Reference Design

Buck Converter Battery Charger Using the Z8F042A MCU

RD001302-0814

Overview

This document describes a Z8 Encore! XP-based Buck Converter Battery Charger reference design that employs Zilog's Z8F042A MCU to control a step-down DC-DC converter (also known as a buck converter) that acts as a regulated power source. This buck converter battery charger hardware is capable of regulating charger output in a number of modes, such as constant voltage, constant current, or constant voltage with a current limit.

The charger can be viewed as a complete control system. The type and capacity of the battery determines the mode of operation of the battery controller – namely, a constant current source or a constant voltage source. The voltage (V_{SET}) and current (I_{SET}) setpoints are also determined by the type and capacity of the battery. All battery control loop operations can be controlled by the user via the Z8F042A MCU's UART block, and feedback is provided in the HyperTerminal console. Additionally, LEDs provide a visual status of the charging process.

Note: The source code file associated with this reference design, RD0013-SC01.zip, is available free for download from the Zilog website. This source code has been tested with version 5.0.0 of ZDS II for Z8 Encore! XP MCUs. Subsequent releases of ZDS II may require you to modify the code supplied with this reference design.

Features

The key features of this buck converter battery charger are:

- Z8F042A MCU featuring a highly accurate Sigma-Delta ADC
- Buck converter with proportional/integral controlled constant current and voltage
- Differential ADC for current measurements
- Complete programmability
- Current/voltage control independent of user input
- UART-controlled operation
- In-circuit programming for upgrading or modifying firmware

Potential Applications

This Buck Converter Battery Charger reference design can be used to develop a number of applications; the brief list below offers a few ideas.

- Security systems
- Information systems
- Warning systems

Discussion

The Buck Converter Battery Charger reference design consists of a buck converter controlled by a Z8F042A MCU, plus circuitry to provide feedback signals to the MCU.

To save memory resources, the provided UART does not implement the `stdio.h` libraries. Instead, a simple UART using only integer values is used. As a result of this implementation, the user is asked to enter values from 10 to 40, representing values from 1.0 to 4.0, respectively, for a set voltage and set current.

An external voltage source 5V or 15V, or USB can be used to supply the operating voltage.

The USB 2.0 port current limit is 500mA.

When the MCU is powered up and if a battery was detected, the MCU will start charging the battery by generating the PWM pulses to the buck converter to the initial current and voltage settings as determined in the `Userinput.h` header file, until the user enters the desired values.

The charge process is executed in the background and controlled via a timer0 interrupt service routine to alternate between current and voltage ADC channel sampling, keeping the UART as a foreground task in the main function. The PI then uses the sampled ADC information from the voltage and current samples to adjust the PWM for the battery charging process.

During the charging process, the MCU measures battery charge current as a voltage drop across R4, a 1 Ohm resistor, in series with the battery and adjusts duty cycle of PWM to the buck converter to keep this voltage drop in respect with a battery-charging curve which is determined by the battery chemistry.

When the battery charge current reaches the voltage charge threshold, the MCU decreases charging current and starts monitoring battery voltage.

While the battery is charging, the red LED is on and the green LED is off. When the charging process is completed, the red LED is turned off and the green LED is turned on and a message to the hyper-terminal (GUI) indicates battery charge is completed. The buck converter off, but the PI control loop keeps monitoring the battery voltage and refreshes the charge, if necessary. The charging process can be stopped by the user at any time.

Components and MCU Peripherals

Switching MOSFET Q1 with 3A current limit and low power MOSFET Q2 used as level shifter for Q1.

Switching MOSFET Q1 with Inductor L1, capacitors C11, C12, C13, and diode D1 forms buck converter circuitry.

Resistive dividers R6/R9 and R7/R10 provide battery voltage/current measurement points for MCU ADC circuitry.

Capacitors C14, C15 may be used as HF filters, if switching noise affects ADC conversion.

Capacitors C1 - C10 are input filter. Divider R2/R5 provides measurement point for input voltage.

LEDs D2 (red) and D3 (green) are used as indicators to reflect battery charge state and fault conditions.

Evaluation board assumes follow MCU ports assignment:

- Analog inputs (ADC):
  - PB0/ANA0 - Thermistor Voltage (Single-ended mode)
  - PB3/ANA3 - Input Voltage (Single-ended mode)
  - PC0/ANA4 - Buck Converter Output Voltage (Single-ended mode)
  - PC1/ANA5 - Battery Voltage (Single-ended mode)
  - PC0/ANA4 - PC1/ANA5 - Battery Current (Differential Mode)
- Digital outputs:
  - PA7/T1out: PWM pulses
  - PA1: Red LED D2
  - PA2: Green LED D3

The Board features two power source options. It can be powered by connecting USB (B) connector J1 to the USB port of the development PC using the USB A to USB (B) cable included in the Kit. The other option is to connect a female plug to an external 6VDC source with at least 300–400mA of current to external power supply connector J3. A drawing of an optional external power supply plug is shown in Figure 1.

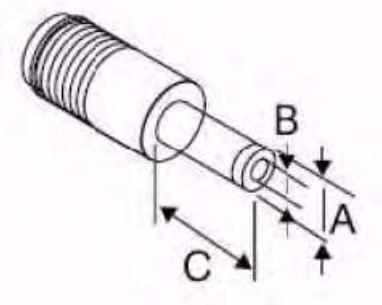

Figure 1. Female Plug

Legend
A = 5.5mm
B = 2.1mm
C = 8.8mm or longer

Figure 1. Female Plug

When the Board is powered via a USB connection, communication with the development PC can be established through the FT232RL chip (U1) to provide a USB-to-serial interface. The UART0 block of the Z8F042A MCU is connected to this chip.

An on-board USB power distribution switch (U6) provides overcurrent protection in the event of a short or if a device is connected to the Board that requires more than 500mA @3.3V. If either condition occurs, LED D4 will illuminate.

The battery must be connected to terminals Vbat+ and Vbat–.

The Battery Charger Reference Board, shown in Figure 2, features two IXYS MOSFETs. MOSFET Q1 exhibits a 3A current limit; the low-power MOSFET Q2 is used as a level shifter for Q1. The buck converter circuit consists of MOSFET Q1, inductor L1, capacitors C11, C12 and C13, and a diode. Resistor R4 is used as a battery current sense resistor, and resistive dividers R6/R9 and R7/R10 provide the battery voltage/current measurement points for the Z8F042A MCU's ADC inputs.

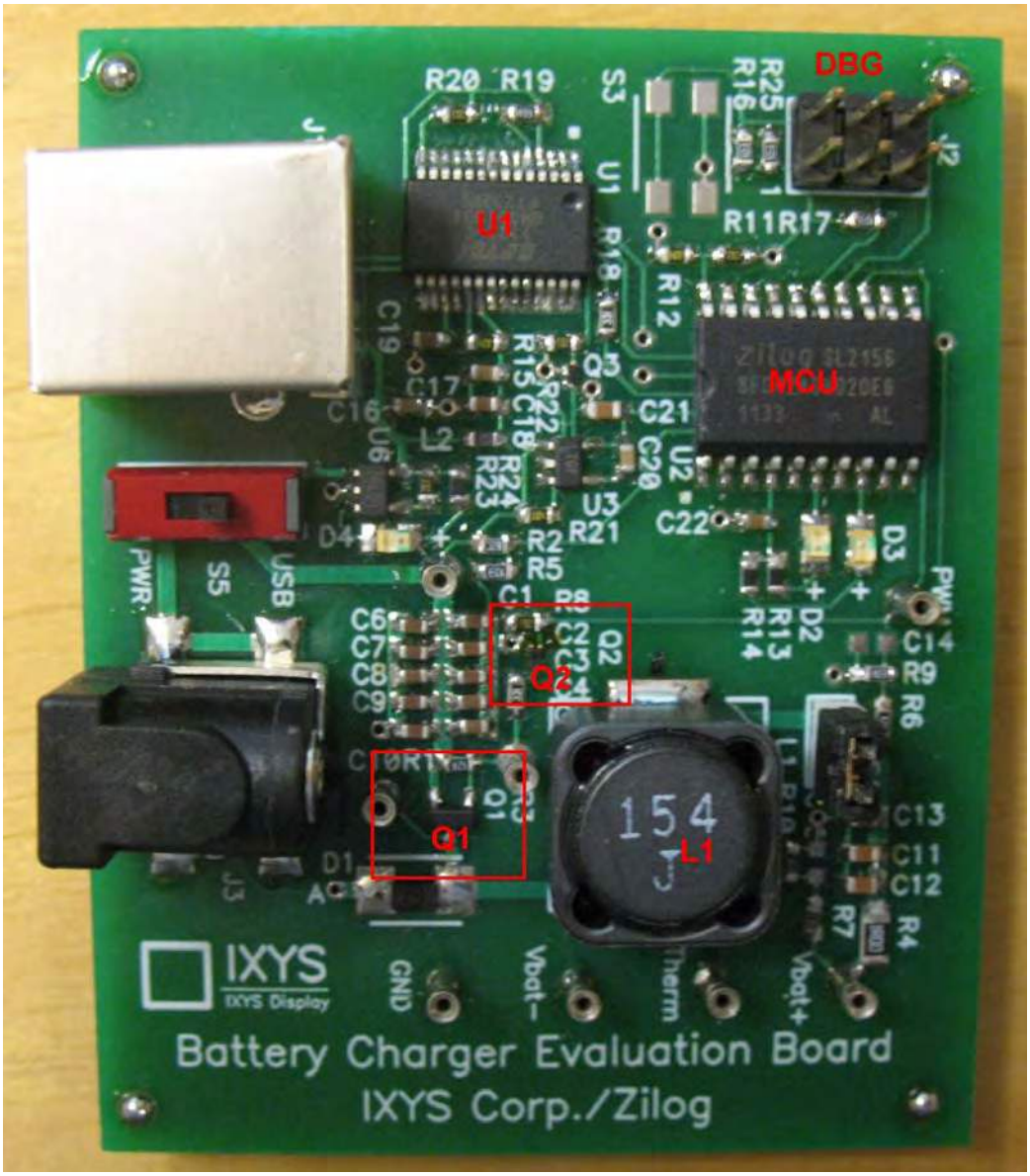

Figure 2. The Buck Converter Battery Charger Reference Design Module

Battery Charger Hardware

The battery charger hardware is built and tested with commonly available components. The main hardware component of this implementation is a Z8 Encore! XP-based F042A 20-pin microcontroller. Figure 3 displays a block diagram of the battery charger; see Appendix C. Schematic Diagrams on page 26 for a schematic illustration of the Battery Charger Reference Board.

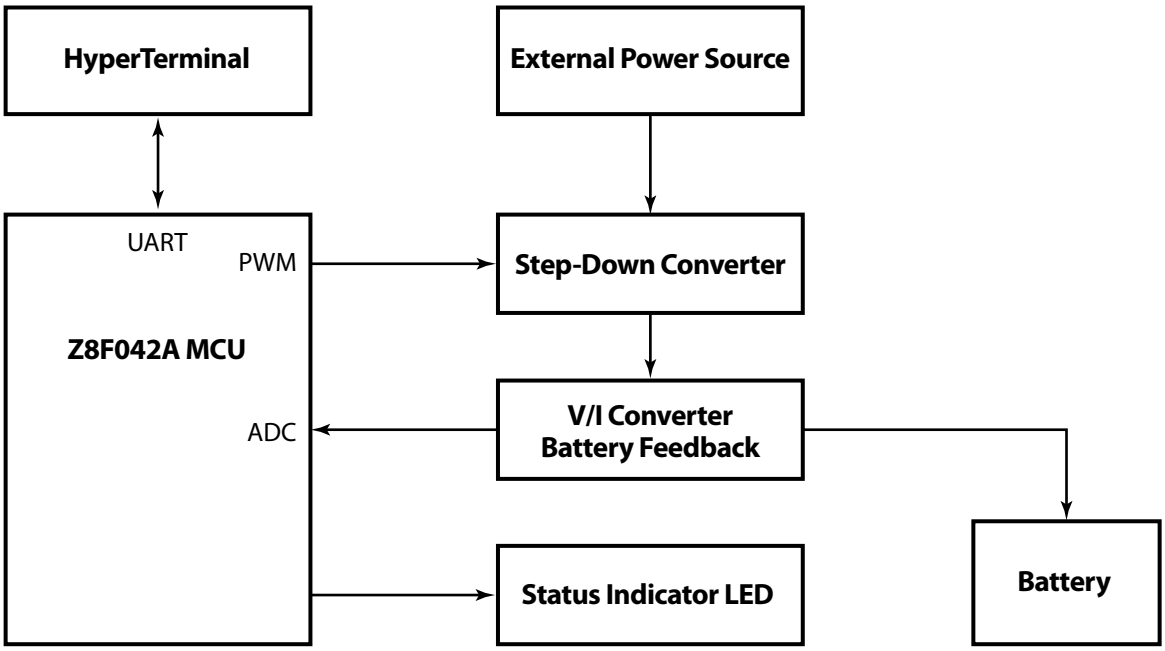

Figure 3. Buck Converter Battery Charger Block Diagram

Z8F082A Series Flash MCUs

The Z8F042A MCU used in this reference design is a member of the Z8F082A Series of Z8 Encore! XP Flash MCUs; it is a 4KB version of the Z8F082A MCU. Z8F082A Series MCUs offer the following features:

- 20MHz eZ8 CPU
- 1KB, 2KB, 4KB, or 8KB Flash memory with in-circuit programming capability
- 256B, 512B, or 1KB register RAM
- Up to 128B nonvolatile data storage (NVDS)
- Internal precision oscillator trimmed to ±1% accuracy
- External crystal oscillator, operating up to 20MHz
- Optional 8-channel, 10-bit analog-to-digital converter (ADC)
- Optional on-chip temperature sensor

- On-chip analog comparator
- Optional on-chip low-power operational amplifier (LPO)
- Full-duplex UART
- The UART baud rate generator (BRG) can be configured and used as a basic 16-bit timer
- Infrared Data Association (IrDA)-compliant infrared encoder/decoders, integrated with the UART
- Two enhanced 16-bit timers with capture, compare and PWM capability
- Watchdog Timer (WDT) with dedicated internal RC oscillator
- Up to 20 vectored interrupts
- 6 to 25 I/O pins depending upon package
- Up to thirteen 5V-tolerant input pins
- Up to 8 ports capable of direct LED drive with no current limit resistor required
- On-Chip Debugger (OCD)
- Voltage Brown-Out (VBO) protection
- Programmable low battery detection (LVD) (8-pin devices only)
- Bandgap generated precision voltage references available for the ADC, comparator,
- VBO and LVD
- Power-On Reset (POR)
- 2.7V to 3.6V operating voltage
- 8-, 20- and 28-pin packages
- 0°C to +70°C and –40°C to +105°C for operating temperature ranges

FT232RL USB-to-Serial UART Interface

The FT232RL (U1) device is a USB-to-serial UART interface between a host PC and the Z8F042A MCU with an optional clock generator output. In addition, asynchronous and synchronous bit-bang interface modes are available. USB-to-serial designs using the FT232RL device have been further simplified by fully integrating external EEPROM, clock circuit and USB resistors onto the device. Its inputs are connected to the USB connector J1, and its outputs are connected to the Z8F042A MCU's UART serial communication ports PA4 and PA5.

U3

U3 is a 3.3V voltage regulator for the Z8F042A MCU and the debug connector.

Buck Converter

In general, a buck converter consists of an inductor, two switches, a transistor and a diode to control the inductor. The PWM alternates between connecting the inductor to a source of voltage to store energy in the inductor, and discharging the inductor into the load.

In this reference design, the PWM is an input to the buck converter which is controlled by the MCU at an 80kHz variable-PWM duty cycle. Depending on the charging speed, the output of the converter usually ranges from 1V to 4V; this voltage is supplied to the battery. Both the feedback charge current and charge voltage are controlled by the PI loop.

R2 and R5 form a voltage divider on the input of the buck converter. The ratio of input voltage and battery voltage (Vbat) is used to determine the start-up duty cycle.

Differential Voltage Divider Circuit

The Differential Voltage Divider Circuit/Feedback Module provides battery voltage, battery current, and converter input using three ADC inputs. These feedback readings are used for constant current charging/voltage charging with a PI control loop, and to terminate the charging process. The constant current charging is applied using a differential resistor network. During the constant current charge, the differential voltage drop across a sense resistor is held constant, therefore the battery charge current is constant. The battery charging process can be monitored via UART and the GUI (HyperTerminal).

Temperature Circuit

A temperature (thermistor) circuit consists of resistors R11 and R12. If the battery features a temperature sensor, then this part of the circuit monitors the voltage drop across voltage divider R11, R12. The thermistor is parallel t

o R12; its voltage drop determines the voltage drop across R12. As a result, the software acts upon this voltage drop threshold.

LED Indicators

Three different LEDs display the various functionalities of the charger:

- Overcurrent indication LED D4
- Battery charging indication LED D2
- Charging complete indication LED D3

The red LED (D2) indicates when a battery charge is in progress; it illuminates the green LED (D3) when the charging process is complete. The Reference Board uses a 4.5V to 15V DC regulated external input power supply to drive the 5V and 3.3V internal supply.

Battery Charger Software

The software provided with this reference design is developed and tested on the hardware per the schematic diagram shown in Figure 12 on page 26. This section describes the functionality of this reference design's different software blocks used for battery charging.

For additional details about this software, refer to the demo firmware contained in the RD0013-SC01.zip file, which includes the following files:

- Main.c
  - Variable declarations
  - Initialization of peripherals
  - UART control
- Control.c
  - Timer0 interrupt service routine (control loop for constant current/voltage charge)
  - PWM duty for battery charging
  - Constant current PI feedback loop
  - Constant voltage PI feedback loop
  - UART transmit function
  - UART receive function
- Initialization.c
  - WDT initialization
  - Input/Output pin initialization
  - UART initialization
  - ADC initialization
  - Timer0 initialization
  - PWM initialization
- UserInput.h
  - User input parameters for the PI control loop
- Main.h
  - Function body declarations
  - External variable declarations
  - Constant definitions
  - Structure declarations

Battery Charger Algorithm

The battery charging process is a PWM PI-controlled, closed-loop algorithm based on battery current and voltage feedback. The input PWM frequency of the buck converter is 80kHz. If the battery is not completely charged, then the duty cycle required for maintaining the setpoints at the converter outputs is calculated by the control algorithm. The control algorithm implements a *Proportional, Integral* (PI) loop to derive the necessary PWM duty cycle based on the following equation:

$$u(t) = k1 * e(t) + k2 * \int e(t)dt$$

In this equation, K1 is the proportional constant coefficient, K2 is the integral constant coefficient, and e(t) is the error between the set value and the actual value as a function of time.

The coefficients are determined empirically based on system response because they are contingent upon reactive/complex load conditions, which may vary. These coefficients should be chosen such that smooth convergence to a current setpoint and a voltage setpoint is achieved.

If UART is enabled, then the current setpoint and voltage setpoint can be entered in HyperTerminal. If UART is not selected, then these voltage and current values must be entered in the `UserInput.c` file.

The interrupt service routine (ISR) timer is invoked every 100µs (at 10kHz). The PWM value computed by the control algorithm is loaded into the PWM generators to be transmitted via the output pin. The 16-bit timer's PWM Mode offers a programmable switching frequency based on the reload value; this flexibility allows a trade-off between accuracy and the frequency of the PWM switching signal. The higher the frequency, the lower the PWM Register reload value and resolution of the PWM; the converse is also true.

The firmware heuristic is designed such that only UART code is executed in the main infinite loop. Timer0 is the main control engine, configured for interrupts at 10kHz intervals to change the ADC channels, if the ADC data is ready, and to determine whether the current control or the voltage control PI loop is required.

Battery Safety and Charge Termination

For safe operation, termination threshold calculations must be based on battery parameters. The setpoints for the DC-DC step down (buck) converter voltage, the current, and the current limit are calculated. When the one-time calculations are complete, the charger software enters into an infinite loop which is terminated by a successful charge completion or safety error.

Inside the Timer0 loop, the ADC reads the actual values of the converter output voltage, the battery voltage, and the current. The ADC measures the output voltage/current output of the buck converter as feedback to the controller.

The ADC performs a basic measurement of output voltage, output current, and battery voltage at the battery's terminals as an input to determine charge termination. To make this determination, the current across the battery terminals must be the same as the measured converter output current.

This safety routine is responsible for the overall safety features associated with a battery charger. The charger ensures safety by comparing the actual converter voltage and the battery voltage with calculated thresholds. Crossing these thresholds switches the PWM output off, which turns off the converter output and terminates the charging functions. Such termination protects the batteries in case of a device failure. If all of the actual values are within limits, the battery is tested for full charge.

Initializing the Z8 Encore! XP Peripherals

All Z8Encore! peripherals are initialized from their power-on states to their required modes of operation via the following ports:

- Port A, bits 1 and 2 are used for the status LEDs
- Port A, bits 4 and 5 are the UART transmit and receive pins to communicate with HyperTerminal
- Port B, bit 3 is used for single-ended input voltage mode
- Port C, bit 0 and 1 are the battery charge voltage pins
- PA 7 is configured to generate the 80kHz PWM frequency for the buck converter

Kit Contents

The tools used to build this reference design are:

- Buck Converter Battery Charger Design Module
- USB cable, Type A to B
- Buck Converter Battery Charger Reference Design Flyer (FL0144)

Required Items Not Supplied

- USB Smart Cable (Zilog PN: ZUSBSC0001ZACG)
- 3.7V lithium-ion battery

Software Installation

This project requires the ZDSII – Z8Encore! Integrated Development Environment. Observe the following procedure to download and install the ZDSII software.

1. Download the latest version of ZDSII for Z8 Encore! from the Downloadable Software category in the Zilog Store.
2. Run the software installation file and follow the on-screen instructions to install ZDSII.

Viewing and Rebuilding the Battery Charger Software

The source code/project is provided in this kit so that users can be familiar with the Buck Converter Battery Charger reference design. However, users are not expected to modify or change any parameters.

1. If you have not downloaded the software for this reference design, refer to the paper insert that was included in your kit (Document Control Number FL0144), and follow the 3-step software download instructions contained on it.

2. Launch the ZDSII – Z8 Encore! application by navigating via the following default path:

   Start → Programs → Zilog ZDS II_Z8Encore!_<version_number> → ZDSII_Z8Encore!<version_number>

3. From the **File** menu, select **Open Project**. The Open dialog box appears.
4. Browse to the `src` folder which, by default, is located in the following path:

   `<ZDS Install>\ZRD0013CHRGZRD_<version>`

5. Select the `Applications.zdsproj` file from the `src` folder, and click **Open** to display the initial ZDSII program screen. To view the source files, double-click the **Project Files** folder on the left side of the IDE interface. Double-click an individual file to open the file in the ZDSII file editor.
6. Click the **Rebuild All** toolbar icon, which is highlighted in red in Figure 4.

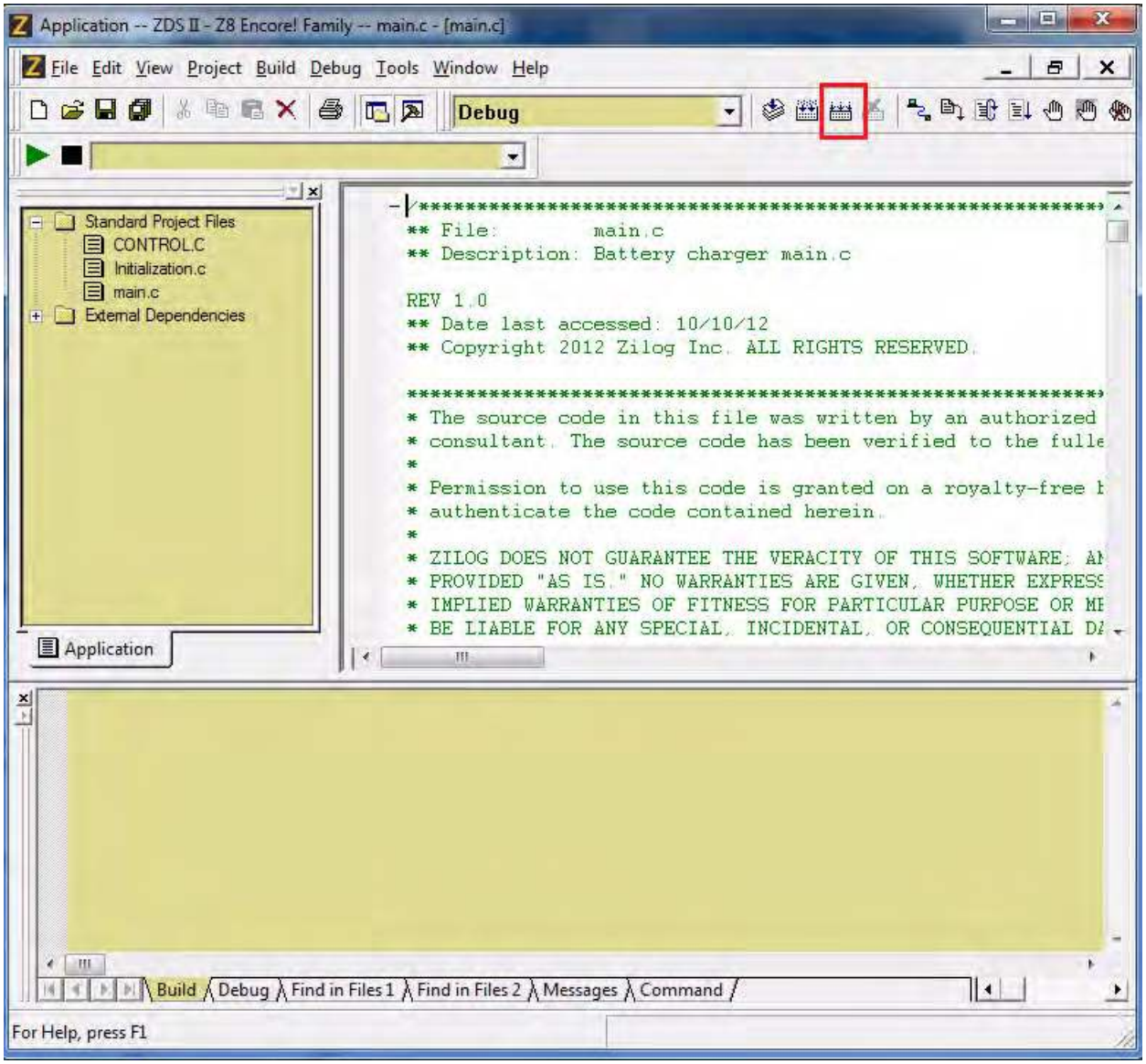

Figure 4. Build Mode Configuration

7. When the rebuild is complete, a `Build succeeded.` message will appear, as highlighted in Figure 5.

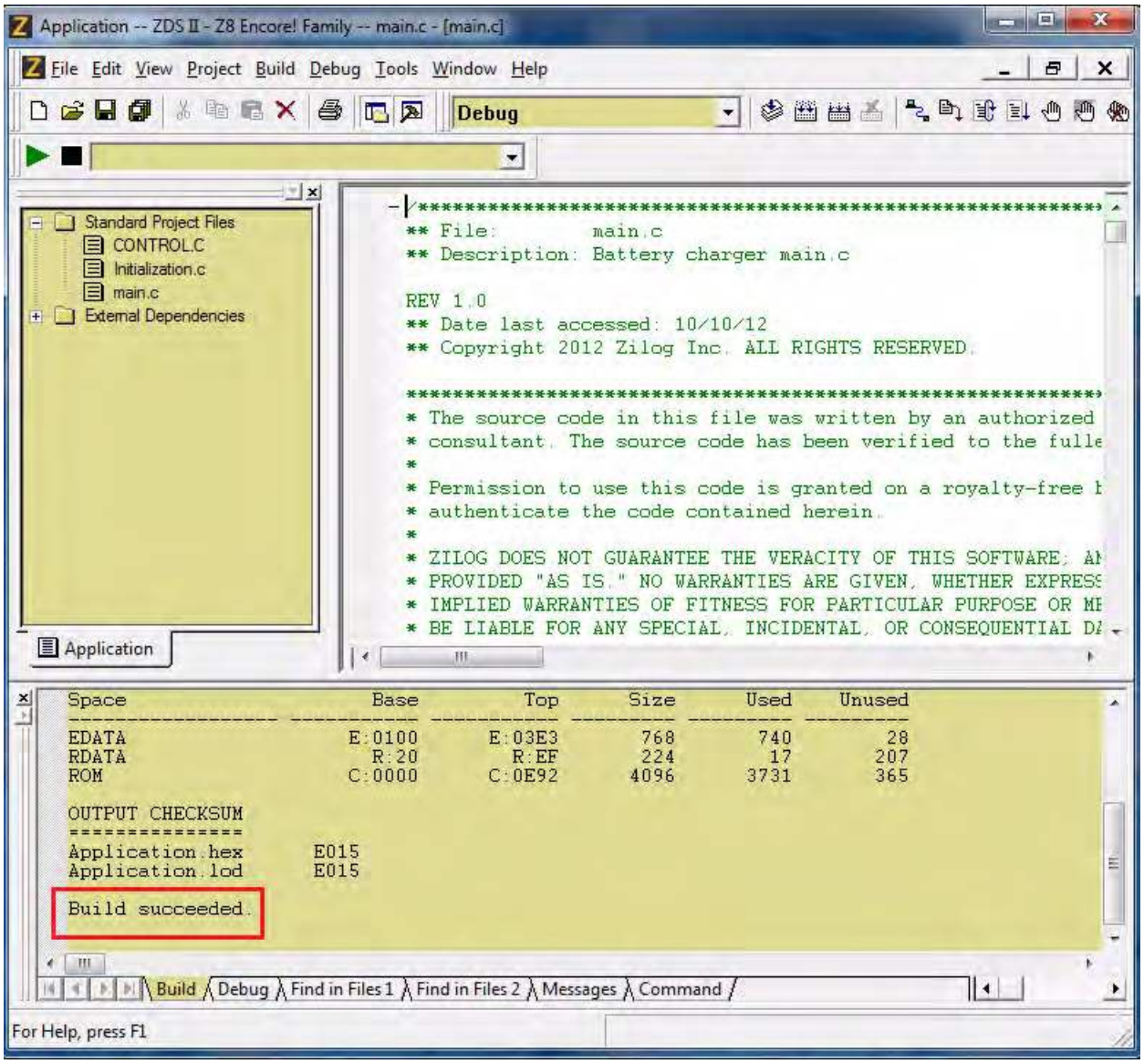

Figure 5. A Successful Build

Note: The output checksum shown in Figure 5 is for reference only and may not display the same results at the time of release.

Running the Demo

Observe the following procedure to set up your reference design and host PC to run the demo/application software.

1. Connect a 3.7 V Lithium Ion battery to the Battery Charger Reference Module across Vbat– and Vbat+ terminals.
2. Connect a USB Smart Cable to J2 of the Reference Module and host PC USB port. See Appendix A. Installing the USB Smart Cable Driver on page 21 for assistance, if required.
3. Ensure that the S1 shunt is ON, and slide switch S5 to the USB position. Figure 6 indicates the locations of S1 and S5 on the Module.

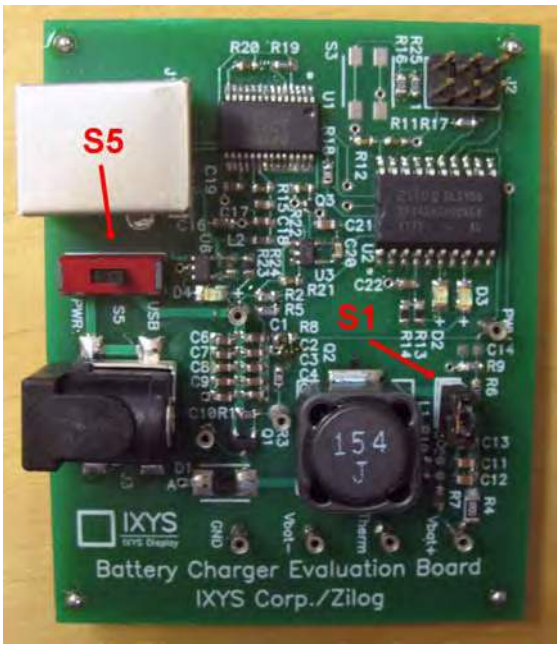

Figure 6. Shunt and Switch Locations on the Battery Charger Reference Module

4. Connect the included USB Cable Type A to B to J1 and to the host PC's USB port. If necessary, see Appendix B. Installing the FTDI USB-to-UART Driver on page 24.
5. In your terminal emulation application, select the COM port assigned to the USB-to-serial cable, and modify the settings to reflect an 57600, 8, N, 1, N configuration, as shown in the Tera Term example in Figure 7.

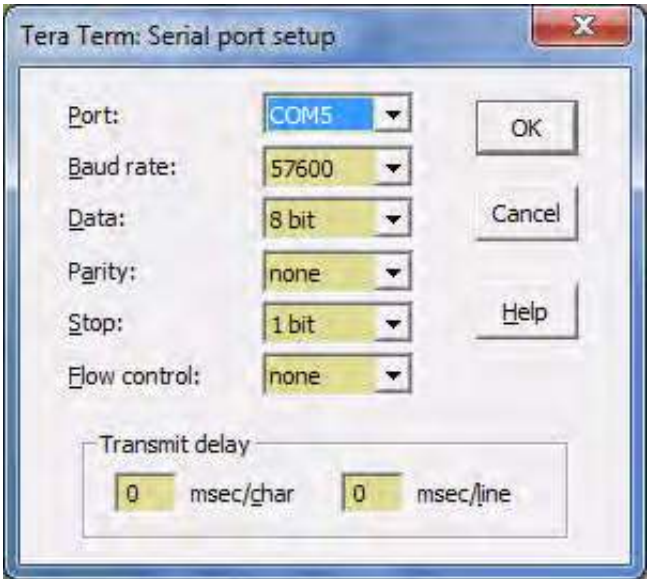

Figure 7. Configuring the Serial Communication Settings

6. Launch ZDSII for Z8 Encore!. If you have not yet installed ZDSII, return to the Software Installation section on page 11.
7. From the **File** menu in ZDSII, select **Open Project**, and navigate to the following file-path:

   `<ZRD0013CHRGZRD_<version>\src`

8. Select the `Applications.zdsproj` project and click **Open**. A list of source files will appear in the Workspace panel.
9. From the **Project** menu in ZDSII, select **Settings** to open the Project Settings dialog box. In this dialog, click the **Debugger** tab.
10. Select **ZRD0013CHRGZRD** from the **Target** list, then select **USB Smart Cable** from the **Debug Tool** drop-down menu.
11. Click **OK** to close the Project Settings dialog box.
12. If you are prompted to rebuild any affected files, click **Yes**. Otherwise, choose **Build** from the menu bar, then click **Rebuild All**. 10. To run the application, select **Go** from the **Debug** menu.
13. After the application has started, console output should be visible in the terminal emulation program.
    a. If there is no battery, or if the battery is less than the default NO_BATTERY value of 0.25V, the RED LED will be ON, and the following message will be displayed.

    ```
    Please mount battery and restart
    ```

b. If the battery voltage level is above the default SET_VOLTAGE level of 1.3 V, the GREEN LED will be ON, and the *Charging complete* message shown in Figure 8 will be displayed.

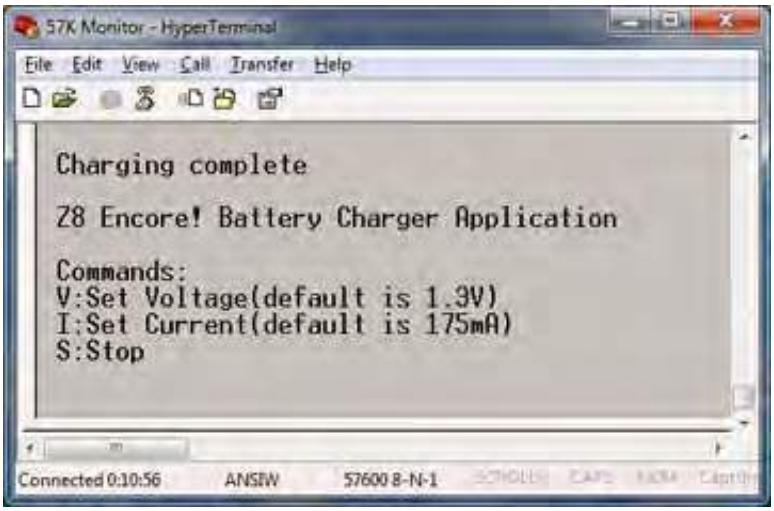

Figure 8. "Charging is Complete" Message

c. If the battery voltage is above the NO_BATTERY level of 0.25 V and below the SET_VOLTAGE level of 1.3 V, both the RED and GREEN LEDs will be ON, and the *Charging* message shown in Figure 9 will be displayed.

Figure 9. Charging in Progress Message

14. You can change the SET_VOLTAGE and SET_CURRENT levels using the `V` and `I` commands. Use the `S` command to stop the charging operation.

Results

Figure 10 displays the result for the Z8 Encore! XP-based Buck Converter Battery Charger. The blue and yellow line shows the differential voltage of the current sensor resistor which, in effect, shows the constant current charging process.

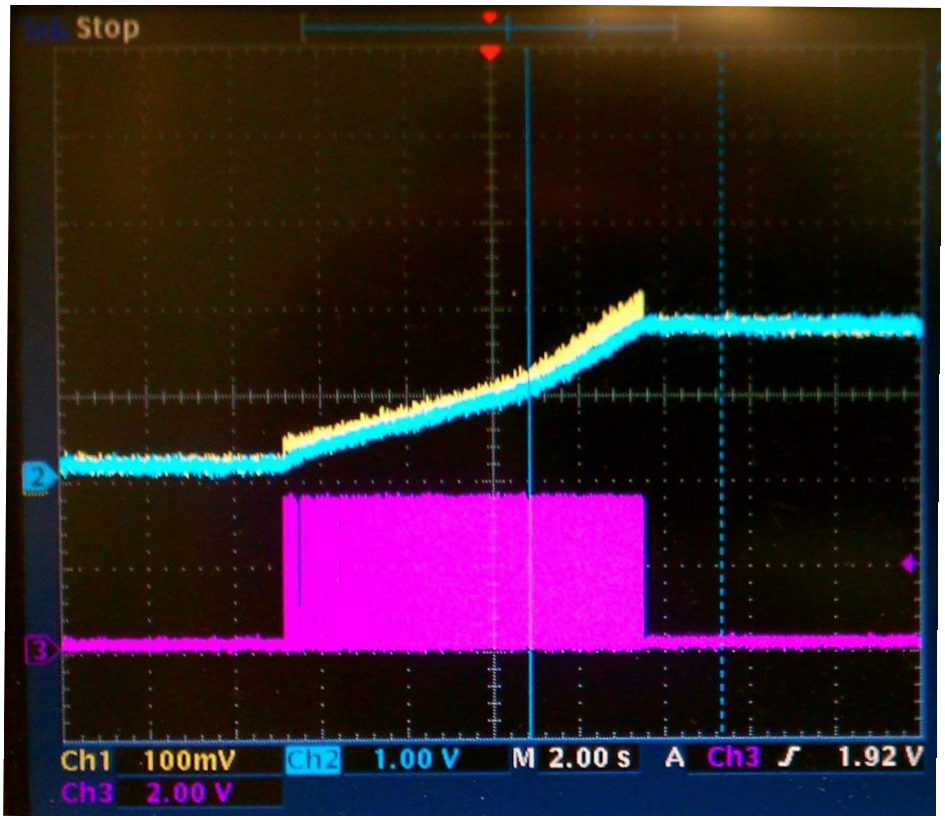

Figure 10. Current and Voltage Charge with Respect to PWM

Electrical Specifications

Table 1 lists the electrical characteristics of the battery charger hardware and reflects all available data as a result of testing prior to qualification and characterization. As such, the data presented in Table 1 are subject to change.

Table 1. Electrical Specifications for the Buck Converter/Battery Charger

Parameter	Min	Max	Units	Notes
V+ (power)	4.5	15	Volts	
Max voltage range, all other pins	–0.3	5.5	Volts	I/O pins and Reset.
Max current for I/O pin connection points	–25	25	Milliamps (mA)	

Table 1. Electrical Specifications for the Buck Converter/Battery Charger (Continued)

Parameter	Min	Max	Units	Notes
I V+		1	Amps	
Ambient operating temperature	–20	50	Degrees Celsius	
Storage temperature	–50	125	Degrees Celsius	

Note: Stresses greater than those listed in Table 1 may cause permanent damage to the battery charger hardware. These ratings are stress ratings only. Operation of this hardware at any condition outside those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods affects device reliability. For improved reliability, unused inputs must be tied to one of the supply voltages (V_{DD} or V_{SS}).

Ordering Information

The Buck Converter Battery Charger Reference Design be purchased from the Zilog Store – simply click the Store Product ID listed in Table 2.

Table 2. Ordering Information for the Buck Converter Battery Charger Reference Design

Part Number	Description	Store Product ID
ZRD0013CHRGZRD	Buck Converter Battery Charger Reference Design	RD10015

Summary

This low-cost reference design demonstrates a Lithium Ion battery charger consisting of a Z8F042A MCU and a buck converter. The charging process utilizes the highly accurate Sigma-Delta ADC peripheral and alternates between current and voltage monitoring, which is controlled in the background with the timer0 interrupt service routine to allow for the UART to be processed in the main function. With the provided hyper terminal GUI, the user can enter desired set-voltages and set-currents .

A proportional/integral (PI) control loop is used to charge the battery and to monitor the battery voltage after the charging process is completed.

To save memory resources, the here provided UART does not implement the STDIO.H libraries. Instead, a simple UART using only integer values is used. This battery charger reference design allows to be operated either via UART or an external 5-15V power supply.

The advantage of this battery charger implementation is in the efficient utilization of the MCU resources, allowing for low cost designs.

References

The documents associated with this reference design are listed in Table 3. Each of these documents can be obtained from the Zilog website by clicking the link associated with its Document Number.

Table 3. Buck Converter Battery Charger Reference Design Documentation

Document Number	Title
RD0013	This Buck Converter Battery Charger Reference Design document
RD0013-SC01	Source code for this reference design
PS0228	Z8 Encore! XP F082A Series Product Specification

Appendix A. Installing the USB Smart Cable Driver

The USB Smart Cable can be installed on PCs that run on Windows 7 (32- and 64-bit), Windows Vista (32- and 64-bit) and Windows XP operating systems. The procedures in this appendix will guide you through the USB Smart Cable installation process.

Windows 7 32/64 Systems

Observe the following steps to install the USB Smart Cable on a Windows 7 system.

1. Connect the USB Smart Cable to a USB port on your development PC. When the PC detects the new hardware, it will display the Installing device driver software dialog.
2. Windows automatically searches for the driver; this process can take a few moments. Because there is no option to terminate this search process, wait for the search to complete.

   If the driver was previously installed, Windows will automatically install the USB Smart Cable driver. If this is the case, skip ahead to Step 9. If Windows cannot find the driver, close the search dialog and proceed to the next step.
3. From the Start menu, navigate via the **Search Programs and files** menu, and enter `Device Manager` in the **Search** field to cause the Device Manager to appear in a list of search results.
4. From this search list, click **Device Manager** to open the Device Manager dialog, which presents a list of devices that operate on your PC. Find **Other devices**, toggle it to view a sublist of additional devices, and right-click your mouse on **USB Smart Cable**.
5. In the submenu that appears, click **Update Driver Software...**.
6. In the **Update Driver Software – USB Smart Cable** dialog that appears, click the **Browse my computer for driver Software** option.
7. Click the **Browse...** button to browse to one of the following driver directories, depending on the configuration of your PC.

   On 32-bit Windows 7 systems, navigate to:

   ```
   <ZDS II Installation Directory>\device drivers\USB\x32
   ```

   On 64-bit Windows 7 systems, navigate to:

   ```
   <ZDS II Installation Directory>\device drivers\USB\x64
   ```
8. Click **Next** to install the driver. On 32-bit: Windows systems, a security dialog will appear; select **Install this driver software anyway**.
9. Click **Close** after the Wizard finishes the installation.

Windows Vista 32/64 Systems

Observe the following steps to install the USB Smart Cable on a Windows Vista system.

1. Connect the USB Smart Cable to a USB port on the development PC.
2. After the PC detects the new hardware, it will display the Found New Hardware Wizard dialog box. Click **Locate and install driver software (recommended)**.
3. Depending on your development PC's User Account Control settings, Windows may ask for permission to continue the installation. Click **Continue**.
4. When the Insert the Disc dialog appears, select **I don't have the disc. Show me other options**. Click the **Next** button to display the Windows couldn't find driver dialog.
5. Select **Browse my computer for driver software (advanced)** to display the Browse For Driver dialog, which prompts you to key in or browse for the location of the driver's `.inf` file. Depending on the type of computer you use (32- bit or 64-bit), use the **Browse...** button to navigate to one of the following paths, then click the **Next** button.

   On 32-bit Vista systems, navigate to:

   ```
   <ZDSII Installation>\device drivers\USB\x32
   ```

   On 64-bit Vista systems, navigate to:

   ```
   <ZDSII Installation>\device drivers\USB\x64
   ```

6. When the Windows Security dialog prompts you whether to install or not install, click **Install this driver software anyway** and wait until the installation is completed (Windows may prompt you more than once).
7. When the software has been installed successfully, click **Close**.

Windows XP Systems

Observe the following steps to install the USB Smart Cable on a Windows XP system.

1. Connect the USB Smart Cable to a USB port on the development PC. When the PC detects the new hardware, it will display the Found New Hardware Wizard dialog.
2. In the Wizard, select **Install from a list or specific location (Advanced)**, then click **Next**.

Note: If the Windows Hardware Installation dialog appears, click **Continue Anyway**.

3. In the Please choose your search and installations dialog, select **Search for the best driver in these locations and include this location in search**.

4. Use the **Browse...** button to navigate to one of the following paths:

   ```
   <ZDSII Installation>\device drivers\USB\x32
   ```

5. Click **Next** to locate the appropriate driver.
6. Click **Next**, then click **Finish** to complete the installation.

Appendix B. Installing the FTDI USB-to-UART Driver

An FTDI USB-to-UART driver is required to allow your PC to communicate through its USB port to the on-chip UART of the Z8F042A MCU. Observe the following procedure to perform these connections.

1. Ensure that the USB cable is not plugged in to the Buck Converter Battery Charger Design Module's USB J1 connector.
2. Navigate to the following filepath and double-click the `CDM20802_setup.exe` file to begin the driver installation.

   ```
   <ZDS II Installation>\device drivers\FTDI Uart
   ```

3. The installation process will begin and you should observe output similar to the following messages on the screen of your PC:

   ```
   32-bit OS detected
   <installation path>\dpinstx86.exe
   Installation driver
   FTDI CDM driver installation process completed...
   ```

4. When the installation is complete, plug in the B connector of the USB cable into the Board, and the larger A connector into the USB port of your PC.
5. If the driver installation was successful, the Ports (COM & LPT) section of the Device Manager will display USB Serial Port (COMx) or similar message, as highlighted in Figure 2.

Figure 11. A Successful USB-to-UART Driver Installation

Note: To launch the Device Manager on Windows 7 systems, launch the **Start** menu, enter `device manager` in the **Search programs and files** field, and press the Enter key.

To open the Device manager on earlier Windows systems, navigate via the following path:

Start → Control Panel → System → Hardware → Device Manager → Ports (COM& LPT).

Appendix C. Schematic Diagrams

Figure 12 displays a schematic diagram of the Buck Converter Battery Charger Reference Design.

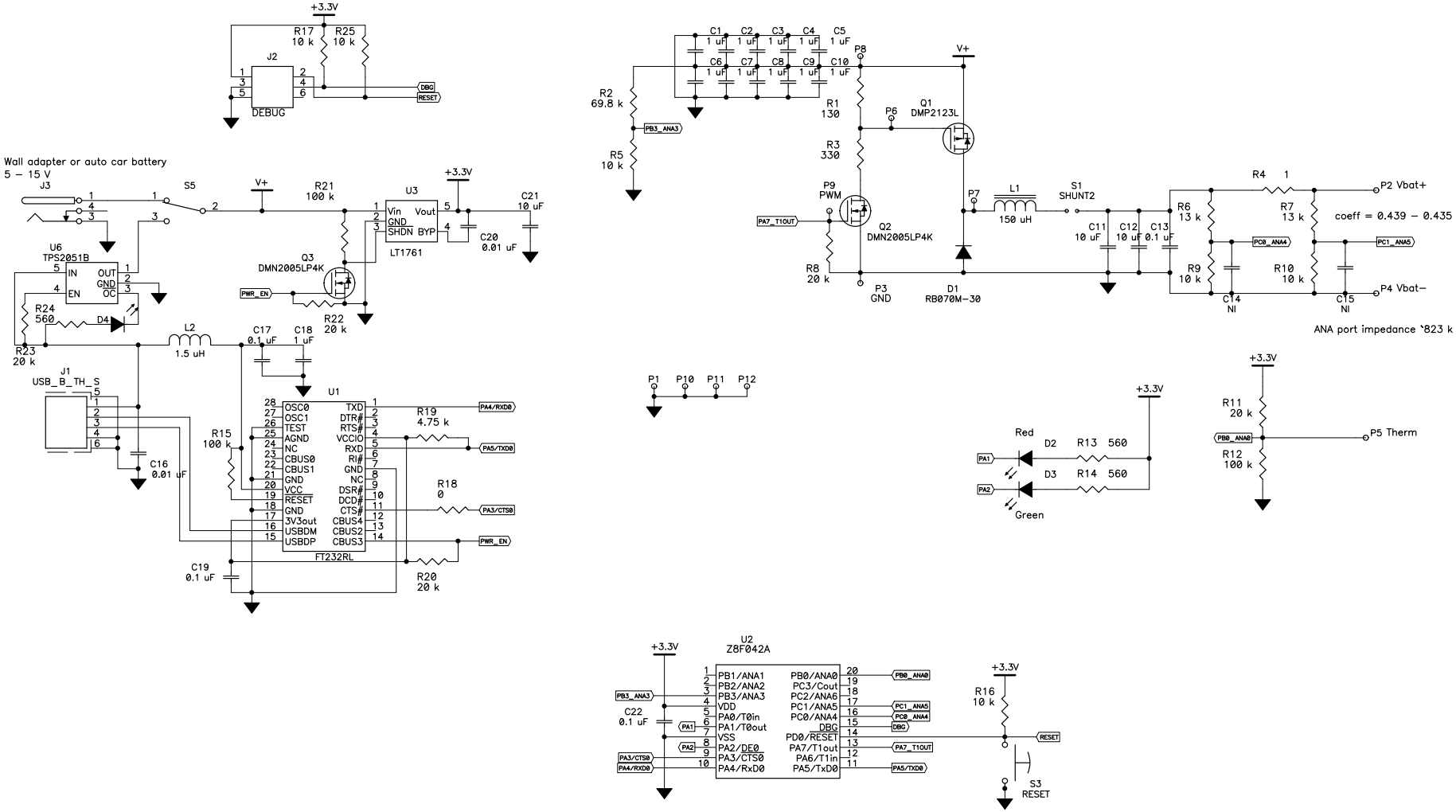

Figure 12. Schematic Diagram of the Buck Converter Battery Charger

Customer Support

To share comments, get your technical questions answered, or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at http://support.zilog.com.

To learn more about this product, find additional documentation, or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base at http://zilog.com/kb or consider participating in the Zilog Forum at http://zilog.com/forum.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at http://www.zilog.com.

Warning: DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

 Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.