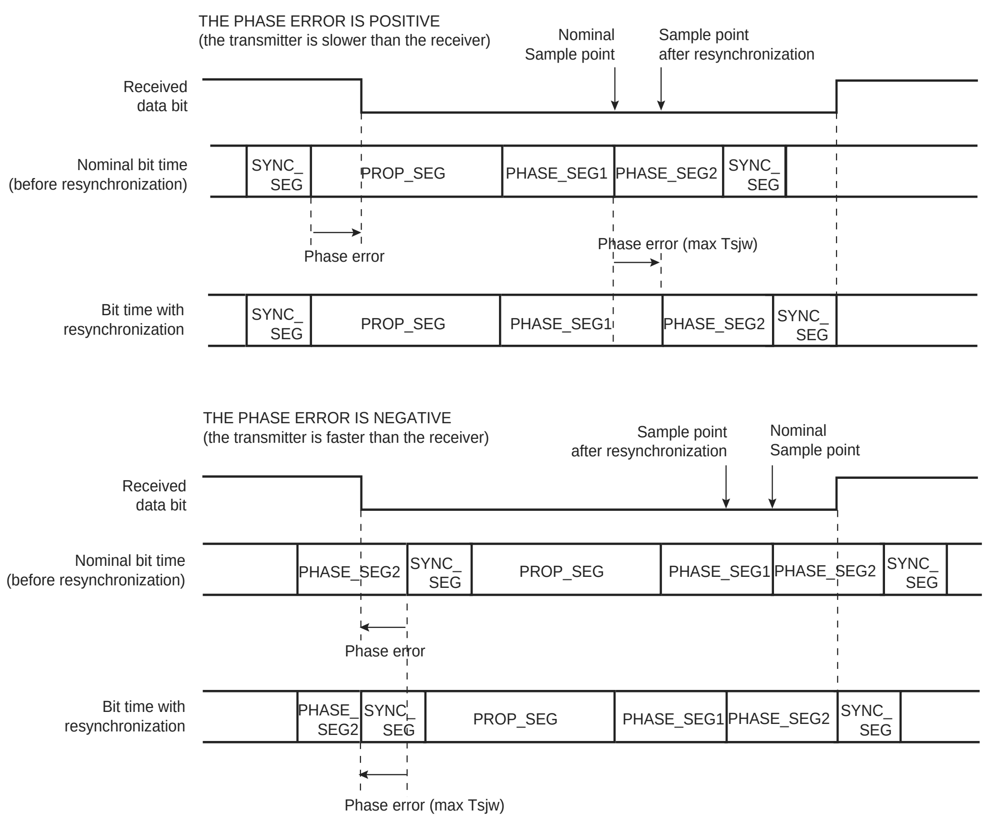
Two types of synchronization are distinguished: “hard synchronization” at the start of a frame and “resynchronization” inside a frame. After a hard synchronization, the bit time is restarted with the end of the SYNC_SEG segment, regardless of the phase error. Resynchronization causes a reduction or increase in the bit time so that the position of the sample point is shifted with respect to the detected edge.

The effect of resynchronization is the same as that of hard synchronization when the magnitude of the phase error of the edge causing the resynchronization is less than or equal to the programmed value of the resynchronization jump width (t_{SJW}).

When the magnitude of the phase error is larger than the resynchronization jump width and

- the phase error is positive, then PHASE_SEG1 is lengthened by an amount equal to the resynchronization jump width.
- the phase error is negative, then PHASE_SEG2 is shortened by an amount equal to the resynchronization jump width.

Figure 40-6. CAN Resynchronization

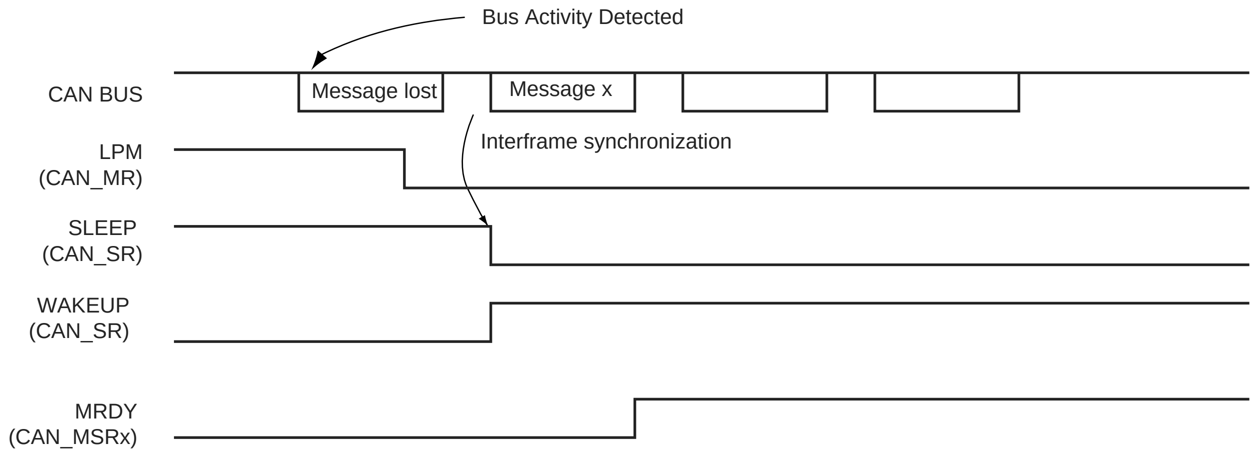


Autobaud Mode

The autobaud feature is enabled by setting the ABM field in the CAN_MR. In this mode, the CAN controller is only listening to the line without acknowledging the received messages. It can not send any message. The errors flags are updated. The bit timing can be adjusted until no error occurs (good configuration found). In this mode, the error counters are frozen. To go back to the standard mode, the ABM bit must be cleared in the CAN_MR.

If there is bus activity when Low-power mode is disabled, the CAN controller is synchronized with the bus activity in the next interframe. The previous message is lost (see Figure 40-9).

Figure 40-9. Disabling Low-power Mode



44.6.5 Receive Buffer Queue Pointer Register

Name: EMAC_RBQP

Address: 0xF802C018

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR						–	–

This register points to the entry in the receive buffer queue (descriptor list) currently being used. It is written with the start location of the receive buffer descriptor list. The lower order bits increment as buffers are used up and wrap to their original values after either 1024 buffers or when the wrap bit of the entry is set.

Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the used bits.

Receive buffer writes also comprise bursts of two words and, as with transmit buffer reads, it is recommended that bit 2 is always written with zero to prevent a burst crossing a 1K boundary, in violation of section 3.6 of the AMBA specification.

- **ADDR: Receive Buffer Queue Pointer Address**

Written with the address of the start of the receive queue, reads as a pointer to the current buffer being used.

44.6.12 PHY Maintenance Register

Name: EMAC_MAN

Address: 0xF802C034

Access: Read/Write

31	30	29	28	27	26	25	24
SOF		RW		PHYA			

23	22	21	20	19	18	17	16
PHYA	REGA					CODE	

15	14	13	12	11	10	9	8
DATA							

7	6	5	4	3	2	1	0
DATA							

Note: To read clause 45 PHYs, bits 31:28 should be written as 0x0011. This overlaps the SOF and RW fields.

- **DATA: PHY Transmit or Receive Data**

For a write operation this is written with the data to be written to the PHY.

After a read operation this contains the data read from the PHY.

- **CODE: Must Be Two**

Must be written to 2. Reads as written.

- **REGA: PHY Register Address**

Specifies the register in the PHY to access.

- **PHYA: PHY Address**

- **RW: PHY Read/Write Command**

1: Write command

2: Read command

Any other value is an invalid PHY management frame.

- **SOF: Start of Frame**

Must be written to one for a valid frame.