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

dsPIC33EV 5V CAN-LIN Starter Kit User's Guide

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.

QUALITY MANAGEMENT SYSTEM CERTIFIED BY DNV

ISO/TS 16949

Trademarks

ISBN: 978-1-63276-749-3

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Object of Declaration: dsPIC33EV 5V CAN-LIN Starter Kit (DM33018)

EU Declaration of Conformity

Manufacturer: Microchip Technology Inc.
2355 W. Chandler Blvd.
Chandler, Arizona, 85224-6199
USA

This declaration of conformity is issued by the manufacturer.

The development/evaluation tool is designed to be used for research and development in a laboratory environment. This development/evaluation tool is not a Finished Appliance, nor is it intended for incorporation into Finished Appliances that are made commercially available as single functional units to end users under EU EMC Directive 2004/108/EC and as supported by the European Commission's Guide for the EMC Directive 2004/108/EC (8th February 2010).

This development/evaluation tool complies with EU RoHS2 Directive 2011/65/EU.

This development/evaluation tool, when incorporating wireless and radio-telecom functionality, is in compliance with the essential requirement and other relevant provisions of the R&TTE Directive 1999/5/EC and the FCC rules as stated in the declaration of conformity provided in the module datasheet and the module product page available at www.microchip.com.

For information regarding the exclusive, limited warranties applicable to Microchip products, please see Microchip's standard terms and conditions of sale, which are printed on our sales documentation and available at www.microchip.com.

Signed for and on behalf of Microchip Technology Inc. at Chandler, Arizona, USA

Derek Carlson
VP Development Tools

12-Sep-14
Date

NOTES:

Table of Contents

Preface

NOTICE TO CUSTOMERS

All documentation becomes dated, and this manual is no exception. Microchip tools and documentation are constantly evolving to meet customer needs, so some actual dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site (www.microchip.com) to obtain the latest documentation available.

Documents are identified with a "DS" number. This number is located on the bottom of each page, in front of the page number. The numbering convention for the DS number is "DSXXXXXA", where "XXXXX" is the document number and "A" is the revision level of the document.

For the most up-to-date information on development tools, see the MPLAB® IDE online help. Select the Help menu, and then Topics to open a list of available online help files.

INTRODUCTION

This chapter contains general information that will be useful to know before using the dsPIC33EV 5V CAN-LIN Starter Kit. Items discussed in this chapter include:

- Document Layout
- Conventions Used in this Guide
- Warranty Registration
- Recommended Reading
- The Microchip Web Site
- Development Systems Customer Change Notification Service
- Customer Support
- Revision History

DOCUMENT LAYOUT

This document describes how to use the dsPIC33EV 5V CAN-LIN Starter Kit as a development tool to emulate and debug firmware on a target board, as well as how to program devices. The document is organized as follows:

- **Chapter 1. "Introduction to the Starter Kit"** provides a brief overview and hardware description of the Starter Kit.
- **Chapter 2. "The Demonstration Application"** describes the Starter Kit's pre-programmed application.
- **Chapter 3. "Modifying the Application"** describes how to use the MPLAB® X IDE to make changes to the demo application
- **Chapter 4. "Troubleshooting"** describes common issues and their solutions.
- **Appendix A. "Starter Kit Schematics"** provides detailed schematics for the Starter Kit.

CONVENTIONS USED IN THIS GUIDE

This manual uses the following documentation conventions:

DOCUMENTATION CONVENTIONS

<table>
<tr><th>Description</th><th>Represents</th><th>Examples</th></tr>
<tr><td colspan="3">Arial font:</td></tr>
<tr><td rowspan="2">Italic characters</td><td>Referenced books</td><td>MPLAB® IDE User's Guide</td></tr>
<tr><td>Emphasized text</td><td>...is the only compiler...</td></tr>
<tr><td rowspan="3">Initial caps</td><td>A window</td><td>the Output window</td></tr>
<tr><td>A dialog</td><td>the Settings dialog</td></tr>
<tr><td>A menu selection</td><td>select Enable Programmer</td></tr>
<tr><td>Quotes</td><td>A field name in a window or dialog</td><td>"Save project before build"</td></tr>
<tr><td>Underlined, italic text with right angle bracket</td><td>A menu path</td><td><u>File>Save</u></td></tr>
<tr><td rowspan="2">Bold characters</td><td>A dialog button</td><td>Click OK</td></tr>
<tr><td>A tab</td><td>Click the Power tab</td></tr>
<tr><td>Text in angle brackets < ></td><td>A key on the keyboard</td><td>Press <Enter>, <F1></td></tr>
<tr><td colspan="3">Courier New font:</td></tr>
<tr><td rowspan="7">Plain Courier New</td><td>Sample source code</td><td>#define START</td></tr>
<tr><td>Filenames</td><td>autoexec.bat</td></tr>
<tr><td>File paths</td><td>c:\mcc18\h</td></tr>
<tr><td>Keywords</td><td>_asm, _endasm, static</td></tr>
<tr><td>Command-line options</td><td>-Opa+, -Opa-</td></tr>
<tr><td>Bit values</td><td>0, 1</td></tr>
<tr><td>Constants</td><td>0xFF, 'A'</td></tr>
<tr><td>Italic Courier New</td><td>A variable argument</td><td>file.o, where file can be any valid filename</td></tr>
<tr><td>Square brackets [ ]</td><td>Optional arguments</td><td>mcc18 [options] file [options]</td></tr>
<tr><td>Curly brackets and pipe character: { | }</td><td>Choice of mutually exclusive arguments; an OR selection</td><td>errorlevel {0|1}</td></tr>
<tr><td rowspan="2">Ellipses...</td><td>Replaces repeated text</td><td>var_name [, var_name...]</td></tr>
<tr><td>Represents code supplied by user</td><td>void main (void)
{ ...
}</td></tr>
</table>

WARRANTY REGISTRATION

Please complete the enclosed Warranty Registration Card and mail it promptly. Sending in the Warranty Registration Card entitles users to receive new product updates. Interim software releases are available at the Microchip web site.

RECOMMENDED READING

This user's guide describes how to use dsPIC33EV 5V CAN-LIN Starter Kit. Other useful documents are listed below. The following Microchip documents are available and recommended as supplemental reference resources.

Readme Files

For the latest information on using other tools, read the tool-specific Readme files in the Readmes subdirectory of the MPLAB X IDE installation directory. The Readme files contain update information and known issues that may not be included in this user's guide.

dsPIC33 Family Reference Manuals

This reference manuals explain the operation of the dsPIC33 digital signal controller family architecture and peripheral modules. The specifics of each device family are discussed in the individual family's device data sheet.

This useful manual is on-line in sections at the Technical Documentation section of the Microchip website. Refer to these for detailed information on dsPIC33 device operation.

dsPIC33EVXXXGM00X/10X Family Data Sheet (DS70005144) and dsPIC33EVXXXGM00X/10X Flash Programming Specification (DS70005137)

Refer to this device data sheet for device-specific information and specifications. Also, refer to the appropriate device flash programming specification for information on instruction sets and firmware development. These files may be found on the Microchip website or from your local sales office.

MPLAB® XC16 C Compiler User's Guide and Libraries (DS50002071)

This document describes the usage of Microchip's MPLAB XC16 C Compiler for application development.

MPLAB® X IDE User's Guide (DS50002027)

This document describes how to use the MPLAB X IDE, Microchip's latest version of its integrated development environment, as well as the MPLAB Project Manager, MPLAB Editor and MPLAB SIM Simulator. Use these development tools to help you develop and debug application code.

THE MICROCHIP WEB SITE

Microchip provides online support via our web site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

DEVELOPMENT SYSTEMS CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

The Development Systems product group categories are:

- **Compilers** – The latest information on Microchip C compilers, assemblers, linkers and other language tools. These include all MPLAB C compilers; all MPLAB assemblers (including MPASM™ assembler); all MPLAB linkers (including MPLINK™ object linker); and all MPLAB librarians (including MPLIB™ object librarian).
- **Emulators** – The latest information on Microchip in-circuit emulators.
- **In-Circuit Debuggers** – The latest information on the Microchip in-circuit debuggers. This includes MPLAB ICD 3 and PICkit 3 debuggers.
- **MPLAB® X IDE** – The latest information on Microchip MPLAB X IDE development environment for all PIC® MCUs. MPLAB X IDE is available for Windows, iOS and Linux operating systems.
- **Programmers** – The latest information on Microchip programmers, including the PM3 production-volume device programmer.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at:

http://www.microchip.com/support.

REVISION HISTORY

Revision A (October 2014)

This is the initial release of this document.

NOTES:

Chapter 1. Introduction to the Starter Kit

Thank you for purchasing the dsPIC33EV 5V CAN-LIN Starter Kit. This board features the dsPIC33EV256GM106 Digital Signal Controller (DSC) for automotive and motor control applications. The Starter Kit contains serial data ports for CAN, LIN and SENT, a self-contained USB programming/debug interface, and an expansion footprint for flexibility in application hardware development.

This chapter introduces the Starter Kit and provides an overview of its features. Topics covered include:

- Overview
- What's In the Kit
- Hardware
- Installing Drivers for the Starter Kit

1.1 OVERVIEW

The dsPIC33EV 5V CAN-LIN Starter Kit is a stand-alone demonstration board, allowing users to explore three popular automotive and industrial serial data formats (CAN, LIN and SENT). The board uses the 5 VDC supplied by the host computer's USB interface, consuming approximately 70 mA. The PICkit On-Board (PKOB) USB programmer and debugger allows simple programming without the need for an additional hardware interface. No other external tools are required to program the device. An optional ICSP™ interface also allows for debugging with more advanced tools.

The Starter Kit board includes several analog features to showcase the capabilities of the dsPIC33EV256GM106 family. The included 64-pin digital signal controller integrates the following features:

- One CAN module with 32 buffers and 16 receive filters
- Two SENT (Single Edge Nibble Transmission) modules
- Two UARTs with LIN/J2606 support
- SPI and I^2C™ serial interfaces
- One high-speed (1.1 Msps), 10/12-bit A/D converter with multiple input channels
- One Charge Time Measurement Unit (CTMU)
- Six Pulse-Width Modulation (PWM) outputs

In addition, the CAN-LIN Starter Kit board also provides:

- Analog temperature sensor (MCP9701A)
- Ratiometric trim potentiometer
- CAN Interface (MCP2561) with DB9 male connector
- LIN Interface (MCP2021A) with required interface electronics
- Three general purpose momentary push buttons
- Three general purpose LEDs
- Breakout Connector footprint

1.2 WHAT'S IN THE KIT

The dsPIC33EV 5V CAN-LIN Starter Kit contains the following:

- dsPIC33EV 5V CAN-LIN Starter Kit board
- USB cable (A to mini-B)
- An insert card with links to the website for this manual, schematics and the demo application

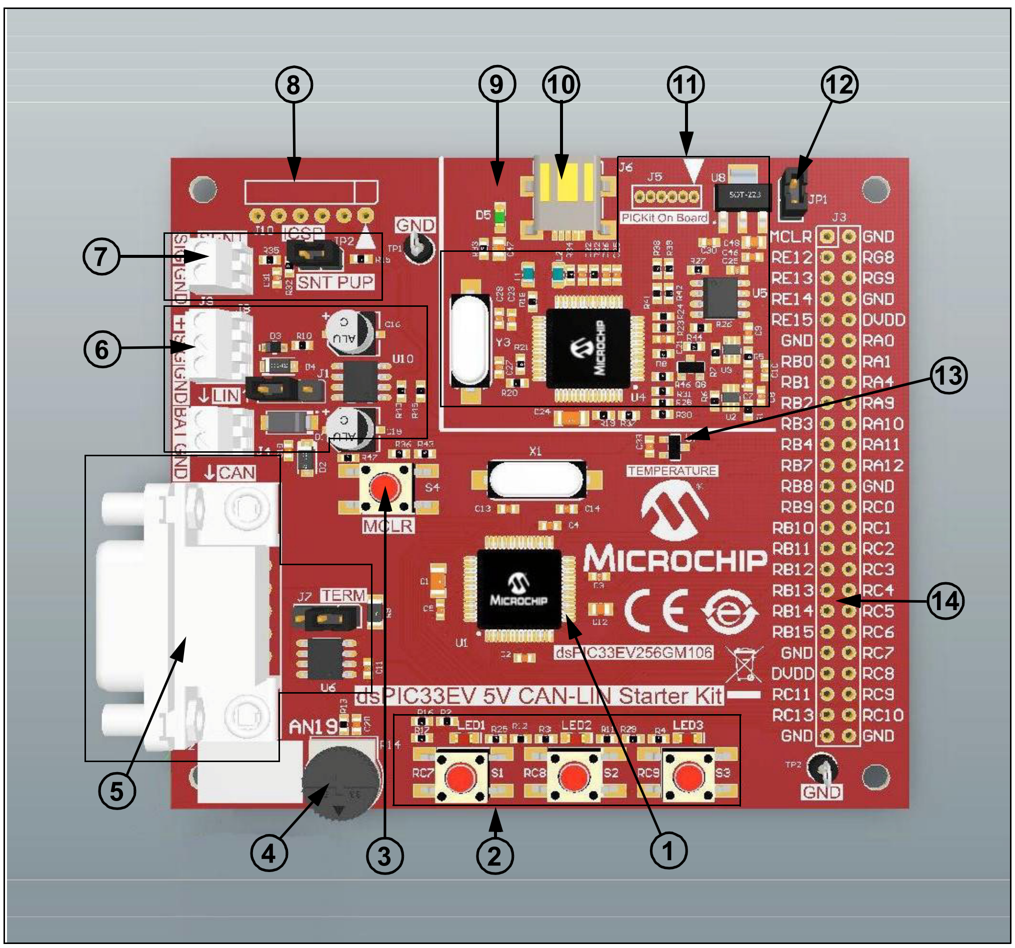
1.3 STARTER KIT HARDWARE

The CAN-LIN Starter Kit ships with a simple pre-programmed application to transmit or receive data packets that simulate sensor readings, using the three automotive data ports. The data transmitted is the temperature reading from the on-board temperature sensor, the 12-bit data from the potentiometer, and the status of the three user push buttons. External protocol analyzers can be used to read the transmitted data. In addition, the secondary UART channel sends an activity log of transmitted/received data via I/O pin RB4. See **Section 2.3 "UART Monitoring Log"** for details.

To develop code and to program and debug the dsPIC33EV256GM106 DSC, you will need the latest versions of MPLAB X IDE (V2.10 or later) and MPLAB XC16 compiler (V1.23 or later).

Figure 1-1 shows the Starter Kit board, along with the major hardware features.

FIGURE 1-1: dsPIC33EV 5V CAN-LIN STARTER KIT FEATURES



1. **dsPIC33EV256GM106 Digital Signal Controller (U1)**: this DSC is at the heart of the application. It is responsible for managing communications for all three serial protocols, using dedicated hardware peripherals for CAN and SENT, and one of the DSC's compatible UARTs for LIN. An external 8 MHz crystal (X1) provides stable timing. All of the I/O pins not used by the sensors and serial interfaces are available on the breakout connector.
2. **User Push Buttons and LEDs:** three push buttons (S1 through S3) are provided to simulate digital sensor data in the pre-programmed application. The switches and their associated red LEDs (LED1 through LED3) are connected to dedicated I/O port pins. The switches have external pull-ups, and thus read a logic low when depressed. The LEDs are lit when their port pins are driven high. The LEDs are low-current types, consuming approximately 2 mA each when lit.
3. **Master Clear Push Button:** pressing this button causes a Master Clear reset of the DSC and the running application.
4. **Potentiometer (R14):** the linear trim potentiometer is used to simulate analog sensor readings for the pre-programmed application. The wiper is connected to the DSC's A/D converter; the body is connected between ground and I/O port RG8. Driving RG8 high makes the potentiometer a divider between V_{DD} and ground. Driving RG8 low disables the potentiometer, and allows for reduced power consumption. R13 and C20 form a low-pass, anti-aliasing filter.
5. **CAN Interface:** this uses the Microchip MCP2561 CAN driver/receiver, which can operate up to 1 MB/s. Connections to external CAN devices are made through J2, a standard DB9 serial connector. An optional 120Ω bus termination resistor is controlled via jumper J7.
6. **LIN Interface:** this uses the Microchip MCP2021A LIN controller (U10), and is fully compliant with the LIN 2.x/SAE J2602-2 specifications. Screw terminal connections are provided for battery power in Master mode (J4) and data bus connections in Master and Slave modes (J8). Jumper J1 is used to select the operating mode.
7. **SENT Interface:** screw terminals (J9) are provided for data connections. The bus pull-up is controlled with jumper JP2.
8. **ICSP Header (J10):** space is provided for a standard 6-pin In-Circuit Serial Programming™ connector (FCI 68016-106HLF) for the dsPIC33EV256GM106 DSC. This allows for in-circuit emulation and debugging using Microchip's MPLAB REAL ICE™ in-circuit emulator, as well as direct programming of the DSC.
9. **Power LED (D5)**: this LED is lit when USB bus power is available to the USB connector.
10. **USB Port (J6):** the mini-B port provides programming/debugging connectivity and power to the Starter Kit. Bus power (+5 VDC) is provided to the DSC side of the Starter Kit through a noise filter network (L1/C47).

Note: The CAN-LIN Starter Kit can only be powered through the USB port. No other provisions to supply power to the board are available.

11. **PICkit 3 On Board (PKOB):** the PIC24FJ256GB106 microcontroller provides a simple programming interface between the dsPIC33EV256GM106 DSC and MPLAB IDE software for programming and debugging. A 3.3V regulator (U8) provides power to the circuit. Since the DSC operates at 5V, level translators (U2 and U3) are used for the internal serial clock programming signals. Space for a legacy 6-pin PICkit programmer expansion header (J5) to program the PKOB microcontroller is also provided.

12. **Current Measurement Shunt (JP1):** the operating current of the DSC can be measured here by removing the jumper and placing an ammeter across the pins. Running the demo at 40 MIPS results in current consumption of approximately 34 mA (from the +5VDC rail).
13. **Temperature Sensor (U7):** the MCP9701A analog temperature sensor measures the board's temperature; its analog output is connected to AN18 of the 12-bit ADC. The sensor's scale is 19.5 mV/°C nominal. The ADC uses V_{DD} (5V) as its reference, so each bit represents a 12.2 mV step. The circuit can resolve ±1°C.
14. **Breakout Connector (J3):** many of the DSC's I/O pins are accessible through this connector footprint. The user can solder in a male or female 2x25 pin header (2.54mm pin spacing). Individual wires can be soldered into the connector holes. Table 1-1 lists the expansion header's connections to the DSC.

TABLE 1-1: EXPANSION CONNECTOR PINOUT

J3 Pin	Function	Device Pin	J3 Pin	Function	Device Pin
1	$\overline{MCLR}$(1)	7	26	GND	—
2	GND	—	27	RB9	49
3	RE12	27	28	RC0	21
4	RG8(1)	6	29	RB10	60
5	RE13	28	30	RC1	22
6	RG9	8	31	RB11	61
7	RE14	29	32	RC2	23
8	GND	—	33	RB12	62
9	RE15	30	34	RC3	35
10	DVDD	10, 26, 38, 57	35	RB13	63
11	GND	—	36	RC4(1)	36
12	RA0	13	37	RB14	2
13	RB0	15	38	RC5(1)	37
14	RA1	14	39	RB15	3
15	RB1	16	40	RC6(1)	50
16	RA4	33	41	GND	—
17	RB2	17	42	RC7(1)	51
18	RA9	34	43	DVDD	10, 26, 38, 57
19	RB3	18	44	RC8(1)	52
20	RA10	64	45	RC11	24
21	RB4(2)	32	46	RC9(1)	55
22	RA11	12	47	RC13	47
23	RB7	46	48	RC10	45
24	RA12	11	49	GND	-
25	RB8	48	50	GND	-

Note 1: Shared I/O pin; refer to schematic for details.
2: RB4 is also used for the UART Log Data for the pre-programmed application.

1.4 INSTALLING DEVICE DRIVERS FOR THE STARTER KIT

The proper USB drivers for the Starter Kit are included in the MPLAB X IDE installation (Windows, iOS or Linux). The first time the Starter Kit is attached to the IDE, a notification window may appear that USB drivers are being installed.

NOTES:

Chapter 2. The Demonstration Application

The Starter Kit board is pre-programmed with a small application to demonstrate its functionality in its three serial automotive interfaces (CAN, LIN and SENT). The application can either operate as a Transmitter (Master) or as a Receiver (Slave), but not both at once.

The application's mode is set at power-up, or when a Master Clear event occurs, as follows:

- if none of the push buttons are depressed during power-up or while the $\overline{\text{MCLR}}$ push button is pressed, the three LEDs will light in sequence 1-2-3, and turn off in reverse sequence (3-2-1). The board is now in Transmit/Master mode.
- if *any* of the push buttons are depressed during power-up or while the $\overline{\text{MCLR}}$ push button is pressed, all three LEDs will blink together five times, and then remain lit until the push button(s) are released. This places the board in Receive/Slave mode.

2.1 TRANSMIT/MASTER MODE

When the application is running in Transmit mode, it executes an endless one-second loop as the main routine, which performs the following:

1. The state of the push buttons is read, and the corresponding LED is lit if the button is pressed.
2. The temperature sensor is sampled, and its analog output converted.
3. RG8 is driven high (+5V), and the voltage on the potentiometer is read.
4. The results of (1), (2) and (3) are formatted and transmitted on each of the three serial interfaces, starting with the CAN interface.
5. The UART monitor sends a formatted ASCII text message from pin RB4 (see **Section 2.3 "UART Monitoring Log"** for details).
6. The application waits until the timer tick ends, and then repeats the process.

While the LIN and SENT interfaces do not need to be connected in Transmit mode, an external active CAN bus or CAN analyzer must be connected to the CAN port; this is due to the CAN requirement of receiving an external ACK signal before transmitting a message. Unless the CAN portion of the application is disabled, the application will pause and wait indefinitely without an external connection, and no LIN or SENT data will be transmitted. See **Section 3.1 "Removing CAN from the Demo Application"** for information on reconfiguring the application.

2.1.1 SENT Data Transmission Formatting

The SENT message consists of the following:

- A Synchronization/Calibration period (pulse) of 56 tick times
- A Status nibble of 12 to 27 tick times. This is user defined
- Up to six data nibbles of 12 to 27 tick times. The demo uses 6 nibbles (3 bytes)
- A CRC nibble of 12 to 27 tick times

The tick time is set by a `#define` statement at 50 µs, with an allowable range of 3 µs to 90 µs.

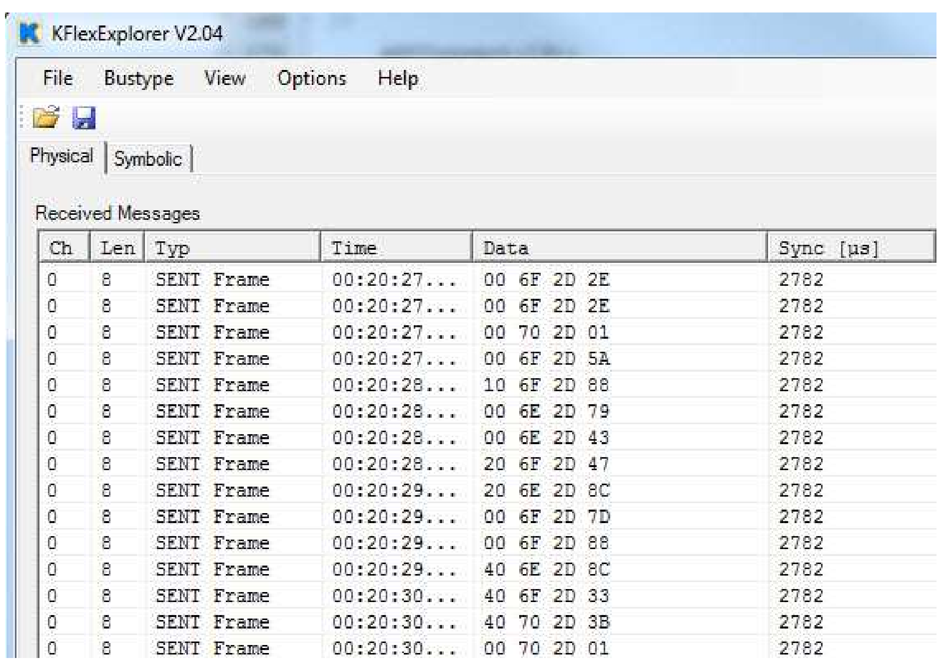
The SENT protocol uses nibbles, not bytes or words, to format the message. In the demo application, the Status nibble encodes the state of the user-defined push buttons. The data nibbles encode the potentiometer and temperature sensor readings. The Status and data nibbles are:

- Status: `0 S3 S2 S1`, where `S1` through `S3` represents the status of the push buttons (a '`1`' indicates that corresponding switch is pressed). The value of this nibble reflects which button (if any) is pressed
- Nibbles 1 through 3: 12 bits of the ADC conversion value for the potentiometer reading (MSbit to LSbit)
- Nibbles 4 through 6: 12 bits of the ADC conversion value for the temperature sensor (MSbit to LSbit)

The CRC is generated by the SENT module in hardware, and is not calculated by the application.

Figure 2-1 shows a typical screen capture of SENT data transmitted by the application. This data was captured using the KOPF Automotive Interface 4 analyzer and KFlexExplorer software (KOPF GmbH).

FIGURE 2-1: SENT MESSAGES FROM THE STARTER KIT



Ch	Len	Typ	Time	Data	Sync [µs]
0	8	SENT Frame	00:20:27...	00 6F 2D 2E	2782
0	8	SENT Frame	00:20:27...	00 6F 2D 2E	2782
0	8	SENT Frame	00:20:27...	00 70 2D 01	2782
0	8	SENT Frame	00:20:27...	00 6F 2D 5A	2782
0	8	SENT Frame	00:20:28...	10 6F 2D 88	2782
0	8	SENT Frame	00:20:28...	00 6E 2D 79	2782
0	8	SENT Frame	00:20:28...	00 6E 2D 43	2782
0	8	SENT Frame	00:20:28...	20 6F 2D 47	2782
0	8	SENT Frame	00:20:29...	20 6E 2D 8C	2782
0	8	SENT Frame	00:20:29...	00 6F 2D 7D	2782
0	8	SENT Frame	00:20:29...	00 6F 2D 88	2782
0	8	SENT Frame	00:20:29...	40 6E 2D 8C	2782
0	8	SENT Frame	00:20:30...	40 6F 2D 33	2782
0	8	SENT Frame	00:20:30...	40 70 2D 3B	2782
0	8	SENT Frame	00:20:30...	00 70 2D 01	2782

2.1.1.1 CONNECTING HARDWARE FOR SENT DATA TRANSMISSION

Connection to another SENT device is by two wires, SIGNAL and GROUND. The connections are made via J9.

The SENT bus requires a pull-up resistor, which is enabled by installing Jumper JP2. If two Starter Kit boards are connected, enable the pull-up on only one of the boards; remove the JP2 jumper on the other board. Similarly, if the Starter Kit is connected to an existing SENT network which has a pull-up installed, remove the JP2 jumper.

> **Note:** The rise-time control filter on the Starter Kit is not guaranteed to meet the requirements of the SENT standard in all possible applications. Refer to SAE J2716 for filter design guidelines.

2.1.2 LIN Data Transmission Formatting

The various LIN timing parameters are set in the demo application in a series of `#define` statements (Example 2-1). Connection to existing LIN networks or peripherals may require adjusting these parameters. The constant FCAN is the MIPS running frequency in Hz.

EXAMPLE 2-1: LIN TIMING PARAMETERS

```
#define         FCAN     40000000
#define         LIN_BAUD    4800
#define         LIN_BIT_TIME  ((1000000/LIN_BAUD) + 0.5)
#define         LIN_BRGVAL   ((FCAN/LIN_BAUD)/16) - 1
#define         LIN_ID      0x23
#define         LIN_BIT_STUFF    0x4
```

The data is transmitted as follows:

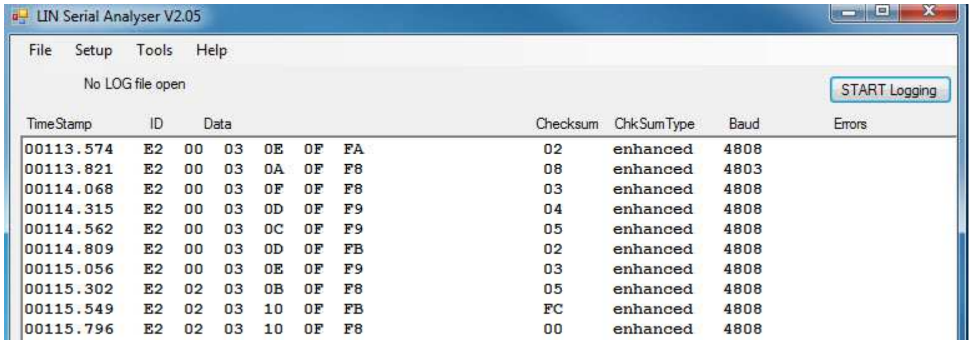
1. The ID byte, which includes two parity bits. The ID is set as a `#define` in the demo program, and may be changed to any valid byte. The application uses 0x23 by default, which is transmitted with parity as 0xE2.
2. The second byte is the data for the three push button switches. S1 represents the LSbit (bit 0), S2 is bit 1 and S3 is bit 2. A ‘`1`’ in a bit position shows the corresponding switch is depressed. Note that the switches are not debounced by the application, and the switch states are sampled once every second; therefore, there may be a delay to register a key press in the serial data.
3. The next two bytes are the raw temperature readings from the ADC. At room temperature, expected data values are from 0x02D0 to 0x0320. The application takes four readings and then averages them to send the final result.
4. The next two bytes are the potentiometer value from the ADC. The range is approximately 0x000 to 0x0FFF (a small offset value is possible).
5. The final byte is the checksum, computed per LIN specification. Note that this is an enhanced checksum calculation, which includes the ID byte in the calculation.

Figure 2-2 shows the LIN data output from the application, captured by using the Microchip LIN Serial Adapter (Microchip part # APGDT001) and associated software. This shows the output of the LIN port running the demo software.

To set this in the LIN analyzer application, open the *Setup* menu and change the settings at the Setup dialog to:

- Baud Rate: 4800
- Timeout (ms): 1000
- COM Port: “APG USB to LIN 0”

FIGURE 2-2: LIN MESSAGES FROM THE STARTER KIT



2.1.2.1 CONNECTING HARDWARE FOR LIN DATA TRANSMISSION

The LIN bus electrical signaling is controlled by a MCP2021A LIN transceiver (U10). The standard LIN bus uses three wires: SIGNAL, GROUND and POWER (+12VDC). The transceiver requires an external +12 VDC supply to be connected. The connections between the LIN bus (or other LIN device) and the Starter Kit are different for Transmit and Receive modes. Table 2-1 summarized the differences.

In Transmit (Master) mode, the +12VDC is applied to the BAT terminal of J4, and the battery ground is connected to the GND terminal. The LIN signal is applied to the SIG terminal of J8. The SIG terminal is for the LIN signal wire, and the GND terminal is used for the LIN ground. In addition, jumper J1 is set to position 1-2 for Transmit mode.

In Receive (Slave mode), J4 is not used. The +12 VDC is supplied over the bus POWER wire, which is connected to the + terminal of J8. The SIG terminal is connected to the LIN SIGNAL wire, and GND is for the GROUND wire. Jumper J1 is set to position 2-3 for Receive mode.

TABLE 2-1: LIN CONNECTIONS FOR TRANSMIT AND RECEIVE MODES

Connector/Jumper		Transmit	Receive
J1	Master/Slave	1-2 (Master)	2-3 (Slave)
J4	BAT	POWER	—
	GND	GROUND	—
J8	+	—	POWER
	SIG	SIGNAL	SIGNAL
	GND	GROUND	GROUND

2.1.3 CAN Data Transmission Formatting

CAN data is transmitted as follows:

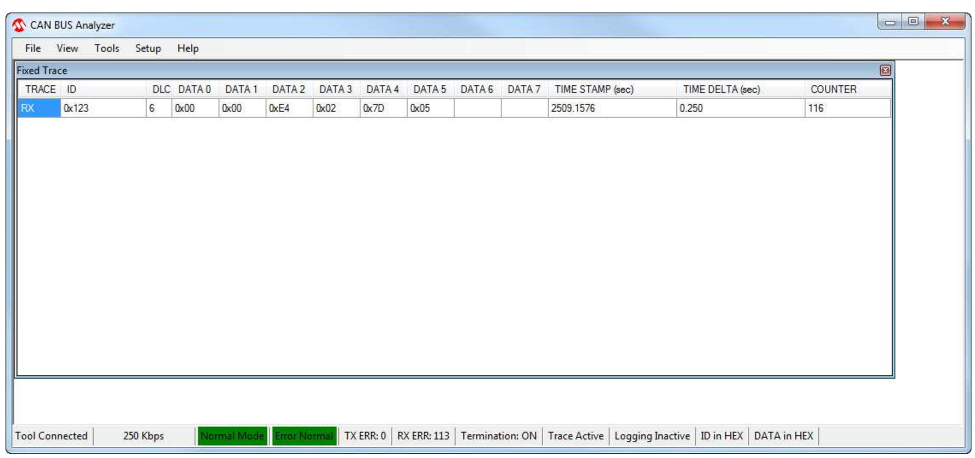
1. The SID. The demo code arbitrarily uses 0x123, which is set by the `#define MSG_SID` statement. It may be changed to any valid SID.
2. The DLC byte, which shows six bytes of data are to be transmitted.
3. The next two bytes (DATA 0 and DATA 1) are the data for the three switches. The switch data appears on DATA 0, while DATA 1 is 0x00. S1 is the LSBit (bit 0), S2 is bit 1 and S3 is bit 2. A '`1`' in a bit position shows the corresponding switch is depressed. Note that the switches are not debounced by the application, and the switch state are sampled once every second; therefore, there may be a delay to register a key press in the serial data.
4. The following two bytes (DATA 2 and DATA 3) are the raw temperature readings from the ADC. At room temperature, expected data values are from 0x02D0 to 0x0320. The application takes four readings and then averages them to send the final result. DATA3 is the Most Significant Byte.
5. The final two bytes (DATA 4 and DATA 5) are the potentiometer value from the ADC. The range is approximately 0x000 to 0x0FFF (a small offset value is possible). DATA 5 is the Most Significant Byte.

Figure 2-3 shows a typical CAN message from the demo application, as captured using the Microchip CAN Bus Analyzer (Microchip part # APGDT002) and associated software running on Windows 7. This shows a potentiometer value reading of 0x057D, and a raw temperature reading of 0x02E4. The CAN Analyzer shows that the CAN messages arrive 250 ms apart, as set by the application's main timer loop.

The CAN Analyzer must be configured to the demo application's parameters with the Setup dialog (*Setup > Hardware Setup*). The correct setting are:

- Bitrate Control: 250 kbps
- Mode Control: Normal
- Termination Control: On

FIGURE 2-3: CAN MESSAGES FROM THE STARTER KIT



2.1.3.1 CONNECTING HARDWARE FOR CAN DATA TRANSMISSION

The CAN bus should be double-terminated with 120Ω resistors at each end of the CAN cable. Jumper J7 controls cable termination across the CAN data lines. When the jumper is in the TERM position (positions 2-3), a 120Ω resistor is connected across the cable.

When connecting two CAN-LIN Starter Kit boards to each other via a DB9 female-to-female serial cable, place the J7 jumpers in the TERM position on both boards. Do not use more than two bus termination resistors.

The CAN connector (J2) follows the *de facto* industry standard for signal connection. Table 2 lists the signal pins.

TABLE 2-2: CAN CONNECTOR PINOUT (J2)

DB9 Pin	CAN Signal Name
1	NC
2	CAN_L
3	Ground
4	NC
5	NC
6	Ground
7	CAN_H
8	NC
9	NC

2.1.3.2 SPECIAL CONSIDERATIONS FOR CAN

For the demo application to properly execute, the CAN port must be connected to an active CAN bus or CAN analyzer tool that provides an Acknowledge (ACK) signal to the Starter Kit. If it is not connected, the demo application's main loop will hang, waiting for an ACK signal to be received; no LIN or SENT transmissions will be sent.

The application can be modified and re-compiled to disable CAN functionality and only run the SENT and LIN portions. See **Section 3.1 "Removing CAN from the Demo Application"** for details.

2.2 RECEIVE/SLAVE MODE

Receive/Slave mode is set by pressing and releasing the $\overline{\text{MCLR}}$ push button while simultaneously pressing and holding any of the three push button switches. The three LEDs rapidly flash in unison, then stay lit. Once they are all lit, release the held push button.

While in Receive mode, the application waits for any of the three serial interfaces to receive and process a valid message from another CAN-LIN Starter Kit running the demo application in Transmit mode, or other external source. Once processed, the LEDs will momentarily flash to indicate a valid received message, as follows:

- Valid CAN message: LED1
- Valid LIN message: LED2
- Valid SENT message: LED3

It is possible that messages sent by other sources will show a valid received message if they are the correct baud/bit rates and have the same number of data bytes transmitted as the demo application.

Messages can be received from any or all of the connected ports at once, and multiple messages will be validated.

2.3 UART MONITORING LOG

The demo application contains an independent message-logging UART in both Transmit and Receive mode. The UART uses ASCII-encoded serial data at 38,400 baud, 8-bit data, with 1 Start bit, 1 Stop bit and no parity (38400-8-1-1-N). The serial data is sent to I/O pin RB4 on the Expansion Connector (J3). In Transmit mode, the logger reports the current temperature (°C), the measured potentiometer voltage (V) and the status of each push button switch. This data is then encoded and sent over each interface (in the order of SENT, LIN, then CAN).

Figure 2-4 displays a typical logging message in Transmit mode, as viewed on a terminal application. Note that this ASCII data is not what is physically sent over the interfaces. The data sent is raw data as described in the prior sections for each interface. The data is described as local, as this is the data measured by the board about to Transmit.

FIGURE 2-4: UART TRANSMIT LOG (TERMINAL DISPLAY)