INTRODUCTION

The MCP2510 stand-alone CAN controller was originally developed to give CAN system and module designers more flexibility in their design by allowing them to choose the best processor for their application. By using the MCP2510, designers were not restricted to using processors with integrated CAN controllers.

Today, the CAN market continues to grow and proliferate into other markets and different applications and, both increasingly complex nodes and simpler nodes are being developed to further distribute control among the CAN network. The complex nodes may require using a 32-bit MCU, ASIC, CPLD, DSP or some other device that does not have an on-board CAN controller. The simple nodes may only require small program space and not need all of the extra peripherals found on many of the MCUs with integrated CAN.

The MCP2515 addresses these new market needs, and is designed to be pin and functionally compatible to the MCP2510. All known MCP2510 errata have been addressed in the MCP2515. Additionally, there are several enhancements with the MCP2515, designed for increased performance.

While the MCP2515 was designed to be functionally compatible to the MCP2510, there are some differences between the two devices due to both the MCP2510 errata being fixed and the enhanced features of the MCP2515. These differences should be invisible in most applications that choose to upgrade to the MCP2515. This application note discusses the differences between the MCP2510 and MCP2515 (and the possible impact of these differences) in an effort to assist with the upgrade process.

MCP2515 ENHANCEMENTS AND DIFFERENCES

Enhancements

The enhancements in the MCP2515 are designed as a super-set to the basic functionality of the MCP2510. These enhancements include:

- 40 MHz operation
- 10 MHz Serial Peripheral Interface™ (SPI™)
- Data byte filtering on the first 16 bits in the data field (standard 11-bit frames only)
- One-shot mode to automatically abort messages that lose arbitration or are interrupted by an error frame
- Start-of-Frame (SOF) output pin used to detect valid start-of-frames
- Three new SPI instructions:
  - Read RX Buffer Command
    Eliminates the eight bit address required by a normal read command.
    Eight bit instruction that sets the address pointer to one of four addresses depending on two bits. Points to the "ID" or "data" of the two receive buffers.
  - RX Status Command
    Used to quickly read important information about a received message.
    Eight bit instruction followed by the status of received message: Standard/Extended, Frame Type (data frame/remote) and filter match.
  - Load TX Buffer Command
    Eliminates the eight bit address required by a normal write command.
    Eight bit instruction that sets the address pointer to one of six addresses to quickly write to a transmit buffer. Points to the "ID" or "data" address of any of the three transmit buffers.

Differences

A summary of the differences (including the enhancements and other changes) is shown in Table 1. The sections following the table describe each difference in greater detail.
**TABLE 1: MCP2510 TO MCP2515 UPGRADE COMPARISON**

<table>
<thead>
<tr>
<th>Description</th>
<th>MCP2515</th>
<th>MCP2510</th>
<th>Upgrade Impact</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fosc (max)</td>
<td>40 MHz</td>
<td>25 MHz</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>SPI clock (max)</td>
<td>10 MHz</td>
<td>5 MHz</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>2.7V to 5.5V</td>
<td>3.0V to 5.5V</td>
<td>None</td>
<td>May affect operation if RXNnEID8 and RXMnEID0 are initialized to non-zero values.</td>
</tr>
<tr>
<td>Data byte filtering</td>
<td>The mask registers POR state is zero (i.e., masks are off).</td>
<td>None. The mask registers POR state is unknown.</td>
<td>Minimal</td>
<td>The OSM bit is in the CANCTRL register (unused in the MCP2510, bit default = 0).</td>
</tr>
<tr>
<td>One-shot mode</td>
<td>Ensures that the transmit message is attempted only one time.</td>
<td>Not implemented</td>
<td>Minimal</td>
<td>The SOF signal control bit is in the CNF3 register (unused in the MCP2510, bit default = 0).</td>
</tr>
<tr>
<td>SOF signal</td>
<td>Generates a pulse output at the beginning of a message.</td>
<td>Not implemented</td>
<td>Minimal</td>
<td></td>
</tr>
<tr>
<td>Three new SPI Instructions</td>
<td>Speeds up data throughput.</td>
<td>Not implemented</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Clocks on CLKOUT before sleep</td>
<td>17</td>
<td>16</td>
<td>None</td>
<td>One extra clock pulse with the MCP2515 before going to sleep.</td>
</tr>
<tr>
<td>Setting ABAT bit</td>
<td>Sets abort flag (ABTF) regardless if TXREQ.</td>
<td>Sets abort flag (ABTF) only if TXREQ is set</td>
<td>Minimal</td>
<td>MCP2515 clears TXREQ without checking if set.</td>
</tr>
<tr>
<td>Aborting pending messages</td>
<td>Can abort any pending message.</td>
<td>Can only abort pending messages that have not attempted to transmit.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Error warning flags (EWARN and RXWARN)</td>
<td>Flags do not clear when transitioning to receive error passive.</td>
<td>Flags will clear if device transitions to receive error passive.</td>
<td>Minimal</td>
<td>The INT pin operation will remain the same if flags are enabled.</td>
</tr>
<tr>
<td>Sleep mode</td>
<td>REQOP changes to b'011' after entering sleep. OPMOD indicates Sleep mode.</td>
<td>REQOP = OPMOD = b'001'</td>
<td>Minimal</td>
<td>REQOP is only used to request operation modes. OPMOD is used to determine the mode.</td>
</tr>
<tr>
<td>REQOP bits while in sleep</td>
<td>REQOP bits are readable and writable.</td>
<td>None</td>
<td>Neither device can wakeup from sleep by modifying the REQOP bits.</td>
<td></td>
</tr>
<tr>
<td>Standby current</td>
<td>8 µA max at 125°C</td>
<td>5 µA max all temps</td>
<td>Minimal</td>
<td>MCP2515 standby currents are similar to the MCP2510 at all other temperatures.</td>
</tr>
<tr>
<td>SPI Bit Modify Command</td>
<td>Using command on other registers forces mask = FFh.</td>
<td>Can only use for specific registers.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Error counters</td>
<td>Does not reset when entering Listen-only mode.</td>
<td>Reset when entering Listen-only mode.</td>
<td>Minimal</td>
<td>Error counters deactivate on both devices while in Listen-only mode.</td>
</tr>
<tr>
<td>Reading masks and filters</td>
<td>Can only read in Configuration mode. Reads 00h in other modes.</td>
<td>Can read in any mode.</td>
<td>Minimal</td>
<td>The masks and filters will typically only be read while in Configuration mode.</td>
</tr>
</tbody>
</table>
FOSC

The maximum frequency of operation for the MCP2510 is 25 MHz (16 MHz for low voltage), whereas the maximum for the MCP2515 is 40 MHz (25 MHz for low voltage).

There is no impact when upgrading to the MCP2515.

SPI Clock

The maximum SPI frequencies for the MCP2510:
- 5 MHz for VDD > 4.5V
- 4 MHz for E-temp VDD > 4.5V
- 2.5 MHz for VDD = 3.0 to 4.5V

The maximum SPI clock frequency for the MCP2515 is 10 MHz across all voltages and temperatures.

There is no impact when upgrading to the MCP2515.

Operating Voltage

The MCP2510 operates from 3.0V to 5.5V, while the MCP2515 operates from 2.7V to 5.5V.

There is no impact when upgrading to the MCP2515.

Data Byte Filtering

When receiving standard data frames (11-bit identifier), the MCP2515 automatically applies 16 bits of the masks and filters normally associated with extended identifiers to the first 16-bits of the data field (data bytes 0 and 1). The MCP2510 does not have this feature.

The difference between the MCP2510 and MCP2515 is the POR default state of the extended mask registers (RXMnEID8 and RXMnEID0). The MCP2510 POR defaults are undefined and can power-up in any state. The MCP2515 POR defaults equals zero for these registers to effectively turn the masks off (i.e., do not apply filters to the data bytes).

If the original application with the MCP2510 does not use extended frames and does not initialize the extended mask registers (or initializes them to zero), the MCP2515 can be placed in the socket with no MCU firmware modifications.

One-shot Mode

The MCP2515 implements a feature to ensure that a transmit message is attempted only one time. With One-shot mode enabled, a message will attempt transmission only one time, regardless of arbitration loss or error frame.

This enable bit is located in CANCTRL.bit3. This location is unused and reads zero in the MCP2510.

If the original application does not attempt to initialize this location to a logic one (which it should not because the bit is unimplemented in the MCP2510), then using the MCP2515 will have no effect on the operation.

Start-of-Frame (SOF) Signal

The MCP2515 implements a feature that, if enabled, will generate a pulse on the CLKOUT/SOF pin if the RXCAN pin detects the beginning of a CAN message.

The SOF bit is located in CNF3.bit7. This location is unused and reads zero on the MCP2510.

If the original application does not attempt to initialize this location to a logic one (which it should not because the bit is unimplemented in the MCP2510), then using the MCP2515 will have no effect on the operation.

Three New SPI Instructions

See the "Enhancements" section and the MCP2515 data sheet for details.

Number of Clocks on CLKOUT Pin Before Entering Sleep Mode

After requesting Sleep mode, the MCP2510 generates 16 additional clocks on CLKOUT (if enabled) before entering Sleep mode. The MCP2515 generates 17 additional clocks.

Setting ABAT Bit to Abort Messages

The MCP2510 will only set the abort flag (TXBnCTRL.ABTF) when requesting an abort via CANCTRL.ABAT if the associated message was pending (TXREQ = 1) and then successfully aborted.

The MCP2515 sets the abort flag (TXBnCTRL.ABTF) regardless of the associated TXREQ value. However, the MCP2515 will abort the message if it is pending.

Using the MCP2515 in an application designed for the MCP2510 will have very little impact because the MCP2515 is better at aborting messages (see “Aborting Pending Messages”).

Aborting Pending Messages

The MCP2510 can only abort messages that are pending and have not attempted to transmit. This includes messages that go back to the pending state due to loss of arbitration, error frames, etc. This is because the TXBnCTRL.TXREQ bit gets locked out and cannot be cleared if the associated buffer attempts to transmit. The only exception is if another transmit buffer becomes pending and has a higher buffer priority.

The MCP2515 can abort any pending message. Setting CANCTRL.ABAT will clear the associated TXREQ bit. If the transmitting buffer is interrupted, it checks the TXREQ bit before attempting to transmit again, and if cleared, will not attempt to transmit.

The enhanced aborting capabilities of the MCP2515 should have minimal affect when replacing the MCP2510.
Error Warning Flags (EWARN and RXWARN)

The EWARN and RXWARN flag bits, located in EFLG, will clear if the MCP2510 transitions from error-warning to error-passive.

For the MCP2515, the EWARN and RXWARN bits stay set if the device transitions to error-passive.

The impact when upgrading to the MCP2515 should be minimal because an interrupt is generated (if enabled) whenever either condition is true. If polling for the error condition, it is possible (though not probable) that the firmware could mistake an error-passive state as an error-warning state.

Sleep Mode

To enter Sleep mode with either device, the CANCTRL.REQOP bits equal b'001'. Once in Sleep mode, the REQOP bits remain unchanged in the MCP2510. However, the MCP2515 REQOP bits will change to b'011' to request Listen-only mode as soon as the device wakes up from Sleep mode. Note that the CANSTAT.OPMOD bits still reflect the current mode, which is Sleep in this case.

The MCP2515 should have minimal affect on the application when replacing the MCP2510 because the application should read CANSTAT.OPMOD when checking the operation mode. The REQOP bits are only used for requesting modes of operation, not verifying modes.

Modifying REQOP Bits While In Sleep Mode

The CANCTRL.REQOP bits are writable on the MCP2510 while in Sleep mode. The REQOP bits are read-only on the MCP2515 while in Sleep mode.

The impact of upgrading to the MCP2515 should be minimal because the modes cannot be changed on either device while in Sleep mode.

Requesting Sleep Mode

When requesting Sleep mode, the MCP2510 will immediately enter Sleep mode, regardless of bus activity. The MCP2515 will wait until a bus idle condition before entering Sleep mode.

There should be no negative impact when upgrading to the MCP2515.

Standby Current

The maximum standby (Sleep mode) current on the MCP2510 is 5 µA across all temperatures. The maximum standby current on the MCP2515 is 5 µA for temperatures up to 85°C and 8 µA for temperatures from 85°C to 125°C.

The impact of an upgrade should be minimal because the typical currents between the two devices are extremely similar.

SPI Bit Modify Command

On the MCP2510, the Bit Modify command can only be used on specific registers, as identified in the device's data sheet. While this is essentially true for the MCP2515 as well, if a Bit Modify command is used on a register whose bits cannot be modified, the mask byte is ignored and effectively becomes FFh. The command is basically a byte write command with eight extra clocks (mask byte).

There should be no impact when upgrading to the MCP2515 because the MCP2510 application would not attempt to Bit Modify a register whose bits cannot be modified.

Error Counters While In Listen-only mode

The MCP2510 error counters are reset and deactivated while in Listen Only mode. The MCP2515 error counters are not reset, but are still deactivated, while in Listen-only mode.

The impact when upgrading to the MCP2515 should be minimal.

Reading The Mask And Filters While Not In Configuration Mode

The MCP2510 can read the masks and filters in all modes, while the MCP2515 can only read the masks and filters while in Configuration mode. The registers will read 00h while not in Configuration mode. This serves as a positive lockout for the other modes.

The impact when upgrading should be minimal because the masks and filters on either device can be modified only when in Configuration mode. The masks and filters most likely will not need to be read after leaving Configuration mode.

SUMMARY

While the MCP2515 was designed to be pin and functionally compatible with the MCP2510, there are some differences between the devices due to enhancements, errata fixes, design differences, process differences, etc. that the MCP2515 incorporates.

This application note helps the design engineer determine the impact of upgrading their system or module from an MCP2510 to a MCP2515. In most cases, the impact should be nonexistent (or invisible) because the functional differences are a superset of the MCP2510 functionality.
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, dsPIC® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, KEELOQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Accuron, Application Maestro, dsPICDEM, dsPICDEM.net, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICKt, PICDEM, PICDEM.net, PowerCal, PowerInfo, PowerMate, PowerTool, rILAB, rIPIC, Select Mode, SmartSensor, SmartShunt, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2003, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.