CAN_W - Wishbone CAN Controller

Summary

This document provides detailed reference information with respect to the CAN_W peripheral component.

The CAN Controller component (CAN_W) is a stand-alone, Wishbone-compliant controller, for a Controller Area Network conforming to the CAN 2.0B specification. It provides an interface between a processor and a CAN bus. The Controller implements the BOSCH CAN 2.0B Data Link Layer Protocol.

The CAN Controller is compatible with the Philips SJA1000 working in its PeliCAN mode.

Features

- Supports Standard Frame Format (11-bit identifier) and Extended Frame Format (29-bit identifier)
- 64-byte Receive FIFO with 13-byte window (Receive Buffer) pointing to first message to be read from FIFO
- Data Overrun control
- Received Message Counter
- Receive Buffer Start Address Register reflecting the current position where the first byte of the received message is stored
- 13-byte Transmit Buffer
- Compound, programmable Acceptance Filter
  - Single and Dual Acceptance Filter mode
  - 4-byte code
  - 4-byte mask
  - Access to bit-position in message identifier where arbitration lost occurred.
- Error Confinement
  - Error Receive and Transmit Counters with read and write access
  - Programmable Error Warning Limit
  - Access to last error code that informs about type of error that occurred and error position in bit-stream.
- Listen Only Mode – CAN Controller has no influence on CAN bus, no active error flags, no overload flags and no acknowledgments can be sent, but it can receive messages.
- Self Test Mode with Self Reception Request – full node test possibility without any other active node; message acknowledgement is not required and transmission with simultaneous reception possible.
- Single Shot Transmission – no re-transmission required
- Sleep Mode with reduced power consumption.
- Programmable Bit Timing Logic
  - Programmable Bit Rate
  - Programmable ClockOut Period
  - Programmable bit-period length – TSeg1, TSeg2
  - Programmable Synchronization Jump Width
  - Single- and triple- sampling modes.
- Output Modes
  - Normal mode
  - ClockOut mode
  - Biphase mode
  - Test mode

  - Receive interrupt output enable (in normal mode).
- Input Modes
  - Normal mode
  - Test mode.
- Compound and Flexible Interrupt Structure.
- Wishbone-compliant.

Available Components

From a schematic document, the CAN_W component can be found in the FPGA Peripherals (Wishbone) integrated library (`FPGA Peripherals (Wishbone).IntLib`), located in the `\Library\Fpga` folder of the installation.

Designing with the CAN_W

Use of a CAN_W component within an FPGA design is a straightforward procedure – simply place the component and wire it up to the applicable circuitry of the design.

Symbols

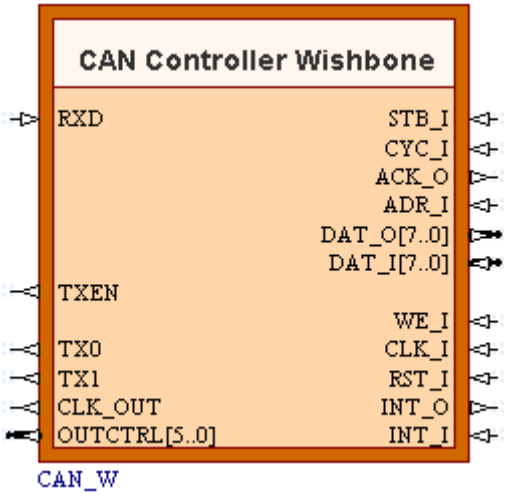


Figure 1. CAN symbol.

Pin Description

The pinout of the CAN Controller has not been fixed to any specific device I/O, thereby allowing flexibility with user application. The CAN Controller contains only unidirectional pins - inputs or outputs.

Table 1. CAN_W pin description

Name	Type	Polarity/Bus size	Description
Control Signals			
CLK_I	I	Rise	External system clock
RST_I	I	High	External system reset. A High state on this pin for two clock cycles while the system clock (CLK_I) is running, resets the component
Host Processor Interface Signals			
STB_I	I	High	Strobe signal. When asserted, indicates the start of a valid Wishbone data transfer cycle
CYC_I	I	High	Cycle signal. When asserted, indicates the start of a valid Wishbone cycle
ACK_O	O	High	Standard Wishbone device acknowledgement signal. When this signal goes high, the Controller (Wishbone Slave) has finished execution of the requested action and the current bus cycle is terminated
ADR_I	I	1	Address bus, used to select an internal Wishbone register of the component for writing to/reading from: 0 = Wishbone Address register (WAREG) 1 = Wishbone Data register (WDREG)

Name	Type	Polarity/Bus size	Description
DAT_O	O	8	Data to be sent to host processor
DAT_I	I	8	Data received from host processor
WE_I	I	Level	Write enable signal. Used to indicate whether the current local bus cycle is a Read or Write cycle: 0 = Read 1 = Write
INT_O	O	High	Interrupt output
INT_I	I	High	Interrupt input
CAN BUS Interface Signals			
RXD	I	-	Input from the physical CAN bus line
TXEN	O	Low	This signal is used to enable tristate output buffers
TX0, TX1	O	-	Outputs from the Controller output drivers to the physical bus line. Function depends on Output Mode selected by OCMODE bits in Output Control Register.
CLK_OUT	O	Rise	Clock output signal. This signal is derived from the system CLK_I signal via the programmable divider. The Clock Off bit within the Clock Divider Register allows this pin to be disabled.
OUTCTRL	O	6	Signals used to control output transistors.

Hardware Description

Block Diagram

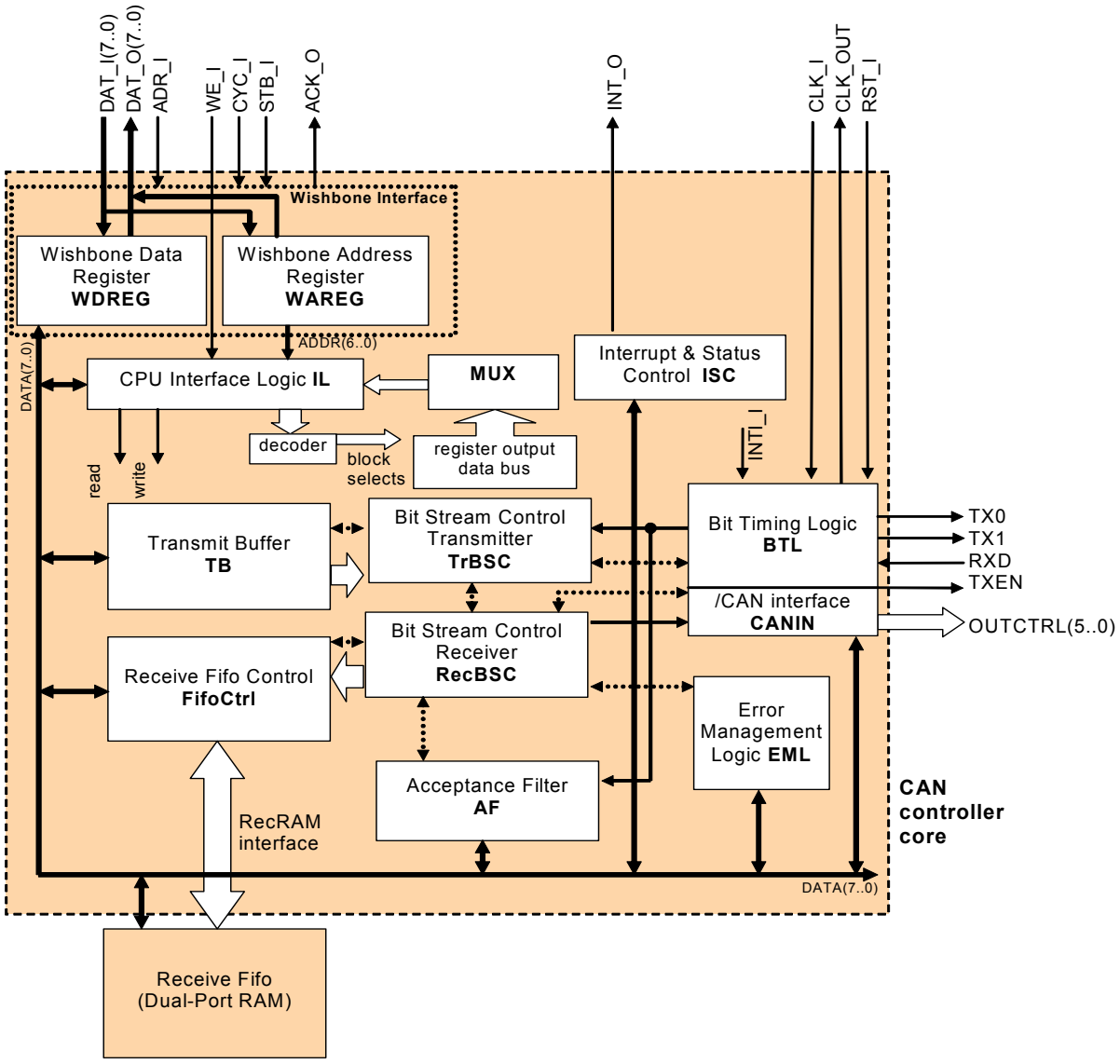


Figure 2. CAN_W block diagram

Architectural overview

FIFO Control Unit (FifoCtrl) with Receiver FIFO (RAM)

The Receive FIFO stores received and accepted messages from the CAN-bus. The Receive Buffer register represents a 13-byte window of the Receive FIFO which itself has a total length of 64 bytes.

With the help of this FIFO, the CPU is able to process one message while other messages are being received. The Receive FIFO is implemented using dual port RAM.

The FIFO Control Unit controls the operation of the Receive FIFO which may implemented either as Receive FIFO or as distributed RAM. Within the FifoCtrl block, there is a Message Counter *(MC)* that informs about the number of messages stored in the Receive FIFO and a Receive Buffer Start Address register (*RBSA),* that informs about the position of the Receive Buffer register window within the Receive FIFO.

In order to shift the RBSA register to the next message position, all message bytes of the current message must be read from the Receive FIFO and the Release Receive Buffer bit subsequently set in the Command Register (CMR.2).

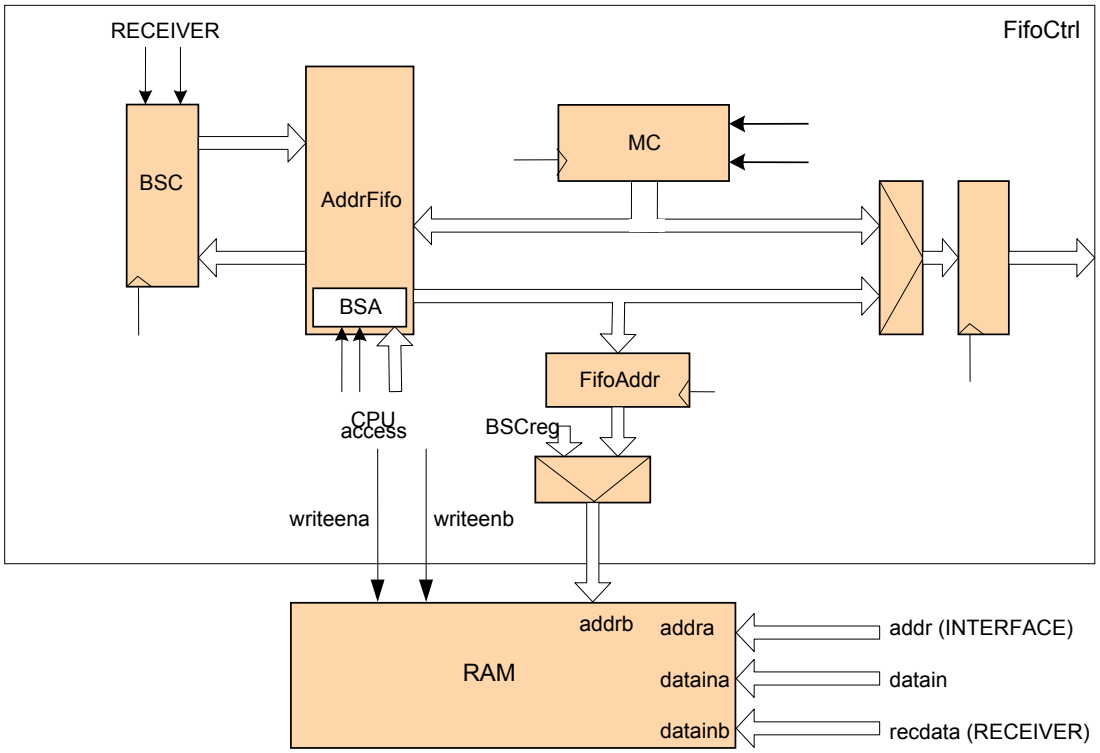


Figure 3. FIFO Control Unit (FifoCtrl) with Receive FIFO (RAM)

Transmit Buffer (TB)

The Transmit Buffer stores a complete message for transmission over the CAN network. The buffer is 13 bytes long. In order to transmit the message, the CPU writes the message into this buffer and sets the Transmission Request bit (CMR.0) or Self Reception Request bit (CMR.4) in the Command Register.

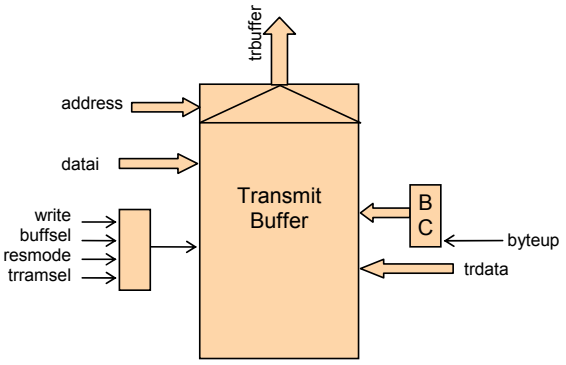


Figure 4. Transmit Buffer (TB)

Acceptance Filter (AF)

The Acceptance Filter compares the received message identifier with the contents of the Acceptance Filter registers and determines whether the message should be accepted and written into the Receive FIFO or not. Within the Acceptance Filter block there are four Acceptance Code registers (ACR0 – ACR3) and four Acceptance Mask registers (AMR0 – AMR3).

Functionality of this block mainly depends on the Acceptance Filter Mode selected in the Mode register (single or dual) and whether the frame format of the message being received is standard or extended.

The Acceptance Filter is controlled by the Bit Stream Control Receiver (RecBSC).

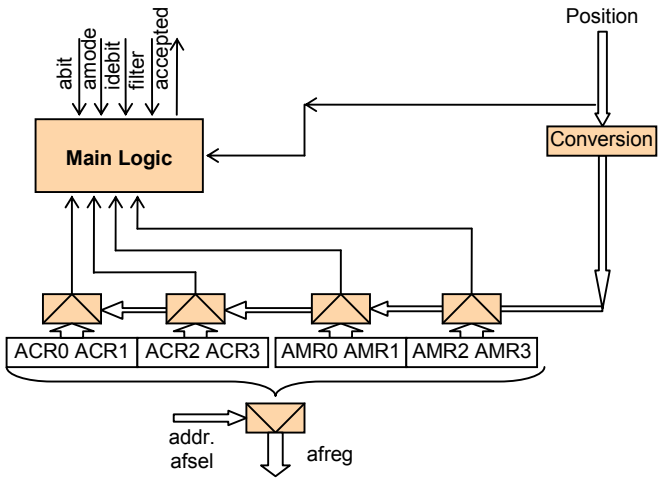


Figure 5. Acceptance Filter (AF)

Bit Timing Logic (BTL)

The Bit Timing Logic block monitors the CAN-bus line and handles the bus line-related bit timing. It is synchronized to the bit stream on the CAN-bus on a 'recessive-to-dominant' bus line transition at the beginning of a message (hard synchronization) and re-synchronized on further transitions during the reception of a message (soft synchronization).

The BTL also provides programmable time segments to compensate for the propagation delay times and phase shifts (e.g. due to oscillator drifts) and to define the sample point and the number of samples to be taken within a bit time. To this end, there are two programmable registers – BTR0 and BTR1.

The BTL also derives the CLK_OUT signal, by dividing the system clock signal (CLK_I) by the number stored in the Clock Divider register (CDR). The CLK_OUT signal can be used as the clock signal for a processor. While entering Sleep Mode, the CLK_OUT signal continues until at least 15 bit times have passed. This allows a host processor clocked by this signal to safely enter its own standby mode before the CLK_OUT signal goes low.

Within the BTL block, there are also programmable registers responsible for output and input mode control and logic that controls waking up the CAN Controller from Sleep Mode.

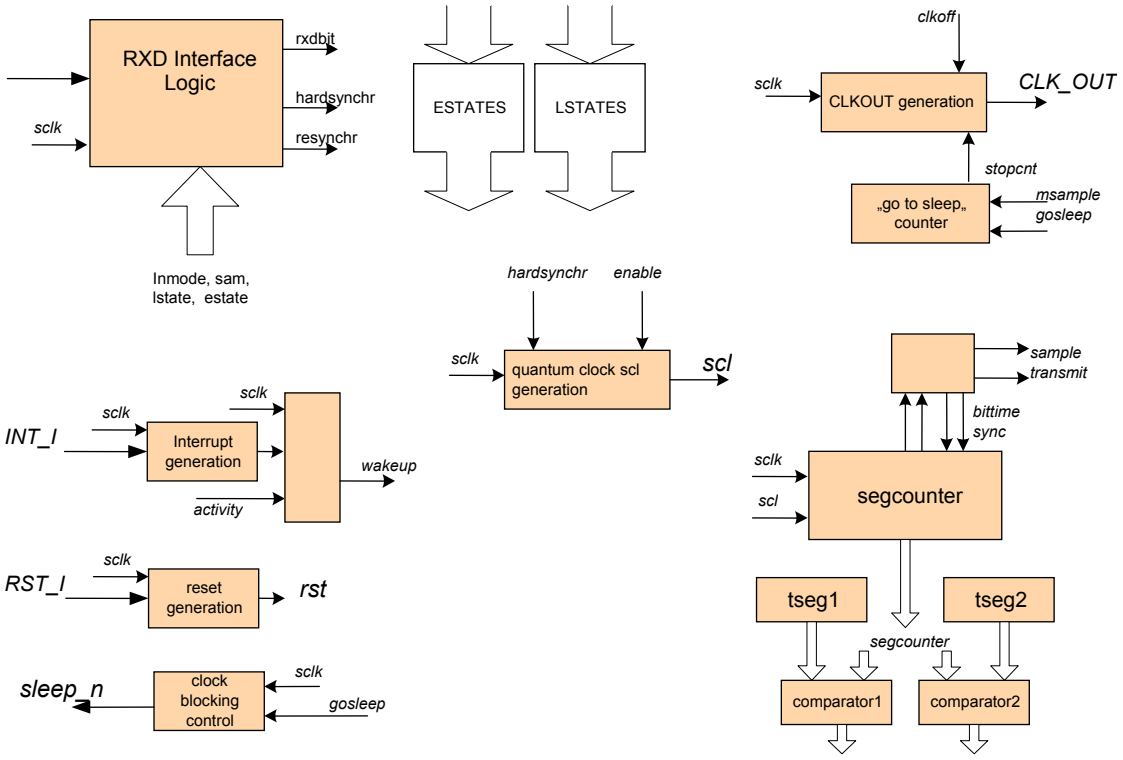


Figure 6. Bit Timing Logic (BTL)

CAN Interface (CANIN)

The CAN Interface is an interface between the CAN-bus line transceiver and the CAN Controller.

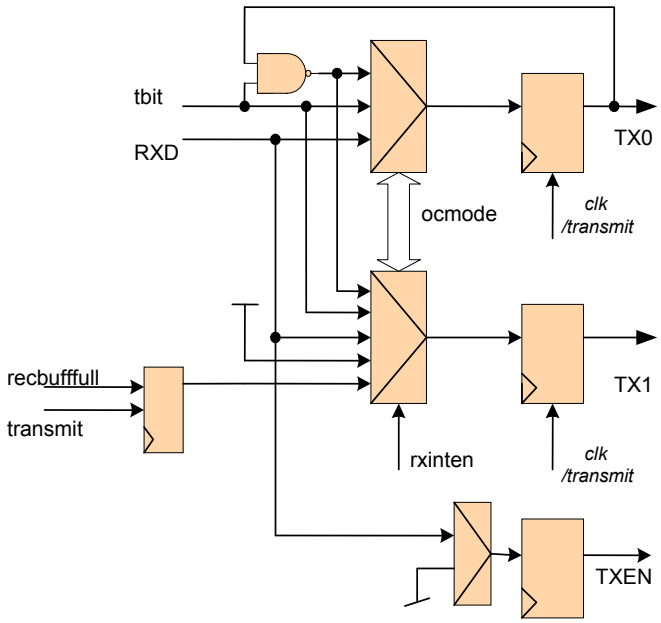


Figure 7. CAN Interface (CANIN)

CPU Interface Logic (IL)

The CPU Interface Logic controls write and read access to the CAN Controller registers from the CPU. Figure 8 illustrates the interface for the CAN_W.

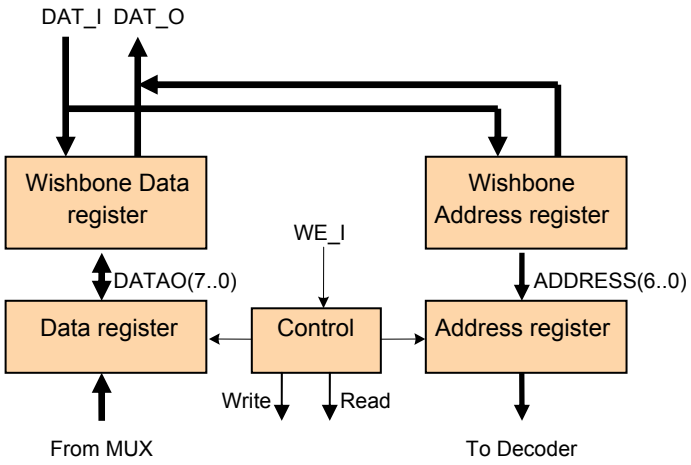


Figure 8. CPU Interface Logic (IL).

Error Management Logic (EML)

The Error Management Logic is responsible for the error confinement of the transfer-layer modules. It receives error announcements from the Receiver and then informs the Interrupt and Status Control Unit (ISC) about error statistics.

The main components of the EML are:

- Receive Error Counter (REC)
- Transmit Error Counter (TEC)
- Error Warning Limit register (EWL).

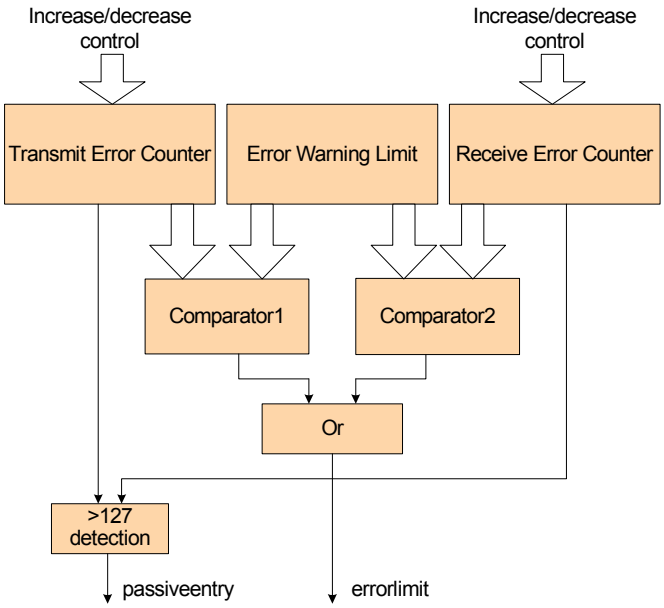


Figure 9. Error Management Logic

Interrupt & Status Control (ISC)

The Interrupt & Status Control unit is responsible for interrupt generation, establishing requested CAN Controller status and interpreting commands in the Command register (CMR).

This block controls all other CAN Controller blocks.

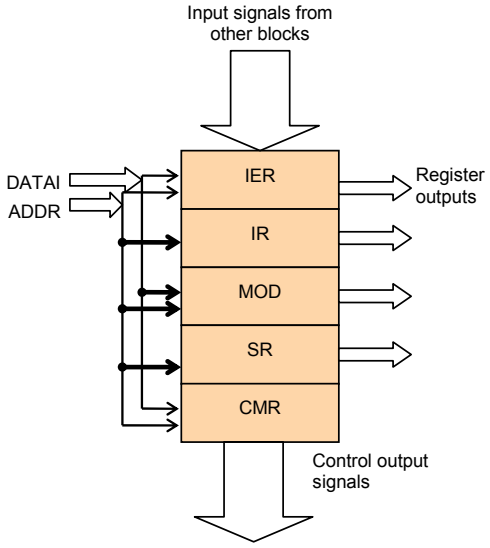


Figure 10. Interrupt & Status Control

Bit Stream Control Receiver (RecBSC)

The Receiver is the main component in the CAN Controller structure. It functions as a Bit Stream Processor. The BSC Receiver monitors the CAN-bus line and controls message reception. This block is responsible for performing BOSCH CAN 2.0B protocol. It performs bus arbitration, CRC calculation and checking, error detection, error signaling and bit destuffing.

The BSC Receiver controls the data stream between the Receiver FIFO and the Bit Timing Logic block. It also derives control signals for the Acceptance Filter, Fifo Control Unit, Transmit Buffer and Error Management Logic.

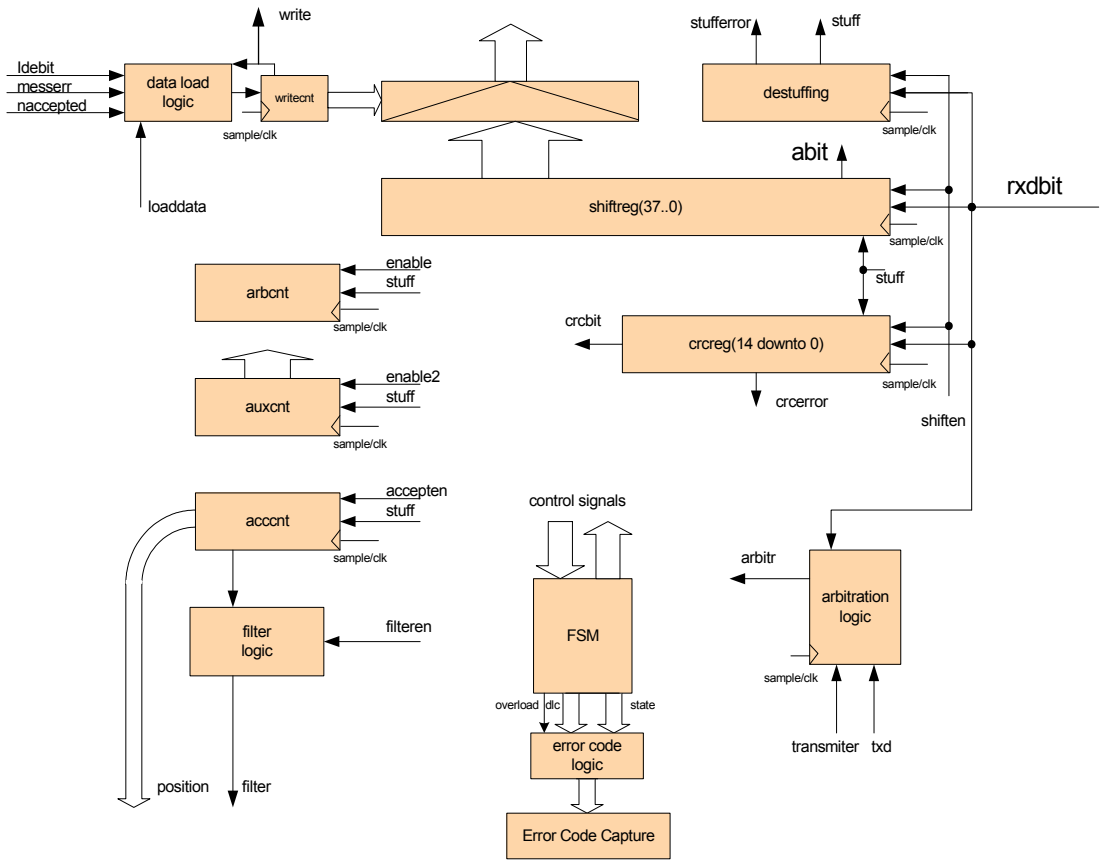


Figure 11. Bit Stream Control Receiver

Bit Stream Control Transmitter (TrBSC)

The main function of the Transmitter Unit is, as its name suggests, message transmission. It also:

- forms frames
- sends acknowledgements, Error and Overload Flags
- performs bit-stuffing.

The Transmitter controls the data stream between the Transmit Buffer and the Bit Timing Logic block.

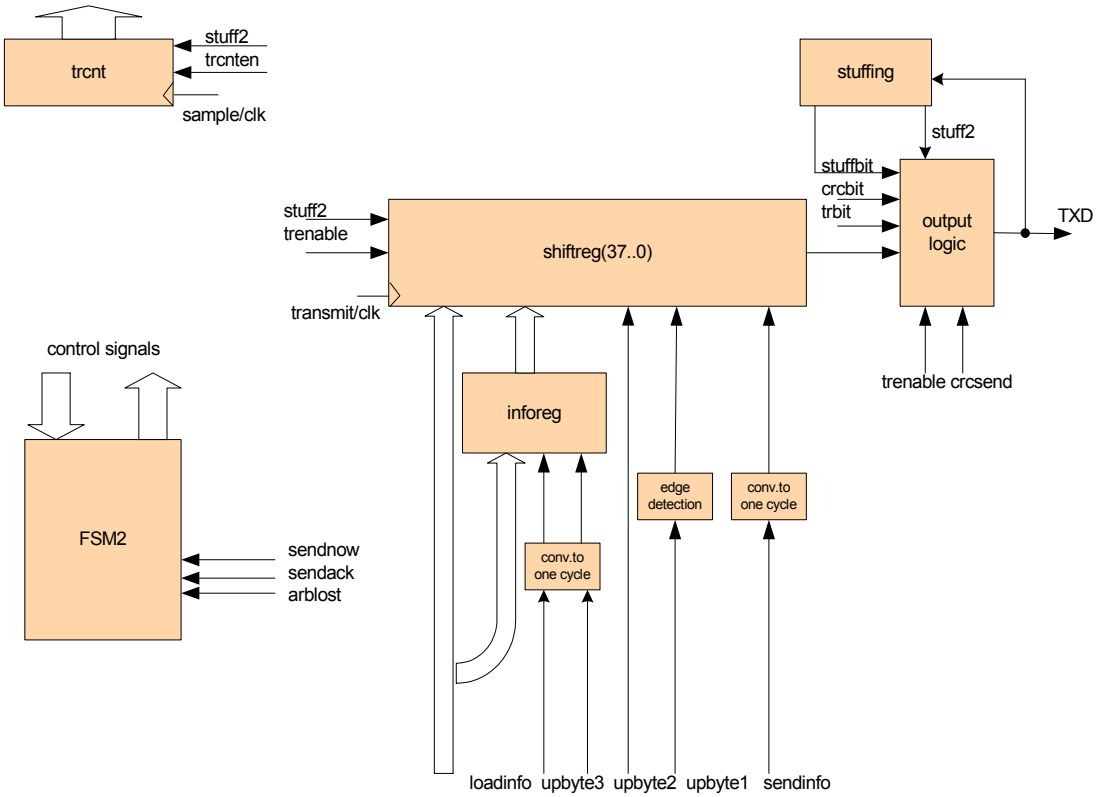


Figure 12. Bit Stream Control Transmitter

Dataflow Diagrams

Main dataflow between Receiver, Transmitter and other CAN blocks

Figure 13 below describes the dataflow between the two main Finite State Machines in the Receiver and Transmitter blocks.

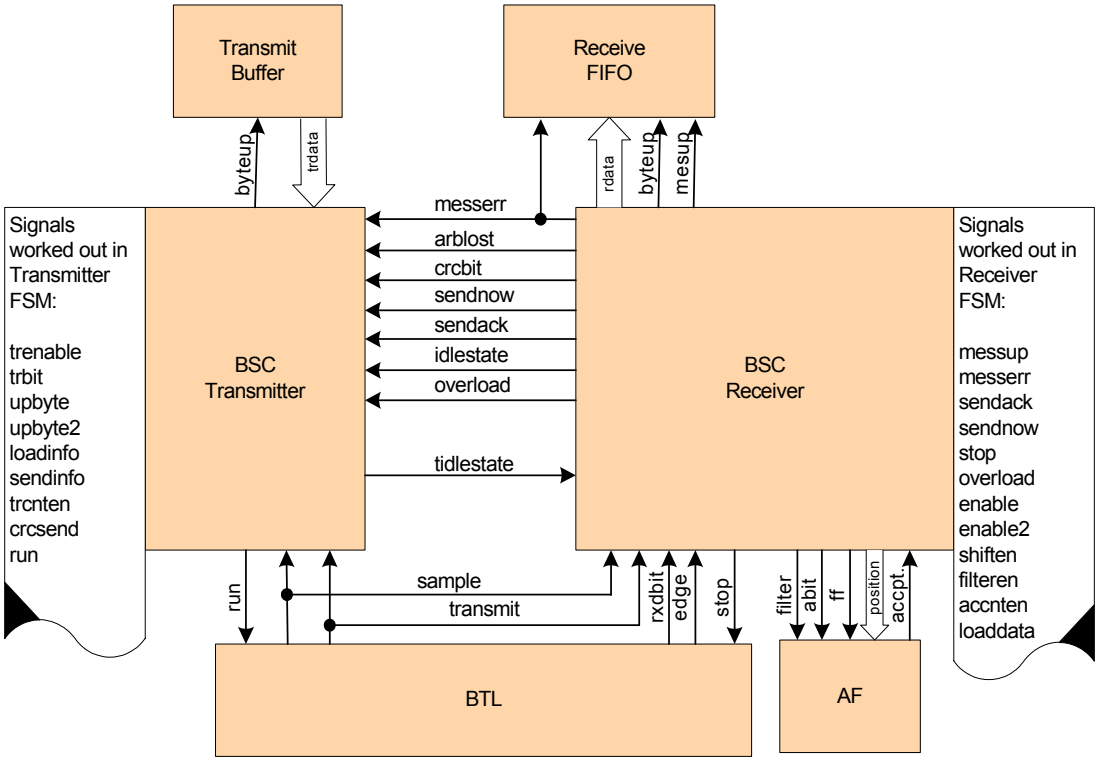


Figure 13. Main dataflow between Receiver and Transmitter

Receiver Finite State Machine Diagram

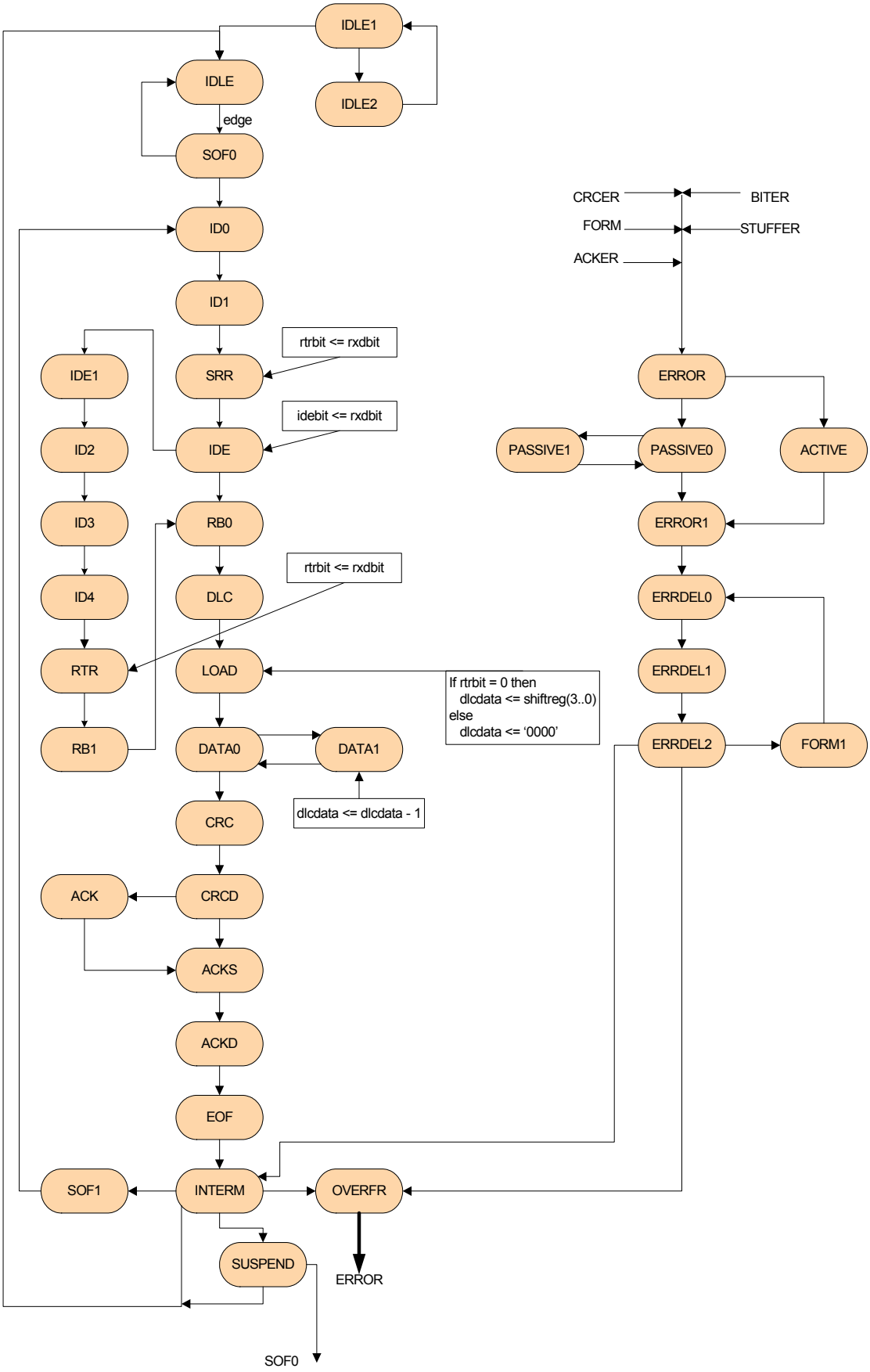


Figure 14. Receiver Finite State Machine diagram

Transmitter Finite State Machine Diagram

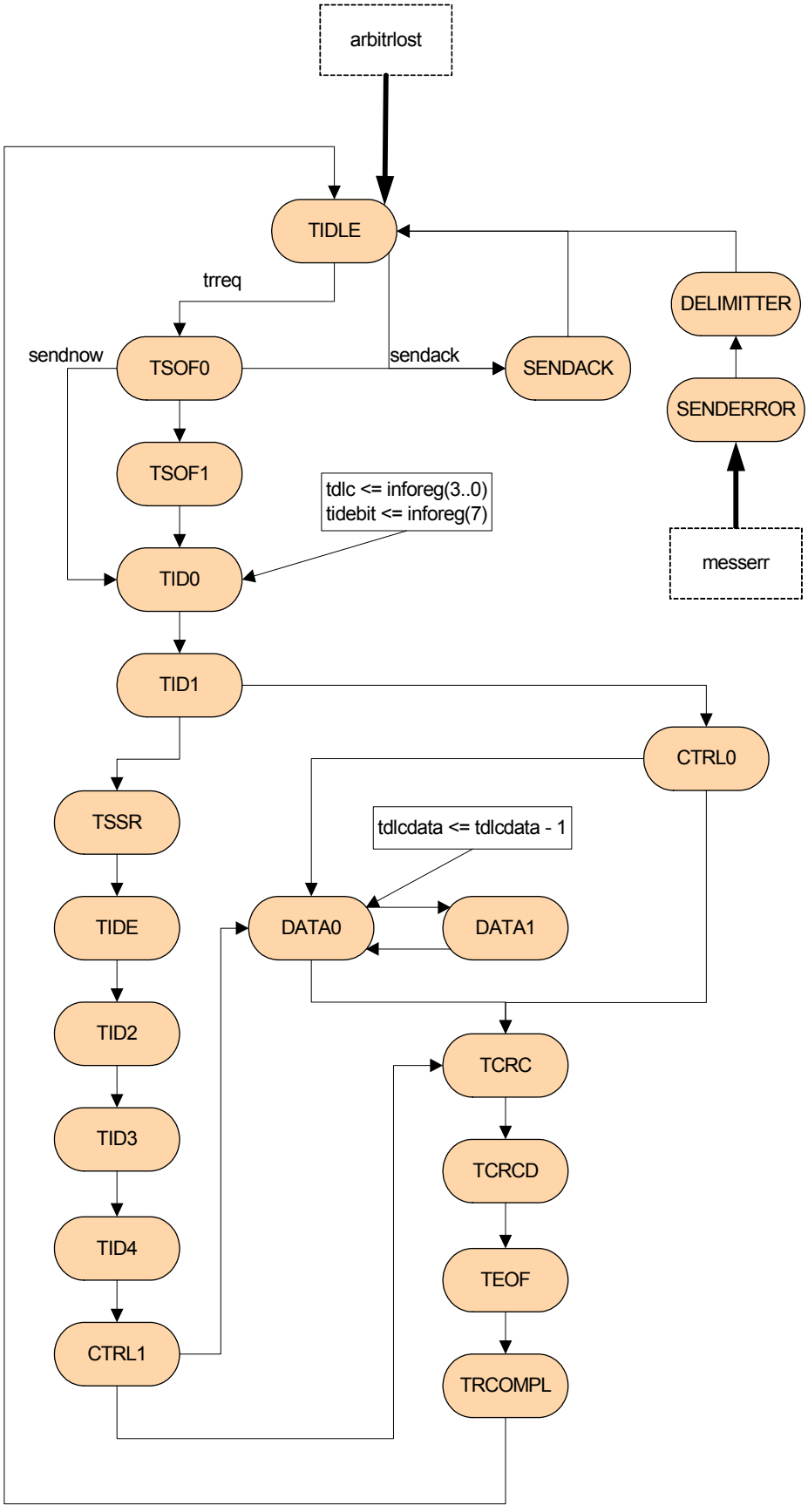


Figure 15. Transmitter Finite State Machine diagram

Internal Wishbone Registers

To simplify communications with internal CAN registers, and to reduce the number of addresses, all Wishbone communication with respect to the CAN_W version of the controller is carried out through two dedicated registers – the Wishbone Address register (WAREG) and Wishbone Data register (WDREG) respectively.

To access any internal CAN register, the host processor must write to the Wishbone Address register with a valid CAN register address and then either write data to or read data from, the Wishbone Data register.

When writing, data will be written directly to the internal CAN register addressed by the Wishbone Address register. When reading, the data in the Wishbone Data register mirrors that currently stored in the internal CAN register address by the Wishbone Address register.

Wishbone Address register (WAREG)

The Wishbone Address register is used to hold the address of the CAN_W's internal CAN register that you wish to write to/read from. The 8-bit address (AB7..AB0) for each internal CAN register is its actual location in CAN address space, as detailed in Table 2.

Table 2. The WAREG register

MSB LSB

AB7	AB6	AB5	AB4	AB3	AB2	AB1	AB0

Table 3. The WAREG register bits description

Bit	Symbol	Function
WAREG.7	AB7	Address bit 7 – this bit is ignored as only a 7-bit address is passed to the CPU Interface Logic block
WAREG.6	AB6	Address bit 6
WAREG.5	AB5	Address bit 5
WAREG.4	AB4	Address bit 4
WAREG.3	AB3	Address bit 3
WAREG.2	AB2	Address bit 2
WAREG.1	AB1	Address bit 1
WAREG.0	AB0	Address bit 0

Wishbone Data register (WDREG)

The Wishbone Data register is a buffer to write to, or read from, any of the CAN_W Controller's internal CAN registers.

Table 4. The WDREG register

MSB LSB

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

Table 5. The WDREG register bits description

Bit	Symbol	Function
WDREG.7	DB7	Data bit 7
WDREG.6	DB6	Data bit 6
WDREG.5	DB5	Data bit 5
WDREG.4	DB4	Data bit 4
WDREG.3	DB3	Data bit 3

Bit	Symbol	Function
WDREG.2	DB2	Data bit 2
WDREG.1	DB1	Data bit 1
WDREG.0	DB0	Data bit 0

Register reset values

Table 6 shows the values contained in both of the CAN_W's internal Wishbone registers after an external system reset has been received on the RST_I input.

Table 6. Internal Wishbone register reset values

Register	Value after reset
WAREG	00h
WDREG	00h

Internal CAN Registers

CAN Registers – summary listing by address

Table 7. Registers list

Location in CAN address space	Parent hardware structure	Register	
00h	Interrupt and Status Control Unit (ISC)	Mode register (MOD)	
01h	Interrupt and Status Control Unit (ISC)	Command register (CMR)	
02h	Interrupt and Status Control Unit (ISC)	Status register (SR)	
03h	Interrupt and Status Control Unit (ISC)	Interrupt register (IR)	
04h	Interrupt and Status Control Unit (ISC)	Interrupt Enable register (IER)	
05h	-	-	
06h	Bit Timing Logic (BTL)	Bus Timing Register 0 (BTR0)	
07h	Bit Timing Logic (BTL)	Bus Timing Register 1 (BTR1)	
08h	Bit Timing Logic (BTL)	Output Control register (OCR)	
09h	-	-	
0Ah	-	-	
0Bh	Acceptance Filter (AF)	Arbitration Lost Capture register (ALC)	
0Ch	Error Management Logic (EML)	Error Code Capture register (ECC)	
0Dh	Error Management Logic (EML)	Error Warning Limit register (EWL)	
0Eh	Error Management Logic (EML)	RX Error Counter register (REC)	
0Fh	Error Management Logic (EML)	TX Error Counter register (TEC)	
10h - 13h	Transmit Buffer (TB) / FIFO Control Unit / Acceptance Filter (AF)	Acceptance Code registers (ACR0 – ACR3)	Transmit Buffer register (TB0 – TB12) (read only) Receive Buffer register (RB0 – RB12) (read only)
14h - 17h	Transmit Buffer (TB) / FIFO Control Unit / Acceptance Filter (AF)	Acceptance Mask registers (AMR0 – AMR3)	
18h - 1Ch	Transmit Buffer (TB) / FIFO Control Unit	-	
1Dh	FIFO Control Unit	Message Counter (MC)	
1Eh	FIFO Control Unit	Receive Buffer Start Address register (RBSA)	
1Fh	Bit Timing Logic (BTL)	Clock Divider register (CDR)	
20h - 5Fh	FIFO Control Unit	Receive Buffer register (RB0 – RB12) (direct read/write)	
60h - 6Ch	Transmit Buffer (TB)	Transmit Buffer register (TB0 – TB12) (direct read/write)	
6Dh - 6Fh	-	-	
70h - 7Fh	-	-	

CAN registers – detailed reference

Mode register (MOD)

Table 8. Mode register

Bit	Name	Function
MOD.7 – MOD.5	-	Not used (when read – 000)
MOD.4	Sleep Mode	0 – Sleep Mode disabled 1 – Sleep Mode enabled. Power consumption reduced.
MOD.3	Acceptance Filter Mode	0 – dual filter mode 1 – single filter mode
MOD.2	Self Test Mode	0 – Self Test Mode disabled 1 – Self Test Mode enabled (no acknowledgement is required).
MOD.1	Listen Only Mode	0 – Listen Only Mode disabled 1 – Listen Only Mode enabled. CAN Controller has no influence on the CAN-bus (no acknowledgements, no error/overload frames sent and no message transmission is possible).
MOD.0	Reset Mode	0 – Reset Mode disabled 1 – Reset Mode enabled. This bit of the register is set when either a software or hardware reset has been issued, or when the bus-off state has been entered (SR.7 set).

Notes:

Address in CAN memory - 00h

Mode.1, Mode.2, Mode.3 – write access possible only in Reset Mode.

Mode.4 – write access possible only in Operating Mode.

Mode.0 – write access possible both in Reset Mode and Operating Mode.

Register value after hardware reset – 00000001

Register value after software reset or when in Bus-Off state – 0000xxx1.

Sleep Mode

The Controller will enter sleep mode if the following conditions are met:

- the Sleep Mode bit (MOD.4) is set
- there is no bus activity
- there is no interrupt pending.

After entering Sleep Mode, the CLK_I signal is stopped and the CLK_OUT signal continues until 15 bit times have passed. This allows a host processor clocked via this signal to enter its own standby mode before the CLK_OUT goes inactive.

The Controller will wake up when one of the following conditions occurs:

- there is bus activity
- pin INTI is driven HIGH (active).

On wake-up, CLK_I is started and a wake-up interrupt is generated, if enabled. If the CAN Controller wakes up due to bus activity, it enters into the BUS-OFF state. In this state, it will not be able to receive any message (including the message that caused the Controller to wake up), until it detects 11 consecutive recessive bits (bus-free sequence).

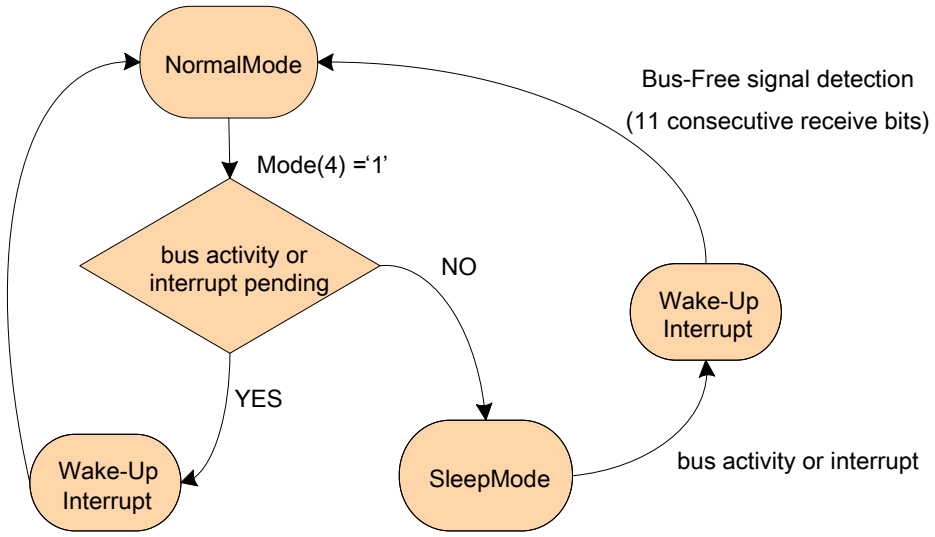


Figure 16. Sleep Mode

Self Test Mode

For proper message transmission, no acknowledge signal is required from the CAN-bus. This mode can be used to test the CAN Controller while there is only one node connected to the Can-bus.

Listen Only Mode

During this mode of operation, the CAN Controller is disconnected from the bus.

Message transmission and message acknowledgements are not possible. This mode of operation also forces the CAN controller to be Error Passive. The Listen Only mode can be used for Can-bus monitoring (e.g. software-driven bit rate detection and 'hot plugging').

Reset Mode

This mode of operation is used for configuring the CAN Controller registers. Detection of a set Reset Mode bit (MOD.0) results in aborting the current transmission/reception of a message and entering the reset mode. On the '1-to-0' transition of the Reset Mode bit, the CAN Controller returns to the mode defined within the Mode register.

During a hardware reset (RST_I) or when the bus status bit is set to logic 1 (Bus-Off), the Reset Mode bit is also set to logic 1.

During a hardware reset, the CPU cannot set the Reset Mode bit to logic 0.

After the Reset Mode bit is set to logic 0, the CAN Controller will wait for:

- One occurrence of bus-free signal (11 consecutive receive bits), if the preceding reset was caused by a hardware reset or a CPU-initiated reset.
- 128 occurrences of the bus-free signal, if the preceding reset was caused by a CAN Controller-initiated Bus-Off, before re-entering the bus-on mode.

Command register (CMR)

Table 9.Command register

Bit	Name	Function
CMR.7 – CMR.5	-	Reserved
CMR.4	Self Reception Request	1 – Message shall be transmitted and received simultaneously
CMR.3	Clear Data Overrun	1 – Clear Data Overrun Status bit (SR.1)
CMR.2	Release Receive Buffer	1 – Completion of reading Receive Buffer indication. Should be set after reading of a received message from the Receive FIFO has completed

Bit	Name	Function
CMR.1	Abort Transmission	1 – Cancel pending transmission
CMR.0	Transmission Request	1 – Message placed in Transmit Buffer shall be transmitted. Should be set after a message has been written into the Transmit Buffer.

Notes:

Address in CAN memory – 01h

This register is Write-only.

Register value returned when read – 0000000

Register value after a reset – 0000000.

Various combinations of bits can be set simultaneously in the Command register, initiating the following actions that have to be performed by the CAN Controller:

- Setting **CMR.0** and **CMR.1** simultaneously results in Single-Shot Transmission.
- Setting **CMR.4** and **CMR.1** simultaneously results in Single-Shot Transmission and simultaneous reception. No re-transmission will be performed in the event of an error or arbitration lost.
- Setting **CMR.0** and **CMR.4** simultaneously will result in self reception transmission (same action as if setting **CMR.4** on its own).

The Transmission Request bit (CMR.0) is cleared automatically either at the moment the Transmit Status bit (SR.5) is set (normal transmission) – signifying successful transmission of a message is in progress - or when the whole message has been read from the Transmit Buffer register by the Transmitter Unit (Single Shot Transmission).

After reading the contents of the Receive Buffer, the CPU can release this memory space in the Receive FIFO (RXFIFO) by setting the Release Receive Buffer bit (CMR.2) to logic 1. This may result in another message becoming immediately available within the Receive Buffer.

The Abort Transmission bit (CMR.1) is used to suspend the previously requested transmission (for example, when you wish to transmit a more urgent message). A transmission already in progress is not stopped.

In order to see whether the previous message has been either transmitted successfully or aborted, the Transmission Complete status bit (SR.3) should be checked. This should be done after the Transmit Buffer Status bit (SR.2) has been set to logic 1 or a transmit interrupt has been generated.

It should be noted that a transmit interrupt is generated even if the message was aborted because the Transmit Buffer Status bit changes to ‘released’.

Status register (SR)

Table 10. Status register

Bit	Name	Function
SR.7	Bus Status	Bus-Off state indication. 1 – Bus-Off state entered Set when Transmit Error Counter exceeds the limit of 255. Simultaneously, the Reset Mode bit (MOD.0) is set and an Error Warning Interrupt is generated, if enabled.
SR.6	Error Status	1 – at least one of the Error Counters has reached or exceeded an Error Warning Limit
SR.5	Transmit Status	CAN Transmitter operation. 1 – CAN is transmitting a message.
SR.4	Receive Status	CAN Receiver operation. 1 – CAN is receiving a message.
SR.3	Transmission Complete	1 – Last requested transmission has been successfully completed.

Bit	Name	Function
		Reset automatically whenever Transmission Request bit (CMR.0) or Self Reception Request bit (CMR.4) is set.
SR.2	Transmit Buffer Status	1 – CPU may write a new message into the Transmit Buffer.
SR.1	Data Overrun Status	Data overrun in Receive FIFO indication. 1 – Message has been lost. Set upon data overrun occurrence. Cleared by receiving the Clear Data Overrun command (CMR.3).
SR.0	Receive Buffer Status	1 – There is at least one message available in the Receive FIFO.

Notes:

Address in CAN memory – 02h

This register is Read-only.

Register value after hardware reset – 00111100

Register value after software reset or in Bus-Off state – xx11x100.

If the CPU tries to write to the Transmit Buffer when the Transmit Buffer Status bit is '0', then the written data will not be accepted and will be lost without any indication.

When a successfully received message passes through the Acceptance Filter but there is not enough space to store the message, then the message is dropped and the data overrun condition is indicated.

In Bus-Off state, when bit SR.7 is set, the Controller enters Reset Mode, the Transmit Error Counter is set to 127 and the Receive Error Counter is cleared. The Controller stays in this mode until the CPU clears the Reset Mode bit (MOD.0). The Controller then waits for 128 occurrences of the bus-free signal counting down the Transmit Error Counter. Only after that Bus Status bit (SR.7) is cleared are the error counters cleared also and an Error Warning Interrupt is generated once again (if enabled).

Errors detected during reception or transmission will effect the error counters according to the CAN 2.0B protocol specification.

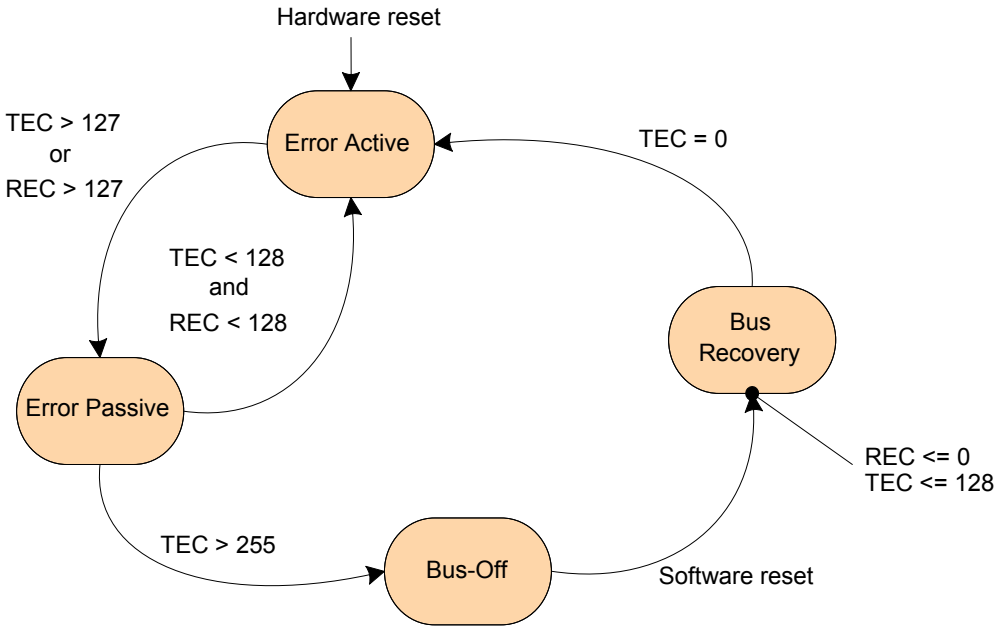


Figure 17. Error states

The CAN Controller can be in one of three error states:

- **Error Active** - the normal operating state. Messages can be received and transmitted. On detecting an error an active error flag is sent.
- **Error Passive** - a state entered when the Controller has frequent problems transmitting or receiving messages. Messages can be received and transmitted. On detecting an error while receiving, a passive error flag is sent.
- **Bus-Off** - entered if the Controller has serious problems with transmitting messages. No messages can be received or transmitted until the CAN Controller is reset by the host processor.

The state of the Controller is controlled by two error counters - the Transmit Error Counter (TEC) and the Receive Error Counter (REC). The following rules apply:

- The CAN Controller is in the Error Active state if TEC <= 127 and REC <= 127
- The Error Passive state is entered if (TEC > 127 or REC > 127) and TEC <= 255
- The Bus-Off state is entered if TEC > 255.

Once the CAN Controller has entered Bus-Off state, it must be reset by the CPU in order to be able to continue operation. In addition, this is only allowed after the reception of 128 occurrences of 11 consecutive recessive bits.

The counters are updated as follows:

1. When a receiver detects an error, the REC will be incremented by 1, except when the detected error was a bit error during the sending of an active error flag or an overload flag.
2. If a receiver detects a bit error while sending an active error flag or an overload flag, the REC is incremented by 8.
3. When a receiver detects a dominant bit as the first bit after sending an error flag, the REC will be incremented by 8.
4. When a transmitter sends an error flag, the TEC is incremented by 8, with the following exceptions:
   - If the transmitter is in the Error Passive state and detects an ack error (due to not detecting a dominant ack) and does not detect a dominant bit while sending its passive error flag.
   - If the transmitter sends an error flag because a stuff error occurred during arbitration - whereby the stuff bit is located before the RTR bit, should have been recessive and has been sent as recessive but monitored as dominant.
5. If a transmitter detects a bit error while sending an active error flag or an overload flag, the TEC is increased by 8.
6. Any node accepts up to 7 consecutive dominant bits after sending an active or passive error flag or an overload flag. After detecting the 14th consecutive dominant bit (in the case of an active error flag or an overload flag), or the 8th consecutive dominant bit following a passive error flag, and after each sequence of additional 8 consecutive dominant bits, every transmitter increases its TEC by 8 and every receiver increases its REC by 8.
7. After the successful transmission of a message (getting acknowledge and no error until end of frame is finished), the TEC is decreased by 1 unless it was already 0.
8. After the successful reception of a message (reception without error up to the AckSlot and the successful sending of the acknowledge bit), the REC is decreased by 1 unless it was already 0.

Note: If a node is the only one on the bus (or during startup the only one that has become active), and it transmits a message, it will get an acknowledgement error and will retransmit the message. This may lead to that node entering the Error Passive state, but not to it entering the Bus-Off state (due to exception 1 under point 4).

Interrupt register (IR))

Table 11. Interrupt register

Bit	Name	Function
IR.7	Bus Error Interrupt	1 – Error on the CAN-bus. Bit, stuff, crc, acknowledge or form error detected.
IR.6	Arbitration Lost Interrupt	1 – CAN Controller lost arbitration and becomes a receiver. Bit position where arbitration has been lost is located within Arbitration Lost Capture register.
IR.5	Error Passive Interrupt	1 – CAN Controller has reached the Error Passive state (at least one Error Counter exceeds 127) or CAN Controller changes error status from Error Passive to Error Active.
IR.4	Wake-Up Interrupt	1 – whilst sleeping, the Controller has detected either CAN-bus activity or an interrupt on the INTI input pin.

Bit	Name	Function
IR.3	Data Overrun Interrupt	1 – Data Overrun Status bit (SR.1) is active.
IR.2	Error Warning Interrupt	1 – Change in either the Error Status (SR.6) or Bus Status (SR.7) bits.
IR.1	Transmit Interrupt	1 – The Transmit Buffer Status bit (SR.2) has changed from ‘0' to '1' (released).
IR.0	Receive Interrupt	1 – Receive FIFO not empty.

Notes:

Address in CAN memory – 03h

This register is Read-only.

Register value after hardware reset – 00000000

Register value after software reset or in Bus-Off state – 00000x00

The Interrupt register allows the identification of an interrupt source.

Activity of each bit depends on the state of the corresponding bit in the Interrupt Enable register. When one or more bits of the Interrupt register are set, an output interrupt will be indicated to the CPU. After this register is read by the CPU all bits are cleared except for the Receive Interrupt bit.

Giving the command to ‘Release Receive Buffer’ (CMR.2) will clear the Receive Interrupt bit temporarily. If there is another message available within the Receive FIFO after the release command, the Receive Interrupt bit is set again. Otherwise it remains cleared.

In the case of a Bus Error Interrupt (when a bus error occurs), the corresponding bus error interrupt is forced, if enabled. At the same time, the current position of the Bit Stream is captured into the Error Code Capture register. The content within this register is fixed until the user’s software has read out its content once. The capture mechanism is then activated again. The corresponding interrupt flag is cleared during the read access to the Interrupt register. A new bus error interrupt is not possible until the capture register is read out once.

Interrupt Enable register (IER)

Table 12. Interrupt Enable register

Bit	Name	Function
IER.7	Bus Error Interrupt Enable	Enable corresponding interrupt bit in Interrupt register.
IER.6	Arbitration Lost Interrupt Enable	
IER.5	Error Passive Interrupt Enable	
IER.4	Wake-Up Interrupt Enable	
IER.3	Data Overrun Interrupt Enable	
IER.2	Error Warning Interrupt Enable	
IER.1	Transmit Interrupt Enable	
IER.0	Receive Interrupt Enable	

Notes:

Address in CAN memory – 04h

Register value after hardware reset – 00000000

Bus Timing registers (BTR0, BTR1)

Addresses in CAN memory – 06h (BTR0), 07h (BTR1),

These registers are Read-only

Register values after hardware reset:

- BTR0 : 00h
- BTR1 : 14h

The Nominal Bit Time, t_{bit}, consists of the four non-overlapping segments SYNC_SEG, PROP_SEG, TSEG1 and TSEG2 with the corresponding time durations t_{SYNC_SEG} , t_{PROP_SEG} t_{TSEG1} and t_{SEG2}. In the CAN-Controller, the segments PROP_SEG and TSEG1 are treated as one segment - TSEG1. This facilitates the programming of bit timing parameters. The only impact of this is that PROP_SEG must be considered while calculating parameters , before the Bit Timing Parameters are programmed.

Mathematically, the time duration of the Nominal Bit Time, t_{bit} , is simply the sum of the individual segments' durations.

$$t_{bit} = t_{SYNC\_SEG} + t_{PROP\_SEG} + t_{SEG1} + t_{SEG2}$$

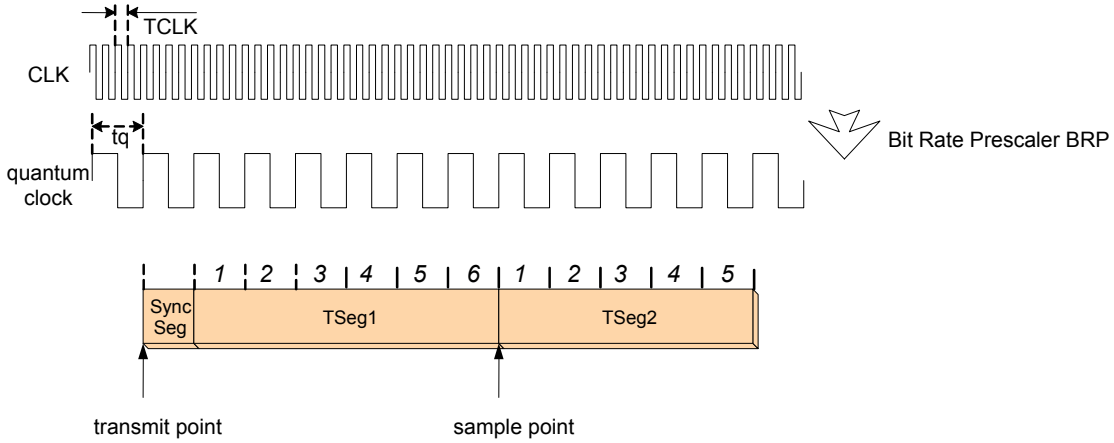


Figure 18. Bus Timing

BTR0.7	BTR0.6	BTR0.5	BTR0.4	BTR0.3	BTR0.2	BTR0.1	BTR0.0
SJW.1	SJW.0	BRP.5	BRP.4	BRP.3	BRP.2	BRP.1	BRP.0

tq – time period of the quantum clock

BRP – Bit Rate Prescaler. The system clock (CLK_I) is divided by BRP to determine the quantum clock

$$t_q = (BRP + 1) * 2t_{CLK\_I}$$
$$BRP = 32\,BRP.5 + 16\,BRP.4 + 8\,BRP.3 + 4\,BRP.2 + 2\,BRP.1 + BRP.0$$

SJW – Synchronization Jump Width

SJW defines the maximum number of time quanta a bit period may be shortened or lengthened by one re-synchronization, to compensate for phase shifts between clock oscillators of different bus controllers and propagation delays between CAN-bus nodes.

$$t_{SJW} = (SJW + 1) * t_q$$
$$SJW = 2SJW.1 + SJW.0$$

BTR1.7	BTR1.6	BTR1.5	BTR1.4	BTR1.3	BTR1.2	BTR1.1	BTR1.0
SAM	TSEG2.2	TSEG2.1	TSEG2.0	TSEG1.3	TSEG1.2	TSEG1.1	TSEG1.0

TSEG1 and TSEG2 determine the number of time quanta per bit period and the location of the sample point, where:

$$t_{TSeg\,1} = (TSeg\,1 + 1) * t_q$$
$$t_{TSeg\,2} = (TSeg\,2 + 1) * t_q$$
$$TSeg\,1 = 8TSeg\,1.3 + 4TSeg\,1.2 + 2TSeg\,1.1 + TSeg\,1.0$$
$$TSeg\,2 = 4TSeg\,2.2 + 2TSeg\,2.1 + TSeg\,2.0$$

SAM – Sampling Mode

- 0 – Single sampling
- 1 – Triple sampling

The CAN in Automation (CiA) International Users and Manufacturers Group has recommended some baud rates to be used in general purpose CAN networks as well as the maximum bus length for a given baud rate. In addition, the bit-timing is recommended, so that nodes from different manufacturers can be connected to one CAN network without calculating the bit-timing parameters.

Table 13.Bus timing

Bit Rate (Bus Length)	Nom. Bit Time [us]	Time Quanta per Bit tq/bit	Length of Time Quantum [us]	Location of Sample Point [tq]	Tseg1 [tq]	Tseg2 [tq]	BTR0 at CLK_I = 16MHz	BTR1 at CLK_I = 16MHz
1Mb/s (25m)	1	8	0.125	6	5	2	00h	14h
0.8Mb/s (50m)	1.25	10	0.125	8	7	2	00h	16h
0.5Mb/s (100m)	2	16	0.125	14	13	2	00h	1Ch
250kb/s (250m)	4	16	0.25	14	13	2	01h	1Ch
125kb/s (500m)	8	16	0.5	14	13	2	03h	1Ch
50kb/s (1km)	20	16	1.25	14	13	2	09h	1Ch
20kb/s (2.5km)	50	16	3.125	14	13	2	18h	1Ch
10kb/s (5km)	100	16	6.25	14	13	2	31h	1Ch

Output Control register (OCR)

Address in CAN memory – 08h

This register is Read-only

Register value after hardware reset – 00000010

The Output Control register allows the set-up of different output driver configurations under software control.

OCR.7	OCR.6	OCR.5	OCR.4	OCR.3	OCR.2	OCR.1	OCR.0
Octrl.5	Octrl.4	Octrl.3	Octrl.2	Octrl.1	Octrl.0	OMODE.1	OMODE.0

OMODE – Output Mode

- 00 – bi-phase output mode
- 01 – test output mode
- 10 – normal output mode
- 11 – clock output mode.

Normal Output Mode

Transmitted bit sequence (TXD) is sent via TX0 and TX1.

Test Output Mode

The level connected to RXD is reflected at TX0 and TX1 with the next positive edge of the system clock (CLK_I).

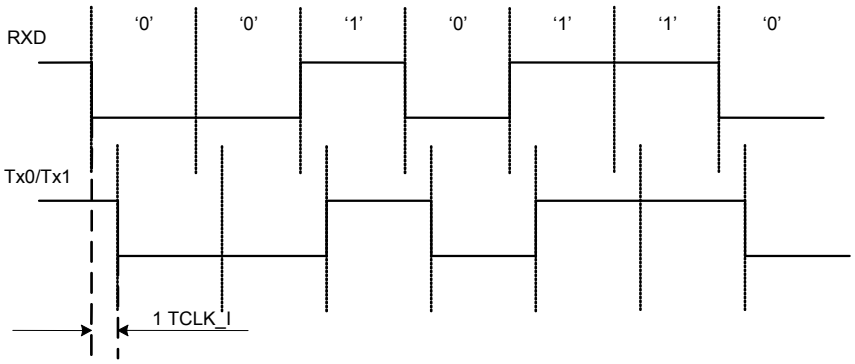


Figure 19. Test output mode

Clock Output Mode

For the TX0 pin, this is the same as in normal output mode. However, the data stream to TX1 is replaced by the transmit clock, whose rising edge marks the beginning of a bit period. The clock pulse width is tq.

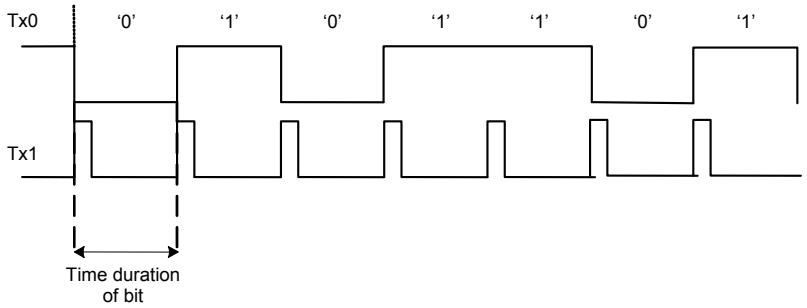


Figure 20. Clock output mode

Bi-phase Output Mode

In contrast to the normal output mode, the bit representation in the bi-phase output mode is time variant and toggled. During recessive bits, all outputs are deactivated (floating) (this is done by setting the **TXEN** output signal).

Dominant bits are sent by toggling the levels on TX0 and TX1.

This mode may be useful if the bus controllers are galvanically decoupled from the bus line by a transformer and the bit stream is not allowed to contain a DC component.

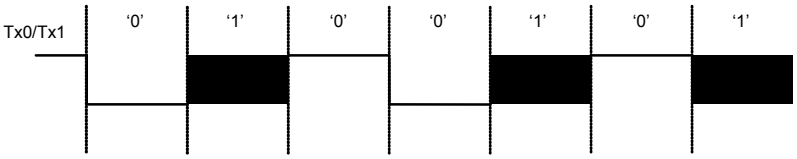


Figure 21. Bi-phase output mode

OCtrl.5 – OCtrl.0 – six bits reflecting the output port OUTCTRL

This port can be used for additional configuration and control of the analog output circuits or the transceiver.

Arbitration Lost Capture register (ALC)

Address in CAN memory – 0Bh

This register is Read-only

Register value after hardware reset – 00000000

ALC.7	ALC.6	ALC.5	ALC.4	ALC.3	ALC.2	ALC.1	ALC.0
-	-	-	ALC				

ALC.7 - ALC.5 – always in low logic state

ALC.4 - ALC.0 – 5-bit value that determines the bit-position at which arbitration lost occurred.

On arbitration lost, the corresponding arbitration lost interrupt is forced, if enabled. At the same time, the current bit position of the Bit Stream is captured into the Arbitration Lost Capture register.

The content within this register is fixed until the user's software has read out its contents once.

The capture mechanism is then activated again. The corresponding interrupt flag is cleared during the read access to the Interrupt register.

A new arbitration lost interrupt is not possible until the Arbitration Lost Capture register has been read once.

Table 14. Interpretation of arbitration lost bit number

Arbitration Lost Capture register (bits ALC.4 – ALC.0)	Standard Frame Format	Extended Frame Format
00000	ID.28	ID.28
00001	ID.27	ID.27
00010	ID.26	ID.26
00011	ID.25	ID.25
00100	ID.24	ID.24
00101	ID.23	ID.23
00110	ID.22	ID.22
00111	ID.21	ID.21
01000	ID.20	ID.20
01001	ID.19	ID.19
01010	ID.18	ID.18
01011	RTR	SRTR
01100	IDE	IDE
01101	-	ID.17
01110	-	ID.16
01111	-	ID.15
10000	-	ID.14
10001	-	ID.13
10010	-	ID.12
10011	-	ID.11
10100	-	ID.10

Arbitration Lost Capture register (bits ALC.4 – ALC.0)	Standard Frame Format	Extended Frame Format
10101	-	ID.9
10110	-	ID.8
10111	-	ID.7
11000	-	ID.6
11001	-	ID.5
11010	-	ID.4
11011	-	ID.3
11100	-	ID.2
11101	-	ID.1
11110	-	ID.0
11111	-	RTR

Error Code Capture register (ECC)

Address in CAN memory – 0Ch

This register is Read-only

Register value after hardware reset – 00000000

ECC.7	ECC.6	ECC.5	ECC.4	ECC.3	ECC.2	ECC.1	ECC.0
ECode.1	ECode.0	Dir	ECC				

ECC.7 - ECC.6 – Error Code

- 00 – bit error
- 01 – form error
- 10 – stuff error
- 11 – crc error or acknowledge error

ECC.5 – Direction when error occurred

- 0 – during reception
- 1 – during transmission

ECC.4 - ECC.0 – 5-bit value that determines the position where the error occurred.

Table 15. Interpretation of error code.

Error Code Capture register (bits ECC.4 – eECC.0)	Function
00000	Idle state
00001	SOF
00011	ID.28 … ID.22
00010	ID.21 … ID.18
00110	SRR
00111	IDE
00101	ID.17 … ID.12

Error Code Capture register (bits ECC.4 – eECC.0)	Function
00100	ID.11 … ID.06
01100	ID.05 … ID.00
01110	RTR
01111	RB.1
01101	RB.0
01001	DLC
01000	CRC sequence
01010	CRC delimiter
01011	ACK slot
10011	ACK delimiter
10010	EOF
10000	Intermission
10101	Active Error Flag
10111	Passive Error Flag
10001	Overload Flag
10110	Tolerate dominant bits
10100	Error Delimiter
11000	Data Field, 8th byte
11001	Data Field, 1st byte
11010	Data Field, 2nd byte
11011	Data Field, 3rd byte
11100	Data Field, 4th byte
11101	Data Field, 5th byte
11110	Data Field, 6th byte
11111	Data Field, 7th byte

The Tolerate dominant bits position results from the superposition of error flags. If a bus error occurs, the corresponding bus error interrupt is forced, if enabled. At the same time, the current position of the Bit Stream is captured into the Error Code Capture register.

The content within this register is fixed until the user's software has read out its content once. The capture mechanism is then activated again. The corresponding interrupt flag is cleared during the read access to the Interrupt register. A new bus error interrupt is not possible until the Error Code Capture register has been read once.

Error Warning Limit register (EWL)

Address in CAN memory – 0Dh

This register is Read-only

Register value after hardware reset – 01100000

EWL.7	EWL.6	EWL.5	EWL.4	EWL.3	EWL.2	EWL.1	EWL.0
EWL							

EWL.7 - EWL.0 – 8-bit value that determines the Error Warning Limit.

When one of the Error Counters reaches the Error Warning Limit and the Error Warning Interrupt Enable bit (IER.2) is set, an Error Warning Interrupt (IR.2) will be generated.

The content of the Error Warning Limit register can only be changed in Reset mode.

A change in the error state and the corresponding Error Warning Interrupt that is generated, will not occur until Reset mode has been cancelled.

Receive Error Counter register (REC)

Address in CAN memory – 0Eh

This register is Read-only

Register value after hardware reset or after entering Bus-Off state – 00000000

REC.7	REC.6	REC.5	REC.4	REC.3	REC.2	REC.1	REC.0
REC							

REC.7 - REC.0 – 8-bit value that determines the output of the Receive Error Counter

The maximum value of the counter is 7Fh.

The content of the Receive Error Counter can only be changed in Reset mode.

A change in the error state and the corresponding Error Warning Interrupt or Error Passive Interrupt forced by the new register content, will not occur until Reset mode has been cancelled.

Transmit Error Counter register (TEC)

Address in CAN memory – 0Fh

This register is Read-only

Register value after hardware reset – 00000000

Register value after entering Bus-Off state – 01111111

TEC.7	TEC.6	TEC.5	TEC.4	TEC.3	TEC.2	TEC.1	TEC.0
TEC							

TEC.7 - TEC.0 – 8-bit value that determines the output of the Transmit Error Counter.

If a Bus-Off event occurs, the Transmit Error Counter is initialized to 127 to count the minimum protocol-defined time (128 occurrences of the bus-free signal). Reading the counter during this time gives information about the status of the Bus-Off recovery.

The content of the Transmit Error Counter can only be changed in Reset mode.

A change in the error state and the corresponding Error Warning Interrupt or Error Passive Interrupt forced by the new register content, will not occur until Reset mode has been cancelled.

Transmit Buffer register (TB0 – TB12)

The Transmit Buffer register allows the user to define a message for transmission. The register is thirteen bytes in length and is accessible at two address spaces:

- Addresses **10h - 1Ch** – accessible during write access in Operating (Normal) mode
- Addresses **60h - 6Ch** – directly readable/writeable memory placed in CAN address space.

In normal, operating mode, this register is Read-only. In Reset mode, both read and write access is possible.

Table 16 shows the content for each of the thirteen bytes in the CAN address space (in this case 10h – 1Ch). One byte (at address 10h) is used to distinguish between Standard Frame Format and Extended Frame Format configurations. In the table, address content has been further divided into two columns – the left hand column representing Standard Frame Format and the right hand column representing Extended Frame Format.

Table 16. Receive/Transmit Buffer register layout

CAN Address	Content	
10h	FRAME INFORMATION	
11h	Identifier, 1st byte	Identifier, 1st byte
12h	Identifier, 2nd byte	Identifier, 2nd byte
13h	1st Data byte	Identifier 3rd byte
14h	2nd Data byte	Identifier 4th byte
15h	3rd Data byte	1st Data byte
16h	4th Data byte	2nd Data byte
17h	5th Data byte	3rd Data byte
18h	6th Data byte	4th Data byte
19h	7th Data byte	5th Data byte
1Ah	8th Data byte	6th Data byte
1Bh	not used	7th Data byte
1Ch	not used	8th Data byte

Frame Information

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
IDE	RTR	RB.1/'0'	RB.0	DLC.3	DLC.2	DLC.1	DLC.0

IDE – Identifier Extension bit **-** defines message format.

- 0 – Standard Frame Format
- 1 – Extended Frame Format.

RTR – Remote Transmission Request bit

BIT 5:

- for Standard Frame Format – '0'
- for Extended Frame Format – RB.1

DLC – Data Length Code

The number of bytes in the data field of the message, calculated as:

$$DataLength\ \ Code\ = 8DLC.3 + 4DLC.2 + 2DLC.1 + DLC.0$$

The data length code is not considered when the RTR bit is set (logic 1)

For reasons of compatibility, the DLC code should be less than or equal to 8. It should be defined in software. Selection of a value greater than 8 will result in frame errors (the CAN Controller will try to send more than 8 data bytes, which is not compatible with the CAN2.0 specification).

Identifier, 1st byte

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

Identifier 2nd byte for Standard Frame Format

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.20	ID.19	ID.18	RTR[1]	-	-	-	-

Identifier, 2nd byte for Extended Frame Format

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13

Identifier, 3rd byte

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5

Identifier, 4th byte

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.4	ID.3	ID.2	ID.1	ID.0	RTR[1]	-	-

The identifier acts as the message's name, which is subsequently used in a receiver for acceptance filtering. It also determines the bus access priority during the arbitration process. The lower the binary value of the identifier, the higher the message priority. This is due to the larger number of leading dominant bits during arbitration.

Data byte

BIT 7	BIT 6	BIT 5	BIT4	BIT 3	BIT 2	BIT 1	BIT 0
D.7	D.6	D.5	D.4	D.3	D.2	D.1	D.0

Receive Buffer register (RB0 – RB12)

The Receive Buffer register is thirteen bytes in length and is accessible at two address spaces:

- Addresses **10h - 1Ch** – Receive Buffer register, accessible during read access in Operating (normal) mode
- Addresses **20h - 5Fh** – direct read/write of entire Receive FIFO (64-bytes)**.**

In normal, operating mode, this register is Read-only. In Reset mode, both read and write access is possible.

[1] With respect to the Transmit Buffer register, this bit is not used. The more significant 6th bit of the **Frame Information** byte is used. With respect to the Receive Buffer register, this bit is used and is a direct copy of the 6th bit in the **Frame Information** byte.

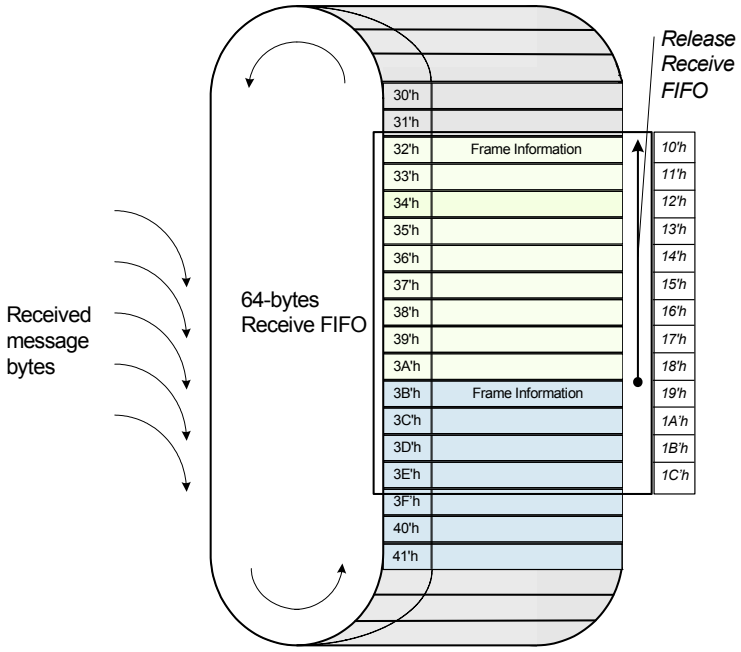


Figure 22. Receive FIFO

The Receive Buffer register is located as a 13-byte overlapping window within the Receive FIFO. The beginning of this window is pointed to by the Receive Buffer Start Address register (RBSA**)**.

It is possible for there to be more than one message accessible within the Receive Buffer register, due to the fact that the register's length is 13-bytes (maximum message length) and individual messages may be shorter in length. The actual length of a message depends on the frame format – standard or extended – and the number of data bytes. Table 11 in the previous section illustrates the message format for both standard and extended frame formats.

With a 64-byte Receive FIFO, it is possible to store between 7 and 21 messages (depending on the frame format and size of the data therein).

The global layout of the Receive Buffer register is similar to that of the Transmit Buffer register.

The received Data Length Code (DLC**),** located in the Frame Information byte, may be greater than 8 (depending on the sender). For compatibility with the CAN2.0 specification however, the maximum number of received data bytes is limited to 8. This should be taken into account when reading a message from the Receive Buffer register.

If there is not enough space for a new message byte within the Receive FIFO, the CAN Controller generates a data overrun condition. A message that has been partially written into the Receive FIFO when the data overrun situation occurs, is deleted. This situation is indicated to the CPU via the Status register (SR.1 – Data Overrun Status) and the corresponding Data Overrun Interrupt (IR.3) that is generated, if enabled.

Acceptance Filter

The Acceptance Filter allows the user to configure the CAN Controller so that only the desired messages are written into the Receive FIFO. This is achieved by comparing the identifier bits of received messages with predefined ones stored within the Acceptance Filter registers. If there is an exact match, the message is written (accepted) into the Receive FIFO.

The Acceptance Filter is defined by the Acceptance Code registers (ACR0 – ACR3) and the Acceptance Mask registers (AMR0 – AMR3).

The identifier bit patterns of messages to be received are defined within the Acceptance Code registers.

The corresponding Acceptance Mask registers allow the definition of certain bit positions in the identifiers to be ‘don’t care’ ('1' means don't care bit).

Two different filter modes are selectable. Mode selection is determined by bit 3 of the Mode register (MOD.3 - Acceptance Filter Mode):

- Single filter mode (MOD.3 is logic 1)
- Dual filter mode (MOD.3 is logic 0).

The bit correspondences between the identifier bytes of the Acceptance Filter and those of the message depend also on the currently received frame format (standard or extended).

Acceptance Code registers (ACR0 - ACR3)

Addresses in CAN memory – 10h - 13h

These registers are accessible for read and write only in Reset mode.

Acceptance Mask register (AMR0 - AMR3)

Addresses in CAN memory – 14h - 17h

These registers are accessible for read and write only in Reset mode.

Register values after hardware reset – 11111111

Single filter configuration, receiving standard frame messages

The complete identifier (two bytes) including the RTR and RB.0 bits and two data bytes are used for acceptance filtering. Messages may also be accepted if there are no data bytes due to Remote Frame, or if there is none or only one data byte because of the corresponding Data Length Code. In this case, filtering of corresponding data bytes is not performed.

For successful reception of a message, all single bit comparisons have to signal acceptance.

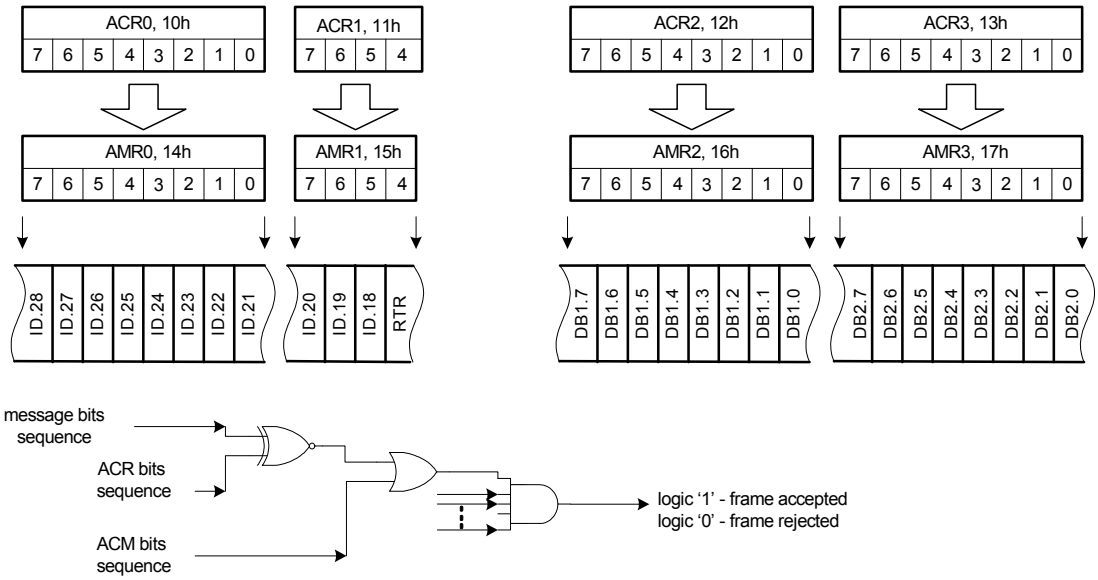


Figure 23. Single filter configuration, receiving standard frame messages

Single filter configuration, receiving extended frame messages

The complete identifier (four bytes), including the RTR, RB.1 and RB.0 bits, is used for acceptance filtering. For successful reception of a message, all single bit comparisons have to signal acceptance.

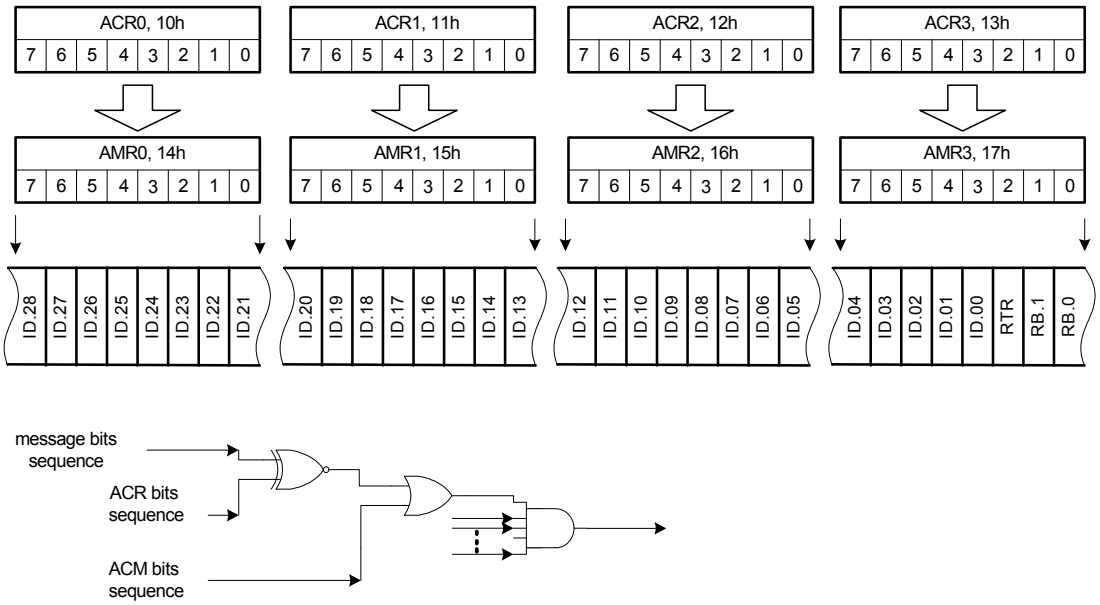
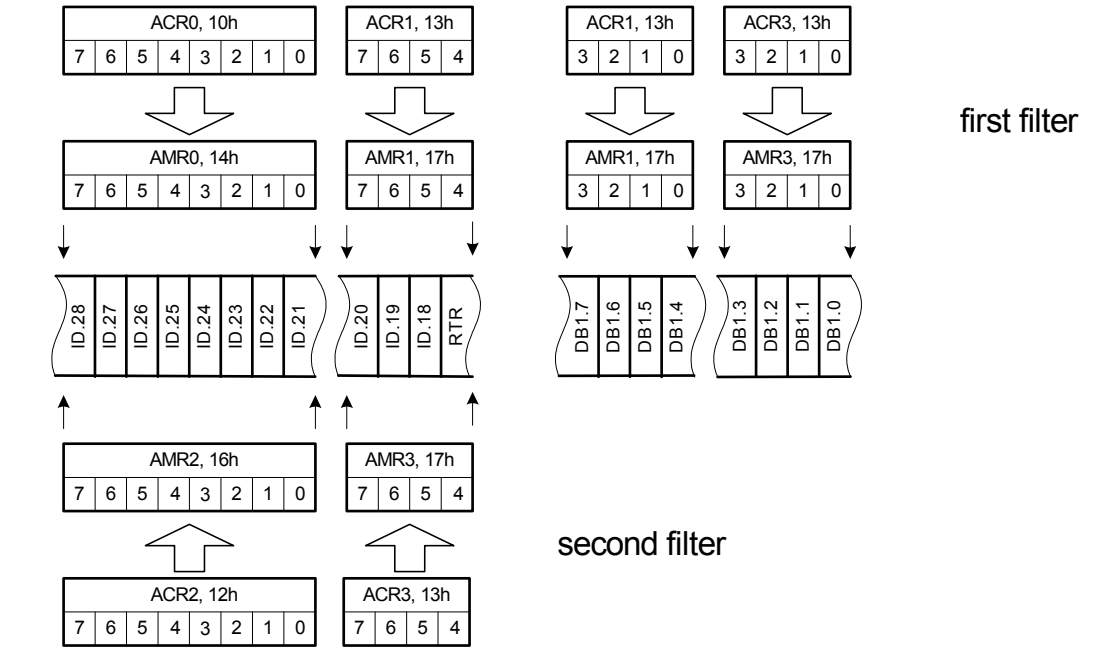


Figure 24. Single filter configuration, receiving extended frame messages

Dual filter configuration, receiving standard frame messages

There are two short filters defined. A received message is compared with both filters to decide whether this message should be copied into the Receive Buffer register or not. If at least one of the filters signals an acceptance, the received message becomes valid.

The first filter compares the complete standard identifier (two bytes) including the RTR bit and the first data byte of the message. The second filter just compares the complete standard identifier including the RTR bit acceptance. In the case of Remote Frame or if no data byte exists, filtering of the corresponding data byte is not performed.



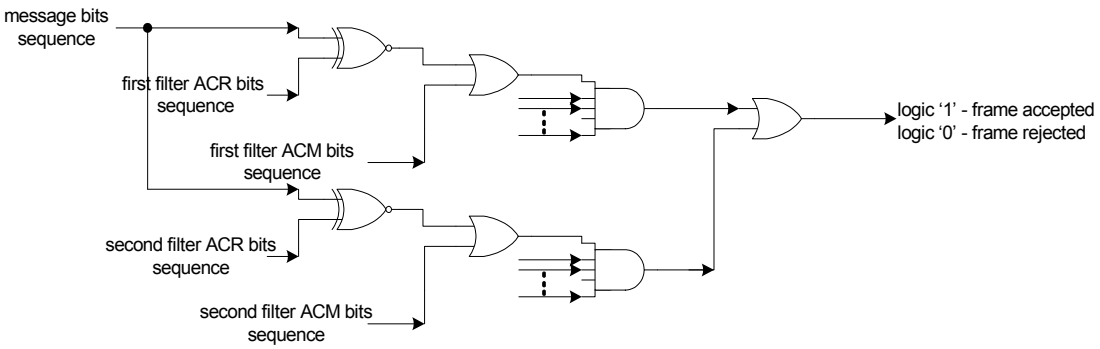


Figure 25. Dual filter configuration receiving standard frame messages

Dual filter configuration, receiving extended frame messages

There are two short filters defined. A received message is compared with both filters to decide whether this message should be copied into the Receive Buffer register or not. If at least one of the filters signals an acceptance, the received message becomes valid.

The two defined filters are identical. Both filters compare the first two bytes of the extended identifier.

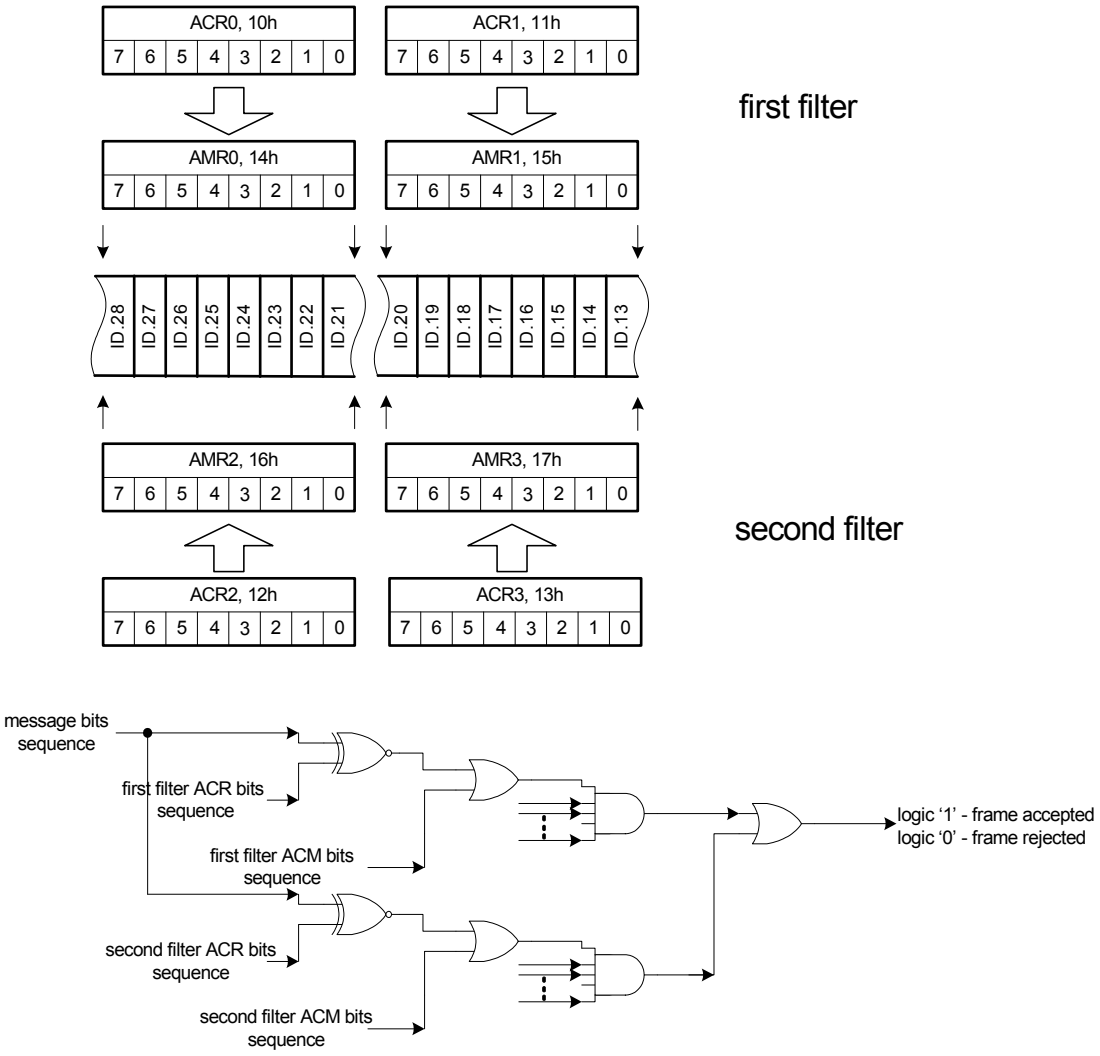


Figure 26. Dual filter configuration receiving extended frame messages

Message Counter (MC)

The Message Counter reflects the current number of messages available within the Receive FIFO. Its value is incremented each time a new message is successfully received and accepted.

Address in CAN memory – 1Dh

This register is Read-only

Register value in Reset mode – 00000000

MC.7	MC.6	MC.5	MC.4	MC.3	MC.2	MC.1	MC.0
-	-	-	MC				

MC.7 - MC.5 – always in low logic state

MC.4 - MC.0 – 5-bit value defining the number of messages stored in the Receive FIFO.

Receive Buffer Start Address register (RBSA)

Address in CAN memory – 1Eh

This register is Read-only

Register value after hardware reset – 00000000

RBSA.7	RBSA.6	RBSA.5	RBSA.4	RBSA.3	RBSA.2	RBSA.1	RBSA.0
-	-	RBSA					

RBSA.7 - RBSA.6 – always in low logic state

RBSA.5 - RBSA.0 – 6-bit value reflecting the current internal Receive FIFO address where the first byte of a received message (which is mapped to the Receive Buffer window) is stored.

The value 00h reflects address 20h of CAN address space (beginning of the Receive FIFO).

As the Receive FIFO is 64-bytes in length, the value of the RBSA register can be in the range 00h - 3Fh.

If a message exceeds address 80h it continues at 00h.

After the Release Receive Buffer command (CMR.2) is issued, the RBSA is updated to point at the start of the next received message in the Receive FIFO.

Clock Divider Register CDR

Address in CAN memory – 1Fh

This register is Read-only

Register value after hardware reset – 11000000

CDR.7	CDR.6	CDR.5	CDR.4	CDR.3	CDR.2	CDR.1	CDR.0
1	1	RxIntEn	Input Mode	Clock off	CD2	CD1	CD0

CDR.7 - CDR.6 – always in high logic state.

RxIntEn – activates dedicated receive interrupt pulse on TX1.

When a received message has been received and written into Receive FIFO a receive interrupt pulse with the length of one bit time is generated at the TX1 pin (during the last bit of end of frame). The output should operate in normal output mode.

InputMode – when 1 activates CAN input test mode.

In this mode, Receiver input is connected internally with Transmitter output.

This mode allows you to perform CAN Controller tests without connecting it to the CAN-bus.

Clock off – setting this bit deactivates the CLK_OUT pin.

If this bit is set, the CLK_OUT output pin is kept in the low logic state during Sleep mode, otherwise it is in the high logic state.

CD2 - CD0 – defines clock divider for generation of CLK_OUT signal from the system CLK_I signal.

$$t_{ClkOut} = 2 * t_{CLK\_I}(4 * CDR.2 + 2 * CDR.1 + CDR.0 + 1)$$

Host to Controller Communications

Communications between a host processor and the CAN Controller is carried out over the standard Wishbone bus. The host processor can write to/read from either of the CAN_W's internal Wishbone registers. Selection of a register – either WAREG or WDREG – is achieved by supplying the unique, binary ID address code of the register. This code is sent to the component and appears at the ADR_I input. Table 17 shows the unique address IDs associated with the registers for the CAN_W.

Table 17. CAN_W Internal Wishbone register unique address IDs.

Register	Unique Register Address ID
WAREG	0
WDREG	1

Writing to the Internal Wishbone Registers

Data is written from the host processor to an internal Wishbone register, in accordance with the standard Wishbone data transfer handshaking protocol. This data transfer cycle can be summarized as follows (the write operation occurs on the rising edge of the CLK_I input):

- The host presents the unique address ID for the register to be written on its ADR_O output and a valid byte of data on its DAT_O output. It then asserts its WE_O signal, to specify a Write cycle
- The CAN_W receives the address ID on its ADR_I input and prepares to receive data into the selected register
- The host asserts its STB_O and CYC_O outputs, indicating that the transfer is to begin. The CAN_W, which monitors its STB_I and CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by latching the byte of data appearing at its DAT_I input into the target register and asserting its ACK_O signal – to indicate to the host that the data has been received
- The host, which monitors its ACK_I input on each rising edge of the CLK_I signal, responds by negating the STB_O and CYC_O signals. At the same time, the CAN_W negates the ACK_O signal and the data transfer cycle is naturally terminated.

Reading from the Internal Wishbone Registers

Data is read from an internal Wishbone register in accordance with the standard Wishbone data transfer handshaking protocol. This data transfer cycle can be summarized as follows (the read operation occurs on the rising edge of the CLK_I input):

- The host presents the unique address ID for the register to be read on its ADR_O output. It then negates its WE_O signal to specify a Read cycle
- The CAN_W receives the address ID on its ADR_I input and prepares to transmit data from the selected register
- The host asserts its STB_O and CYC_O outputs, indicating that the transfer is to begin. The CAN_W, which monitors its STB_I and CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by presenting a valid byte of data at its DAT_O output and asserting its ACK_O signal – to indicate to the host that valid data is present
- The host, which monitors its ACK_I input on each rising edge of the CLK_I signal, responds by latching the byte of data appearing at its DAT_I input and negating the STB_O and CYC_O signals. At the same time, the CAN_W negates the ACK_O signal and the data transfer cycle is naturally terminated.

Revision History

Date	Version No.	Revision
12-Feb-2009	1.0	Initial release
30-Aug-2011	-	Updated template.