Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Not Recommended for New Designs

Use MCP2515 or MCP25625

MCP2502X/5X

CAN I/O Expander Family

Features

- Implements CAN V2.0B
  - Programmable bit rate up to 1 Mb/s
  - One programmable mask
  - Two programmable filters
  - Three auto-transmit buffers
  - Two message reception buffers
  - Does not require synchronization or configuration messages
- Hardware Features
  - Non-volatile memory for user configuration
  - User configuration automatically loaded on Power-up
  - Eight general-purpose I/O lines individually selectable as inputs or outputs
  - Individually selectable transmit-on-pin-change for each input
  - Four 10-bit, analog input channels with programmable conversion clock and VREF sources (MCP2505X devices only)
  - Message scheduling capability
  - Two 10-bit PWM outputs with independently programmable frequencies
  - Device configuration can be modified via CAN bus messages
  - In-Circuit Serial Programming™ (ICSP™) of default Configuration memory
  - Optional 1-wire CAN bus operation
- Low-power CMOS technology
  - Operates from 2.7V to 5.5V
  - 10 mA active current, typical
  - 30 µA standby current (CAN Sleep mode)
- 14-pin PDIP (300 mil) and SOIC (150 mil) packages
- Available temperature ranges:
  - Industrial (I): -40°C to +85°C
  - Extended (E): -40°C to +125°C

Description

The MCP2502X/5X devices operate as I/O expanders for a Controller Area Network (CAN) system, supporting CAN v2.0B active, with bus rates up to 1 Mb/s. The MCP2502X/5X allows a simple CAN node to be implemented without the need for a microcontroller.

The devices are identical, with the following exceptions:

Device	A/D	One Wire Digital CANbus
MCP25020	No	No
MCP25025	No	Yes
MCP25050	Yes	No
MCP25055	Yes	Yes

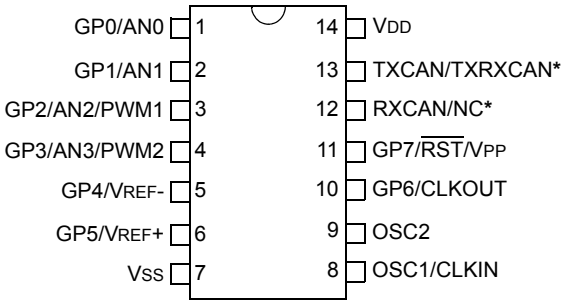
The MCP2502X/5X devices feature a number of peripherals, including digital I/Os, four-channel 10-bit A/D (MCP2505X), and PWM outputs with automatic message transmission on change-of-input state. This includes an analog input exceeding a preset threshold.

One mask and two acceptance filters are provided to give maximum flexibility during system design with respect to identifiers that the device will respond to. The device can also be configured to automatically transmit a unique message whenever any of several error conditions occur.

The device is pre-programmed in non-volatile memory so that the part defaults to a specific configuration at Power-up.

Package Types

PDIP/SOIC



* One-wire option available on MCP250X5 devices.

Definition of Terms

The following terms are used throughout this document:

I/O Expander – refers to the integrated circuit (IC) device being described (MCP2502X/5X).

Input Message – term given to messages that are received by the MCP2502X/5X and cause the internal registers to be modified. Once the register modification has been performed, the MCP2502X/5X transmits a Command Acknowledge message to indicate that the command was received and processed.

Command Acknowledge Message – term given to the message that is automatically transmitted by the MCP2502X/5X after receiving and processing an input message.

Information Request Message – term given to the Remote Request messages that are received by the MCP2502X/5X that subsequently generate an output message (data frame) in response.

Output Message – term given to the message that the MCP2502X/5X sends in response to an Information Request message.

On Bus Message – term given to the message that the MCP2502X/5X transmits after completing the Power-On and/or Self-Configuration sequences at timed intervals, if enabled.

Self-Configuration – term used to describe the process of transferring the contents of the EPROM memory array to the SRAM memory array.

On Bus – term used to describe the condition when the MCP2502X/5X is fully-configured and ready to transmit or receive on the bus. This is the only state in which the MCP2502X/5X can transmit on the bus.

Edge Detection – refers to the MCP2502X/5X's ability to automatically transmit a message based on the occurrence of a predefined edge on any digital input.

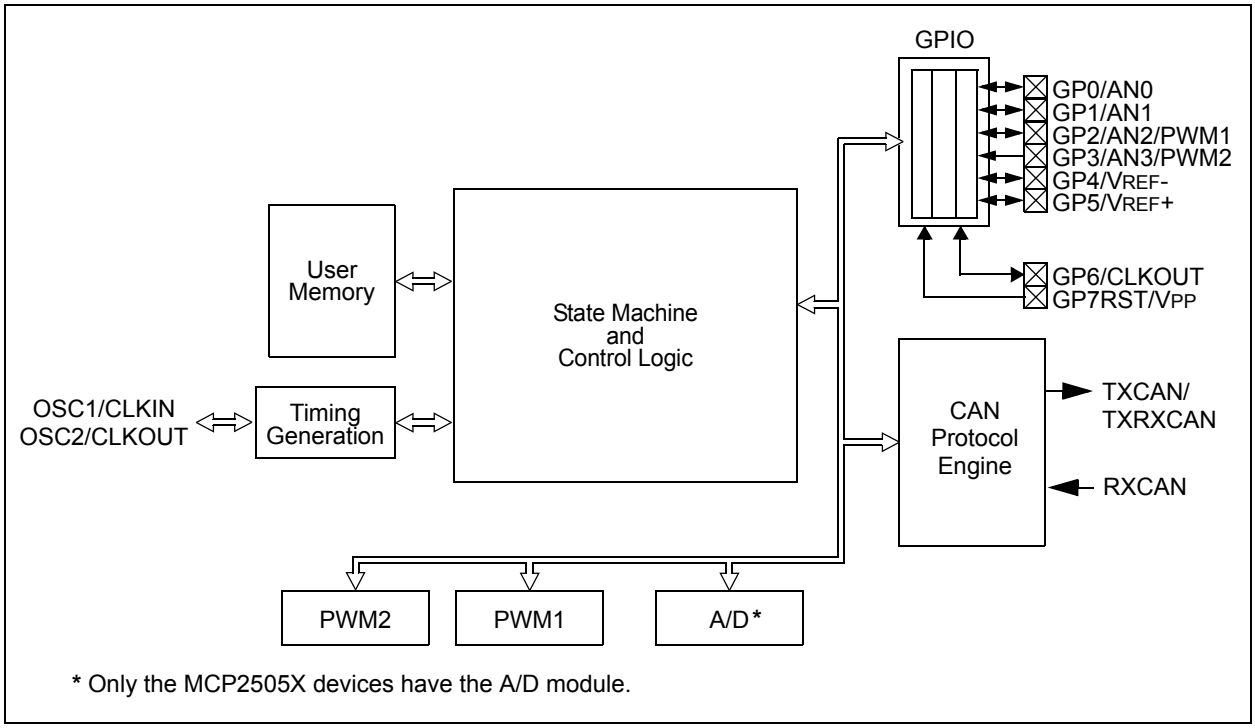
Threshold Detection – refers to the MCP2502X/5X's ability to automatically transmit a message when a predefined analog threshold is reached.

1.0 DEVICE OVERVIEW

This document contains device-specific information on the MCP2502X/5X family of CAN I/O expanders. The CAN protocol is not discussed in depth in this document. Additional information on the CAN protocol can be found in the CAN specification, as defined by Robert Bosch GmbH.

Figure 1-1 is the block diagram of the MCP2502X/5X and Table 1-1 is the pinout description.

FIGURE 1-1: MCP2502X/5X BLOCK DIAGRAM



* Only the MCP2505X devices have the A/D module.

TABLE 1-1: PINOUT DESCRIPTION

Pin Name	Pin Number	Standard Function	Alternate Function	Programming Mode Function
GP0/AN0 *	1	Bidirectional I/O pin, TTL input buffer	Analog input channel	None
GP1/AN1 *	2	Bidirectional I/O pin, TTL input buffer	Analog input channel	None
GP2/AN2/PWM2 *	3	Bidirectional I/O pin, TTL input buffer	Analog input/PWM output	None
GP3/AN3/PWM3 *	4	Bidirectional I/O pin, TTL input buffer	Analog input/PWM output	None
GP4/VREF-	5	Bidirectional I/O pin, TTL input buffer	External VREF-	Data
GP5/VREF+	6	Bidirectional I/O pin, TTL input buffer	External VREF+ input	Clock
VSS	7	Ground	None	Ground
OSC1/CLKIN	8	External oscillator input	External clock input	None
OSC2	9	External oscillator output	None	None
GP6/CLKOUT	10	Bidirectional I/O pin, TTL input buffer	CLKOUT output	None
GP7/$\overline{\text{RST}}$/VPP	11	Input pin, TTL input buffer	External Reset input	VPP
RXCAN	12	CAN data receive input	Not connected for 1-wire operation	None
TXCAN/TXRXCAN	13	CAN data transmit output	CAN TX and RX for 1-wire operation (MCP250X5)	None
VDD	14	Power	None	Power

* Only the MCP2505X devices have the A/D module.

NOTES:

2.0 CAN MODULE

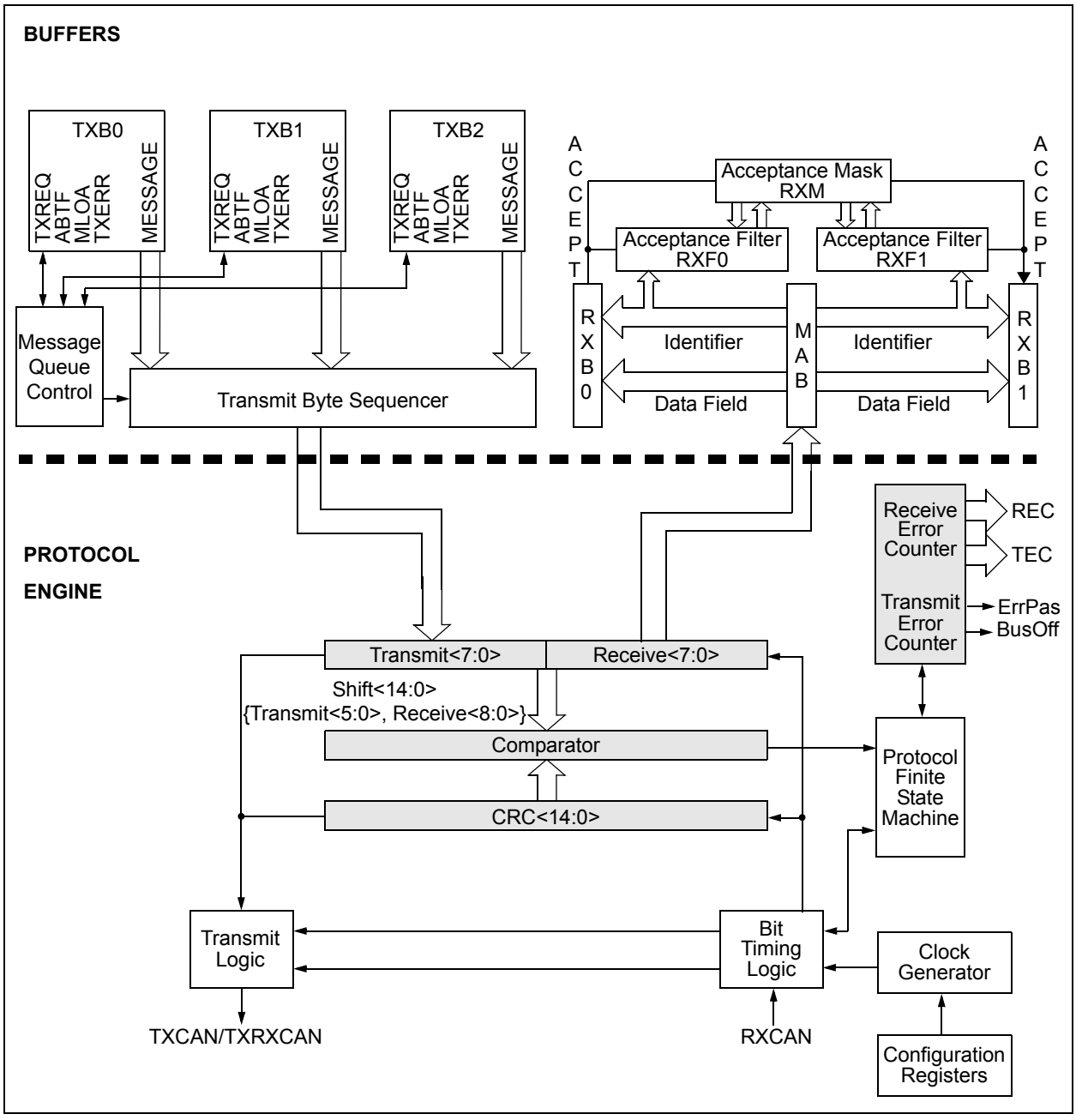
The CAN module is a protocol controller that converts between raw digital data and CAN message packets. The main functional block of the CAN module is shown in Figure 2-1 and consists of:

- CAN protocol engine
- Buffers, masks and filters

The module features include:

- Implementation of the CAN protocol
- Double-buffered receiver with two separate receive buffers
- One full-acceptance mask (standard and extended)
- Two full-acceptance filters (standard and extended)
- One filter for each receive buffer
- Three prioritized transmit buffers for transmitting predefined message types
- Automatic wake-up on bus traffic function
- Error management logic for transmit and receive error states
- Low-power SLEEP mode

FIGURE 2-1: CAN MODULE



2.1 CAN Protocol Finite State Machine

The heart of the engine is the Finite State Machine (FSM). This state machine sequences through messages on a bit-by-bit basis, changing states as the fields of the various frame types are transmitted or received. The FSM is a sequencer controlling the sequential data stream between the TX/RX Shift register, the CRC register and the bus line. The FSM also controls the Error Management Logic (EML) and the parallel data stream between the TX/RX Shift registers and the buffers. The FSM ensures that the processes of reception, arbitration, transmission and error signaling are performed according to the CAN protocol. The automatic retransmission of messages on the bus line is also handled.

2.2 Cyclic Redundancy Check (CRC)

The CRC register generates the CRC code that is transmitted after either the Control field (for messages with 0 data bytes) or the Data field, and is used to check the CRC field of incoming messages.

2.3 Error Management Logic

The error management logic is responsible for the fault confinement of the CAN device. Its two counters, the Receive Error Counter (REC) and the Transmit Error Counter (TEC), are incremented and decremented by commands from the bit stream processor. According to the values of the error counters, the MCP2502X/5X is set into one the following states: Error-Active, Error-Passive, or Bus-Off.

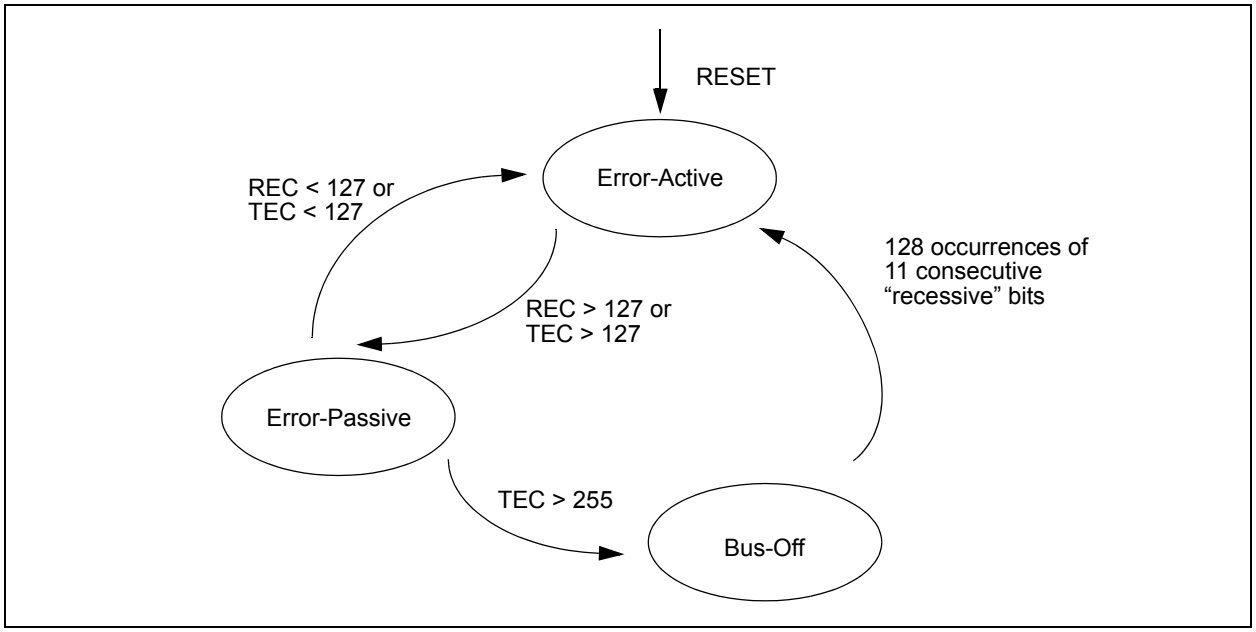
Error-Active: both error counters are below the error-passive limit of 128.

Error-Passive: at least one of the error counters (TEC or REC) equals or exceeds 128.

Bus-Off: the transmit error counter (TEC) equals or exceeds the bus-off limit of 256. The device remains in this state until the bus-off recovery sequence is received. The bus-off recovery sequence consists of 128 occurrences of 11 consecutive recessive bits.

Note:	The MCP2502X/5X, after going bus-off, will recover to error-active automatically if the bus remains idle for 128 x 11 bits. OPTREG2.ERRE must be set to force the MCP2502X/5X to enter Listen-Only mode, instead of Normal mode, during bus recovery. The current error mode (except for bus-off) of the MCP2502X/5X can be determined by reading the EFLG register via the Read CAN error message.

FIGURE 2-2: ERROR MODES STATE DIAGRAM



REGISTER 2-1: TEC - TRANSMITTER ERROR COUNTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0
bit 7							bit 0

bit 7-0 **TEC7:TEC0:** Transmit Error Counter bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

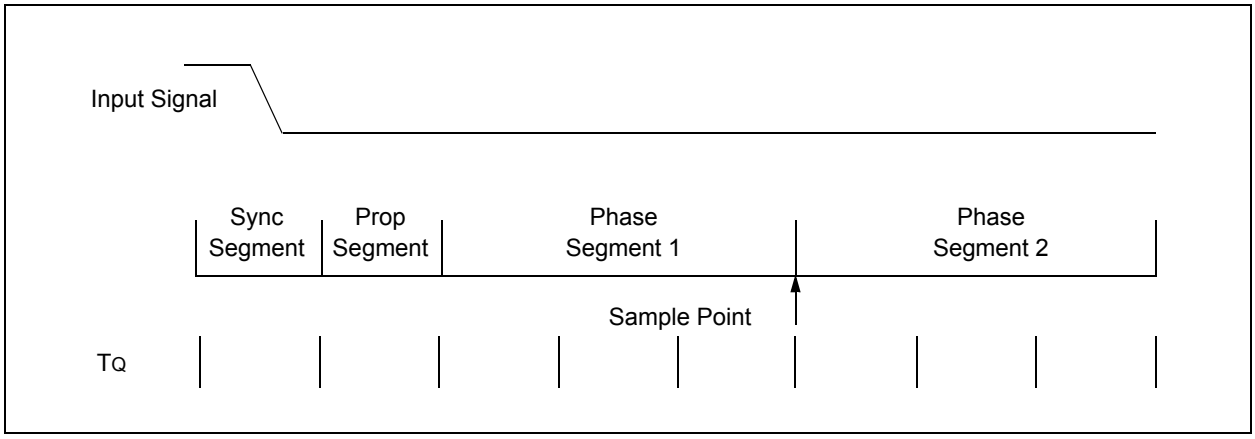
REGISTER 2-2: REC - RECEIVER ERROR COUNTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
bit 7							bit 0

bit 7-0 **REC7:REC0:** Receive Error Counter bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

FIGURE 2-3: BIT TIME PARTITIONING



2.4 Bit Timing Logic

The Bit Timing Logic (BTL) monitors the bus line input and handles the bus-related bit timing, based on the CAN protocol. The BTL synchronizes on a recessive-to-dominant bus transition at Start-of-Frame (hard synchronization) and on any further recessive-to-dominant bus line transition if the CAN controller itself does not transmit a dominant bit (resynchronization). The BTL also provides programmable time segments to compensate for the propagation delay time, phase shifts, and to define the position of the sample point within the bit time. These programmable segments are made up of integer units called Time Quanta (TQ).

The nominal bit time is calculated by programming the TQ length and the number of TQ in each time segment, as discussed below.

2.4.1 TIME QUANTUM (TQ)

TQ is a fixed unit of time derived from the oscillator period. There is a programmable baud rate prescaler (BRP) (with integral values ranging from 1 to 64), as well as a fixed division by two for clock generation.

The base TQ is defined as twice the oscillator period. Adding the BRP into the equation yields:

$$T_Q = 2 * T_{OSC} * (BRP + 1)$$

where BRP = binary value represented by CNF1.BRP<5:0>

By definition, the nominal bit time is programmable from a minimum of 8 TQ to 25 TQ. Also, the minimum nominal bit time is 1 µs, which corresponds to 1 Mbps.

2.4.2 TIME SEGMENTS

Time segments make up the nominal bit time. The nominal bit time can be thought of as being divided into separate non-overlapping time segments. These segments are shown in Figure 2-3.

- Synchronization Segment (SyncSeg)
- Propagation Segment (PropSeg)
- Phase Buffer Segment 1 (PS1)
- Phase Buffer Segment 2 (PS2)

$$Nominal\ Bit\ Time = T_Q * (Sync\_Seg + PropSeg + Phase\_Seg1 + Phase\_Seg2)$$

Rules for Programming the Segments

There are a few rules to follow when programming the time segments:

- PropSeg + PS1 ≥ PS2
- PS2 > Sync Jump Width
- PS2 ≥ Information Processing Time

2.4.2.1 Synchronization Segment

The Synchronization Segment (SyncSeg) of the bit time is used to synchronize the various CAN nodes on the bus. The edge of the input signal is expected to occur during the SyncSeg. The duration is fixed at 1 TQ.

2.4.2.2 Propagation Segment

This part of the bit time is used to compensate for physical delay times within the network. These delay times consist of the signal propagation time on the bus line and the internal delay time of the nodes. The delay is calculated as being the round-trip time from transmitter to receiver (twice the signal's propagation time on the bus line), the input comparator delay and the output driver delay. The length of the Propagation Segment can be programmed from 1 TQ to 8 TQ by setting the PRSEG2:PRSEG0 bits of the CNF2 register.

2.4.2.3 Phase Buffer Segments

The Phase Buffer Segments are used to optimally locate the sampling point of the received bit within the nominal bit time. The sampling point occurs between PS1 and PS2. These segments can be automatically lengthened or shortened by the resynchronization process. Thus, the variation of the values of the phase buffer segments represent the DPLL functionality.

PS1: the end of PS1 determines the sampling point within a bit time. PS1 is programmable from 1 TQ to 8 TQ in duration.

PS2: PS2 provides delay before the next transmitted data transition and is also programmable from 1 TQ to 8 TQ in duration. However, due to Information Processing Time (IPT) requirements, the actual minimum length of PS2 is 2 TQ. It can also be defined as equal to the greater of PS1 or the IPT.

2.4.3 SAMPLE POINT

The sample point is the point of time at which the bus level is read and the value of the received bit is determined. The sampling point occurs at the end of PS1. If desired, it is possible to specify multiple sampling of the bus line at the sample point. The value of the received bit is determined to be the value of the majority decision of three values. The three samples are taken at the sample point, and twice before, with a time of TQ/2 between each sample.

2.4.4 INFORMATION PROCESSING TIME

IPT is the time segment (starting at the sample point) that is reserved for calculation of the subsequent bit level. The CAN specification defines this time to be less than or equal to 2 TQ. The MCP2502X/5X defines this time to be 2 TQ. Thus, PS2 must be at least 2 TQ long.

2.4.5 SYNCHRONIZATION JUMP WIDTH (SJW)

To compensate for phase shifts and oscillator tolerances between the nodes in the system, each CAN controller must be able to synchronize to the relevant signal edge of the incoming signal. When a recessive-to-dominant edge in the transmitted data is detected, the logic will compare the location of the edge to the expected time (SyncSeg). The circuit will then adjust the values of PS1 and PS2, as necessary, using the programmed SJW. This adjustment is made for resynchronization during a message and not hard synchronization, which occurs only at the message Start-of-Frame (SOF).

As a result of resynchronization, PS1 may be lengthened or PS2 may be shortened. The amount of lengthening or shortening of the phase buffer segments has an upper-boundary given by the SJW. The SJW is programmable between 1 TQ and 4 TQ. The value of the SJW will be added to PS1 (or subtracted from PS2) depending on the phase error (e) of the edge in relation to the receiver's SyncSeg. The phase error is defined as follows:

- e = 0 if the edge lies within SYNCESEG
  No resynchronization is required.
- e > 0 if the edge lies before the sample point
  PS1 will be lengthened by the amount of the SJW.
- e < 0 if the edge lies after the sample point of the previous bit and before the SyncSeg of the current bit
  PS2 will be shortened by the amount of the SJW.

2.4.6 CONFIGURATION REGISTERS

There are three registers (in the Configuration register module) associated with the CAN bit timing logic that controls the bit timing for the CAN bus interface.

2.4.6.1 CNF1

The BRP<5:0> bits control the baud rate prescaler. These bits set the length of TQ relative to the OSC1 input frequency, with the minimum length of TQ being 2 TOSC in length (when BRP<5:0> are set to 000000). The SJW<1:0> bits select the synchronization jump width in terms of number of TQ's.

2.4.6.2 CNF2

The PRSEG<2:0> bits set the length (in TQ's) of the propagation segment. The PS1<2:0> bits set the length (in TQ's) of phase segment 1. The SAM bit controls how many times the RXCAN pin is sampled. Setting this bit to a '1' causes the bus to be sampled three times. Twice at TQ/2 before the sample point and once at the normal sample point (which is at the end of PS1). The value of the bus is determined to be the value read during at least two of the samples. If the SAM bit is set to a '0', the RXCAN pin is sampled only once at the sample point. The BTLMODE bit controls how the length of PS2 is determined. If this bit is set to a '1', the length of PS2 is determined by the PS2<2:0> bits of CNF3. If the BTLMODE bit is set to a '0', then the length of PS2 is the greater of PS1 and the information processing time (which is fixed at 2 TQ for the MCP2502X/5X).

2.4.6.3 CNF3

The PS2<2:0> bits set the length, in TQ's, of PS2, if the CNF2.BTLMODE bit is set to a '1'. If the BTLMODE bit is set to a '0', the PS2<2:0> bits have no effect.

Additionally, the wake-up filter (CNF3.WAKFIL) is implemented in the CNF3 register. This filter is a low-pass filter that can be used to prevent the MCP2502X/5X from waking up due to short glitches on the CAN bus.

REGISTER 2-3: CNF1 - CAN CONFIGURATION REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
bit 7							bit 0

bit 7-6 **SJW1:SJW0:** Synchronized Jump Width bits
11 = Length = 4 x TQ
10 = Length = 3 x TQ
01 = Length = 2 x TQ
00 = Length = 1 x TQ

bit 5-0 **BRP5:BRP0:** Baud Rate Prescaler bits
111111 = TQ = 64 x 1/Fosc
-
-
000000 = TQ = 64 x 1/Fosc

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-4: CNF2 - CAN CONFIGURATION REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BTLMODE	SAM	PHSEG12	PHSEG11	PHSEG10	PRSEG2	PRSEG1	PRSEG0
bit 7							bit 0

bit 7 **BTL MODE:** Length determination of PHSEG2 bit
1 = Length of Phase_Seg2 determined by bits 2:0 of CNF3
0 = Length of Phase_Seg2 is the greater of Phase_Seg1 or IPT(2TQ)

bit 6 **SAM:** Sample of the CAN bus line bit
1 = Bus line is sampled three times at the sample point
0 = Bus line is sampled once at the sample point

bit 5-3 **PHSEG12:PHSEG10:** Phase Buffer Segment1 bits
111 = Length = 8 x TQ
-
-
-
000 = Length = 1 x TQ

bit 2-0 **PRSEG2:PRSEG0:** Propagation Time Segment bits
111 = Length = 8 x TQ
-
-
-
000 = Length = 1 x TQ

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-5: CNF3 - CAN CONFIGURATION REGISTER 3

U-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	WAKFIL	—	—	—	PHSEG22	PHSEG21	PHSEG20
bit 7							bit 0

bit 7 **Unimplemented:** (Reads as 0)

bit 6 **WAKFIL:** Wake-up filter bit
1 = Wake-up filter enabled
0 = Wake-up filter disabled

bit 5-3 **Unimplemented:** (Reads as 0)

bit 2-0 **PHSEG22:PHSEG20:** Phase Buffer Segment2 bits
111 = Length = 8 x TQ
-
-
-
001 = Length = 2 x TQ
000 = Invalid

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.5 Buffers, Masks, and Filters

This part of the CAN module supports the transmitting, receiving and acceptance of CAN messages.

Three transmit buffers are used for the three transmit message IDs, as discussed later in this section. Two receive buffers store the CAN message's arbitration field, control field and the data field.

One mask defines which bits are to be applied to either filter. The mask can be regarded as defining "don't care" bits for the filter.

Each of the two filters define a bit pattern that will be compared to all incoming messages. All filter bits that have not been defined as "don't care" by the mask are applied to the message.

2.5.1 TRANSMIT MESSAGE IDs

The MCP2502X/5X device contains three separate transmit message IDs: TXID0, TXID1 and TXID2. The data length code is predefined for each of the various output messages, with the data that is transmitted coming directly from the contents of the device's peripheral registers.

2.5.1.1 Transmit Message ID0 (TXID0)

TXID0 contains the identifier that is used when transmitting the On Bus message. If enabled (STCON.STEN = 1), the On Bus message will be transmitted at predefined intervals. Depending on the message-select bit (STCON.STMS = 1), the CAN message will send GPIO and A/D data.

Transmit Message ID0 will not be sent automatically when the device is brought out of sleep.

2.5.1.2 Transmit Message ID1 (TXID1)

TXID1 contains the identifier that is used when the MCP2502X/5X sends the Command Acknowledge message, the Receive Overflow message and/or the Error Condition message. All message types use the same identifier.

The CAEN bit, in the OPTREG2 register, selects between the Command Acknowledge and Receive Overflow operation. These message types have a DLC of 0 and do not contain any data. The Error Condition message can occur anytime, has a DLC of 3 and contains the EFLG, TEC and REC data values.

Note:	A zero-data-length On Bus message will be transmitted once after Power-up, regardless of the scheduled transmission-enable status.

Command Acknowledge: TXID1 sends a Command Acknowledge message when the MCP2502X/5X receives an Input Message and processes the instruction (and OPTREG2.CAEN = 1). This message is used as a hand shake for the node requesting the modification of the MCP2502X/5X. There is no data associated with this message.

Receive Overflow: TXID1 sends a Receive Overflow message if there is a Receive Overflow condition (and OPTREG2.CAEN = 0). This only occurs if the device has received a valid message before processing the previous valid message from the same receive buffer. There is no data associated with this message.

Error Condition: An Error Condition message is transmitted if the TEC or REC counters reach error warning (> 95) or error passive (> 127). This message contains the TXID1 identifier and the TEC, REC and EFLG counters.

A hysteresis is implemented in hardware that prevents messages from repeatedly being transmitted due to error counts changing by one or two bits. When a message is sent for an error warning (TEC or REC > 95), the message will not trigger again until the error counter ≤ 79 and back to > 95 (hysteresis = 17 counts). Similarly, an error passive message is sent at TEC or REC > 127 and is not sent again until the error counter ≤ 111 and back to >127 (hysteresis = 17 counts).

2.5.1.3 Transmit Message ID2 (TXID2)

Transmit ID2 contains the identifier that is used when transmitting auto-conversion-initiated messages, including digital input edge detection and/or analog input exceeding a threshold. This message will also be sent when the device wakes up from sleep due to a digital input change-of-state condition (i.e., change-of-state occurs on input configured to transmit on change-of-state).

2.6 Receive Buffers

The MCP2505X contains two receive buffers, each with their own filter. There is also a Message Assembly Buffer (MAB) that acts as a third receive buffer (see Figure 2-1).

The two receive buffers, combined with the MAB help, ensure that received messages will be processed while minimizing the chances of receive buffer overrun due to maximum bus loading of messages destined for the MCP2502X/5X.

Note:	The receive buffers are used by the MCP2502X/5X to implement the command messages. They are not externally accessible.

2.7 Acceptance Mask

The acceptance mask is used to define which bits in the CAN ID are to be compared against the programmable filters. Individual bits within the mask correspond to bits in the CAN ID that, in turn, correspond to bits in the acceptance filters. Any bit in the mask that is set to a '1' will cause the corresponding CAN ID bit to be compared against the associated filter bit. Any bit in the mask that is set to a '0' is not compared and effectively sets the associated CAN ID bit to 'don't care'.

2.7.1 MASKS AND STANDARD/ EXTENDED IDS

To insure proper operation of the information request and input messages, some mask bits (as configured in the mask registers) may be ignored as explained:

Message with a standard ID - the three least significant bits of a standard identifier (RXMSIDL.SID2:SID0) are 'don't care' for the mask registers and effectively become '0'.

Message with an extended ID - the three least significant bits of the standard identifier (RXMSIDL.SID2:SID0) are configurable and the three least significant bits of the extended identifier (RXMEID0.EID2:EID0) are always 'don't cares' and effectively becomes '0'.

Note:	The EXIDE bit in the Mask register (RXMSIDL) can be used to mask the IDE bit in the corresponding Receive buffer register (RXBnSIDL).

2.8 Acceptance Filters

There are two separate acceptance filters defined for the MCP2502X/5X: RXF0 and RXF1. RXF0 is used for Information Request messages and RXF1 is used for input messages (see Table 4-2 and Table 4-3). Each bit in the filters corresponds to a bit in the CAN ID. Every bit in the CAN ID, for which the corresponding Mask bit is set, must match the associated filter bit in order for the message to be accepted. Messages that fail to meet the mask/filter criteria are ignored.

REGISTER 2-6: TXIDNSIDH - TRANSMIT IDENTIFIER N STANDARD IDENTIFIER HIGH

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

bit 7-0 **SID10:SID3:** Standard Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-7: TXIDNSIDL - TRANSMIT IDENTIFIER N STANDARD IDENTIFIER LOW

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0**: Standard Identifier bits

bit 4 **Unimplemented**: Read as '0'

bit 3 **EXIDE**: Extended Identifier Enable bit

1 = Message will transmit extended identifier
0 = Message will transmit standard identifier

bit 2 **Unimplemented**: Read as '0'

bit 1-0 **EID17:EID16**: Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-8: TXIDNEID8 - TRANSMIT IDENTIFIER N EXTENDED IDENTIFIER HIGH

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

bit 7-0 **EID15:EID8**: Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-9: TXIDNEID0 - TRANSMIT IDENTIFIER N EXTENDED IDENTIFIER LOW

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-0 **EID7:EID0**: Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-10: RXMSIDH - ACCEPTANCE FILTER MASK STANDARD IDENTIFIER HIGH

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3*
bit 7							bit 0

bit 7-0 **SID10:SID3**: Standard Identifier bits

* If OPTREG2.MTYPE = 1, then SID3 is forced to zero

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-11: RXMSIDL - ACCEPTANCE FILTER MASK STANDARD IDENTIFIER LOW

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0**: Standard Identifier bits
Standard messages, bits = b'000'
Extended messages, bits = SID2:SID0

bit 4 **Unimplemented:** Read as '0'

bit 3 **EXIDE**: Extended Identifier Enable bit
1 = Apply filter to RXFnSIDL.EXIDE (filter applies to standard **or** extended message frames, depending on filter bit)
0 = Do not apply filter to RXFnSIDL.EXIDE (filter will be applied to both standard **and** extended message frames)

bit 2 **Unimplemented:** Read as '0'

bit 1-0 **EID17:EID16:** Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-12: RXMEID8 - ACCEPTANCE FILTER MASK EXTENDED IDENTIFIER MID

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

bit 7-0 **EID15:EID8**: Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-13: RXMEID0 - ACCEPTANCE FILTER MASK EXTENDED IDENTIFIER LOW

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-3 **EID7:EID3**: Extended Identifier bits

bit 2-0 **EID2:EID0**: Extended Identifier bits (always reads as '0')

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-14: RXFNSIDH - ACCEPTANCE FILTER N STANDARD IDENTIFIER HIGH

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3*
bit 7							bit 0

bit 7-0 **SID10:SID3**: Standard Identifier bits

* If OPTREG2.MTYPE = 1, then SID3 = X

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-15: RXFNSIDL - ACCEPTANCE FILTER N STANDARD IDENTIFIER LOW

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

bit 7-5 **SID2:SID0**: Standard Identifier bits
1 = When EXIDE = 1, SID2:SID0 = b'xxx'
0 = When EXIDE = 0, SID2:SID0 = as configured

bit 4 **Unimplemented:** Read as '0'

bit 3 **EXIDE**: Extended Identifier Enable bit
1 = Filter will apply to extended identifier
0 = Filter will apply to standard identifier

bit 2 **Unimplemented:** Read as '0'

bit 1-0 **EID17:EID16**: Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2-16: RXFNEID8 - ACCEPTANCE FILTER N EXTENDED IDENTIFIER MID

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

bit 7-0 **EID15:EID8**: Extended Identifier bits

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

REGISTER 2-17: RXFNEID0 - ACCEPTANCE FILTER N EXTENDED IDENTIFIER LOW

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

bit 7-0 **EID7:EID0**: Extended Identifier bits (always = b’xxx’)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

REGISTER 2-18: EFLG - ERROR FLAG REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
ESCF	RBO	TXBO	TXEP	RXEP	TXWAR	RXWAR	EWARN
bit 7							bit 0

bit 7 **ESCF**: Error State Change bit (for sending Error state message)
1 = An error state change occurred
0 = No error state change

bit 6 **RBO**: Receive Buffer Overflow bit
1 = Overflow occurred
0 = No overflow occurred

bit 5 **TXBO**: Transmitter in Bus Off Error State bit
1 = TEC reaches 256
0 = Indicates a successful bus recovery sequence

bit 4 **TXEP**: Transmitter in Error Passive State bit
1 = TEC is equal to or greater than 128
0 = TEC is less than 128

bit 3 **RXEP**: Receiver in Error Passive State bit
1 = REC is equal to or greater than 128
0 = REC is less than 128

bit 2 **TXWAR**: Transmitter in Error Warning State bit
1 = TEC is equal to or greater than 96
0 = TEC is less than 96

bit 1 **RXWAR**: Receiver in Error Warning State bit
1 = REC is equal to or greater than 96
0 = REC is less than 96

bit 0 **EWARN**: Either the Receive Error counter or Transmit Error counter has reached or exceeded 96 errors
1 = TEC or REC is equal to or greater than 96 (TXWAR or RXWAR = 1)
0 = Both REC and TEC are less than 96

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

NOTES:

3.0 USER REGISTERS

3.1 Description

The MCP2502X/5X allows the user to pre-program registers pertaining to CAN module and device configuration into non-volatile EPROM memory. In this way, the device is initialized to a default state after Power-up. The user registers are transferred to SRAM during the Power-up sequence. Many of the registers are able to be accessed via the CAN bus, once the device establishes a connection with the bus. Additionally, there are 16 user-defined registers that can be used to store information about the part (e.g., serial number, node identifier, etc.). The registers are summarized in Table 3-1.

Note 1: When transferred to RAM, the register addresses are offset by 1Ch. Accessing individual registers using the "Write Register" or "Read Register command requires use of the offset address. Also, see Table 3-2 for information on accessible registers not contained in user EPROM.

2: Do not address locations outside of the user memory map or unexpected results may occur.

TABLE 3-1: USER MEMORY MAP

Address	Name	Description	Address	Name	Description
00h	IOINTEN	Enable inputs for Transmit-On-Change feature	1Bh	RXF0EID0	Acceptance Filter 0, Extended ID LSB
01h	IOINTPO	Defines polarity for I/O, or greater-than/less-than operator for A/D Transmit-On-Change inputs	1Ch	RXF1SIDH	Acceptance Filter 1, Standard ID MSB
02h	GPLAT	General Purpose I/O (GPIO) Register	1Dh	RXF1SIDL	Acceptance Filter 1, Standard ID LSB, Extended ID USB, Extended ID enable
03h	0xFF	Reserved	1Eh	RXF1EID8	Acceptance Filter 1, Extended ID MSB
04h	OPTREG1	Configuration options, including GPIO pull-up enable, clockout enable and prescaler	1Fh	RXF1EID0	Acceptance Filter 1, Extended ID LSB
05h	T1CON	PWM1 Timer Control Register; contains enable bit, clock prescale and DC LSBs	20h	TXID0SIDH	Transmit Buffer 0, Standard ID MSB
06h	T2CON	PWM2 Timer Control Register; contains enable bits, clock prescale and DC LSBs	21h	TXID0SIDL	Transmit Buffer 0, Standard ID LSB, Extended ID USB, Extended ID enable
07h	PR1	PWM1 Period Register	22h	TXID0EID8	Transmit Buffer 0, Extended ID MSB
08h	PR2	PWM2 Period Register	23h	TXID0EID0	Transmit Buffer 0, Extended ID LSB
09h	PWM1DCH	PWM1 Duty Cycle (DC) MSBs	24h	TXID1SIDH	Transmit Buffer 1, Standard ID MSB
0Ah	PWM2DCH	PWM2 Duty Cycle (DC) MSBs	25h	TXID1SIDL	Transmit Buffer 1, Standard ID LSB, Extended ID USB, Extended ID enable
0Bh	CNF1 [3]	CAN module register configures synchronization jump width and baud rate prescaler	26h	TXID1EID8	Transmit Buffer 1, Extended ID MSB
0Ch	CNF2 [3]	CAN module register configures propagation segment, phase segment 1, and determines number of sample points	27h	TXID1EID0	Transmit Buffer 1, Extended ID LSB

Note 1: GPDDR is mapped to 1Fh is SRAM and not offset by 1Ch.
2: User memory (35h-44h) is not transferred to RAM on Power-up and can only be accessed via "Read User Mem" commands.
3: Cannot be modified from initial programmed values.
4: Unimplemented on MCP2502X devices and read 0x00 (exception, ADCON1 = 0x0F).

TABLE 3-1: USER MEMORY MAP (CONTINUED)

Address	Name	Description	Address	Name	Description
0Dh	CNF3 [3]	CAN module register configures phase buffer segment 2, Sleep mode	28h	TXID2SIDH	Transmit Buffer 2, Standard ID MSB
0Eh	ADCON0[4]	A/D Control Register; contains enable, conversion rate, channel select bits	29h	TXID2SIDL	Transmit Buffer 2, Standard ID LSB, Extended ID USB, Extended ID enable
0Fh	ADCON1[4]	A/D Control Register; contains voltage reference source, conversion rate and A/D input enable bits	2Ah	TXID2EID8	Transmit Buffer 2, Extended ID MSB
10h	STCON	Scheduled Transmission Control Register	2Bh	TXID2EID0	Transmit Buffer 2, Extended ID LSB
11h	OPTREG2	Configuration options, including Sleep mode, RTR message and error recovery enables	2Ch	ADCMP3H[4]	Analog Channel 3 Compare Value MSB
12h	—	Reserved	2Dh	ADCMP3L[4]	Analog Channel 3 Compare Value LSb's
13h	—	Reserved	2Eh	ADCMP2H[4]	Analog Channel 2 Compare Value MSB
14h	RXMSIDH	Acceptance Filter Mask, Standard ID MSB	2Fh	ADCMP2L[4]	Analog Channel 2 Compare Value LSb's
15h	RXMSIDL	Acceptance Filter Mask, Standard ID LSB Extended ID USB	30h	ADCMP1H[4]	Analog Channel 1 Compare Value MSB
16h	RXMEID8	Acceptance Filter Mask, Extended ID MSB	31h	ADCMP1L[4]	Analog Channel 1 Compare Value LSb's
17h	RXMEID0	Acceptance Filter Mask, Extended ID LSB	32h	ADCMP0H[4]	Analog Channel 0 Compare Value MSB
18h	RXF0SIDH	Acceptance Filter 0, Standard ID MSB	33h	ADCMP0L[4]	Analog Channel 0 Compare Value LSb's
19h	RXF0SIDL	Acceptance Filter 0, Standard ID LSB, Extended ID USB, Extended ID enable	34h	GPDDR[1]	General Purpose I/O Data Direction Register
1Ah	RXF0EID8	Acceptance Filter 0, Extended ID MSB	35-44h	USER[0:F][2]	User Defined Bytes (0-15)

Note 1: GPDDR is mapped to 1Fh is SRAM and not offset by 1Ch.

2: User memory (35h-44h) is not transferred to RAM on Power-up and can only be accessed via "Read User Mem" commands.

3: Cannot be modified from initial programmed values.

4: Unimplemented on MCP2502X devices and read 0x00 (exception, ADCON1 = 0x0F).

TABLE 3-2: ACCESSIBLE RAM REGISTERS NOT IN THE EPROM MAP

Addr*	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Value on POR	Value on $\overline{RST}$
1Fh**	GPDDR	—	DDR6	DDR4	DDR4	DDR3	DDR2	DDR1	DDR0	-111 1111	-111 1111
18h	EFLG	ESCF	RBO	TXEP	TXEP	RXEP	TXWAR	RXWAR	EWARN	0000 0000	0000 0000
19h	TEC	Transmit Error Counters								0000 0000	0000 0000
1Ah	REC	Receive Error Counters								0000 0000	0000 0000
50h	ADRES3H	AN3.9	AN3.8	AN3.6	AN3.6	AN3.5	AN3.4	AN3.3	AN3.2	xxxx xxxx	uuuu uuuu
51h	ADRES3L	AN3.1	AN3.0	—	—	—	—	—	—	xx-- ----	uu-- ----
52h	ADRES2H	AN2.9	AN2.8	AN2.6	AN2.6	AN2.5	AN2.4	AN2.3	AN2.2	xxxx xxxx	uuuu uuuu
53h	ADRES2L	AN2.1	AN2.0	—	—	—	—	—	—	xx-- ----	uu-- ----
54h	ADRES1H	AN1.9	AN1.8	AN1.6	AN1.6	AN1.5	AN1.4	AN1.3	AN1.2	xxxx xxxx	uuuu uuuu
55h	ADRES1L	AN1.1	AN1.0	—	—	—	—	—	—	xx-- ----	uu-- ----
56h	ADRES0H	AN0.9	AN0.8	AN0.6	AN0.6	AN0.5	AN0.4	AN0.3	AN0.2	xxxx xxxx	uuuu uuuu
57h	ADRES0L	AN0.1	AN0.0	—	—	—	—	—	—	xx-- ----	uu-- ----

* These addresses are used when using the “Write Register” or “Read Register” command

** The GPDDR register is not offset to RAM the same as the other registers in the EPROM

NOTES:

4.0 DEVICE OPERATION

4.1 Power-Up Sequence

The following sections describe the events/actions of the MCP2502X/5X during normal Power-up and operation.

4.1.1 POWER-ON RESET

The MCP2502X/5X goes through a sequence of events at Power-on Reset (POR) to load the programmed configuration and insure that errors are not introduced on the bus. During this time, the device is prevented from generating a low condition on the TXCAN pin. The TXCAN pin must remain high from Power-on until the device goes on bus.

Operational Mode at Power-On

The MCP2502X/5X initially powers up in Configuration mode. While in this mode, the MCP2502X/5X will be prevented from sending or receiving messages via the CAN interface. The ADC and PWM peripherals are disabled while in this mode.

Self-Configuration

Once the MCP2502X/5X is out of Reset, it will perform a self-configuration. This is accomplished by transferring the contents of the EPROM array to the corresponding locations within the SRAM array. In addition, the checksum of the data written to SRAM will be compared to a pre-programmed value as a test of valid data.

Going On Bus

Once the self-configuration cycle has successfully completed, the MCP2502X/5X switches to Listen-only mode. It will remain in this mode until an error-free CAN message is detected. This is done to ensure that the device is at the correct bus rate for the system.

Once the device detects an error-free message, it waits for CAN bus idle before switching to Normal mode. This prevents it from going on bus in the middle of another node's transmission and generating an error frame.

Alternately, the MCP2505X may directly enter Normal mode, without first entering Listen-only Mode, after completing its self-configuration. This is configured by the user via a control bit (OPTREG2.PUNRM).

Once the MCP2502X/5X enters Normal mode, it is ready to send/receive messages via the CAN interface. At this point the ADC and PWM peripherals are operational, if enabled.

Scheduled Transmissions

When the MCP2502X/5X has gone on bus it will transmit the On Bus message once, regardless of whether it is enabled or not. This message notifies the network of the MCP2502X/5X's presence. The On Bus message will (if enabled, STCON.STEN) repeat at a frequency determined by the STCON register (Register 4-1).

This message can also be configured to send the "Read A/D Register" data bytes with the predefined identifier in TXID2 by setting STCON.STMS = `1`.

Notes: The first On Bus message sent after Power-up will NOT send the "Read A/D Register" data bytes, regardless of the STCON.STMS value.

If the MCP2502X/5X enters SLEEP mode, the scheduled transmissions will cease until the device wakes up again. This implies that SLEEP mode has priority over scheduled transmissions.

4.2 Message Functions

The MCP2502X/5X uses the global mask (RXMASK), two filters (RXF0 and RXF1), and two receive buffers (RB0 and RB1) to determine if a received message should be acted on. There are 16 functions that can be performed by the MCP2502X/5X, based on received messages (see Table 4-1).These functions allow the device to be accessed for Information Request/Input/Output operations, but also to be reconfigured via the CAN bus, if necessary.

4.3 Message Types

There are three types of messages that are used to implement the functions of Table 4-1.

- Information Request Messages (IRMs):
  Received by the MCP2502X/5X
- Output Messages:
  Transmitted from the MCP2502X/5X as a response to IRMs
- Input Messages:
  Received by the MCP2502X/5X and used to modify registers

Notes: IRMs and Input messages are both input messages to the MCP2502X/5X.

IRMs are received into receive buffer 0 Input Messages are received into receive buffer 1.

This must be taken into account while configuring the acceptance filters.

4.3.1 INFORMATION REQUEST MESSAGES

IRMs are messages that are received by the the MCP2502X/5X into the Receive Buffer 0 (matches Filter 0); and then, responded to by transmitting a message (output message) containing the requested data.

IRMs can be implemented as either a Remote Transfer Request (RTR) or a Data Frame message by configuring the MTYPE bit in the OPTREG2 register.

TABLE 4-1: MESSAGE FUNCTION

Name	Description
Read A/D Registers	Transmits a single message containing the current state of the analog and I/O registers, including the configuration
Read Control Registers	Transmits several control registers not included in other messages
Read Configuration Registers	Transmits the contents of many of the Configuration registers
Read CAN error states	Transmits the error flag register and the error counts
Read PWM Configuration	Transmits the registers associated with the PWM modules
Read User Registers 1	Transmits the values in bytes 0 - 7 of the user memory
Read User Registers 2	Transmits the values in bytes 8 -15 of the user memory
Read Register*	Transmits a single byte containing the value in an addressed user memory register
Write Register	Uses a mask to write a value to an addressed register
Write TX Message ID0 (TXID0)	Writes the identifiers to a specified value
Write TX Message ID1 (TXID1)	Writes the identifiers to a specified value
Write TX Message ID2 (TXID2)	Writes the identifiers to a specified value
Write I/O Configuration Registers	Writes specified values to the three IOCON registers
Write RX Mask	Changes the receive mask to the specified value
Write RX Filter0	Changes the specified filter to the specified value
Write RX Filter1	Changes the specified filter to the specified value

* The Read Register command is available when using extended message format only. Not available with standard message format.

4.3.1.1 RTR Message Type

When RTR message types are selected (OPTREG2.MTYPE) and a node in the system wants information from the MCP2502X/5X, it has to send a remote frame on the bus. The identifier for the remote frame must be such that it will be accepted through the MCP2502X/5X's mask/filter process (using RXF0). The RTR message type (remote frames) is the default configuration (MTYPE bit = 0).

Information Request "RTR" messages must not only meet the RXMASK/RXF0 criteria but must also have the RTR bit of the CAN ID set (since the filter registers do not contain an explicit RTR bit). If a message passes the mask/filter process and the RTR bit is a '0', that message will be ignored.

Once the MCP2502X/5X has received a remote frame, it will determine the function to be performed based upon the three LSb's (RXB0SIDL.SID2:SID0 for standard messages and RXB0EID0.EID2:EID0 for extended messages) of the received remote frame.

Additionally, a predefined Data Length Code (DLC) must be sent to signify the number of data bytes that the MCP2502X/5X must return in the output message (see Table 4-2 and Table 4-3).

4.3.1.2 Data Frame Message Type

When a non-RTR (or data frame) message type is selected and a node in the system wants information from the MCP2502X/5X, it sends an Information Request in the form of a data frame. The identifier for this request must be such that it will be accepted through the MCP2502X/5X's mask/filter process (using RXF0).

Information request messages in the data frame format must not only meet the RXMASK/RXF0 criteria, but must also have the RTR bit of the CAN ID cleared (since the filter registers do not contain an explicit RTR bit). If a message passes the mask/filter process and the RTR bit is a '1', that message will be ignored.

Once the MCP2502X/5X has received a data frame information request, it will determine the function to be performed based upon the three LSb's (RXB0SIDL.SID2:SID0 for standard messages and RXB0EID0.EID2:EID0 for extended messages) of the received data frame. Also, Bit 3 of the received message ID must be set to a '1'.

In addition, the data length code (DLC) must be set to a zero. Refer to Table 4-2 and Table 4-3 for more information.

Regardless of the message format, all messages (except the Read Register message) can use either standard or extended identifiers.

The Read Register message has one additional requirement; it must be an extended identifier. This is discussed in more detail in Table 4-1 and Table 4-3.

4.3.2 OUTPUT MESSAGES

The data frame sent in response to the information request message is defined as an output message.

If the data fame is in response to a remote frame, it has the same identifier (standard or extended) and contains the same number of data bytes specified by the DLC of the remote frame (per the CAN 2.0B specification).

Note: If the DLC of the incoming remote frame differs from the message definitions summarized in Table 4-2 and Table 4-3, the resulting output message will limit itself to the erroneous DLC that was received (to maintain compliance with the Bosch CAN specification). The output message will concatenate the number of data bytes for an erroneous DLC that is less than the defined number. For an erroneous DLC that is greater than the defined number, the MCP2502X/5X will extend the number of data bytes, with the data value of the last defined data byte being repeated in the extra bytes in the data field.

If the output message is in response to a data frame, the lower-three LSb's of the identifier (standard or extended) must be the same as the received message, as well as the upper-seven MSb's in the case of a standard identifier, or the upper 25 MSb's in the case of an extended identifier. Bit 3 of a standard or extended identifier of the output message will differ from the received information request message in that the value equals '1' for an IRM and equals '0' for the resulting output message.

Output messages contain the requested data (in the data field). **Example:** The information request message Read CAN error is a remote transmit request received by the MCP2502X/5X with a DLC of 3. The responding output message will return a data frame that contains the same identifier (standard or extended) as the receive message. The accompanying data bytes will contain the values of the predefined GPIO registers and related control/status registers, as shown in Table 4-2 and Table 4-3.

4.3.3 INPUT MESSAGES

Input messages are received into receive buffer 1 and are used to change the values of the pre-defined groups of registers. There is also an input message that can change a single register's contents. The primary purpose of input messages are to reconfigure MCP2502X/5X parameters (if needed) while in an operating CAN system and are, therefore, optional in system implementation. These messages are in the form of standard (or extended) data frames (per the CAN 2.0B specification) that have identifiers which pass the MCP2502X/5X's mask/filter process (using RXF1).

After passing the mask and filter, the lower-three bits of the standard identifier (RXF1SIDL.SID2:SID0) will indicate which register(s) are to be written. The values for the register(s) are contained in the data byte registers as defined in Table 4-2.

Note: If using more than one controlling node, the MCP2502X/5X must be set up to accept input messages with different identifiers in order to avoid possible message collisions in the DLC or data bytes if transmitted at the same time.

Notes: IRMs can theoretically be sent by more than one controlling node because the message is a predefined constant and destructive collisions will not occur.

The number of data bytes in an input message must match the DLC number as defined in Table 4-2 and Table 4-3. If the user specifies and transmits an input message with a DLC that is less than the required number of data bytes, the MCP25020 will operate on corrupted data for the bytes that it did not receive and unknown results will occur.