Keywords: MAXQ2010 EV kit, MAXQ, 2010, analog to digital, ADC, analog to digital converter, LCD, liquid crystal display, digital voltmeter, voltmeter, photocell, photoresistor, thermistor, a/d, a to d, analog, digital

APPLICATION NOTE 4716

How to Build a Simple, Interactive Digital Voltmeter

By: Stephen Hall
Jul 28, 2010

Abstract: This application note uses the MAXQ2010 microcontroller's evaluation (EV) kit Rev D to implement a digital voltmeter. The article demonstrates how to measure the voltages with the ADC and display them on the EV kit's LCD. An onboard 5-way switch is used to select the ADC channel to be displayed on the LCD. No additional parts are needed aside from the EV kit.

Download an example project with the code described in this application note.

Introduction

The MAXQ2010-KIT showcases the MAXQ2010 microcontroller. The kit has a versatile board with the following features:

- Two pushbuttons for reset and interrupt lines
- One 5-way switch
- 19 exposed GPIO pins
- Eight ADC channels
- A photocell
- A thermistor
- An LCD screen

Using these resources it is simple to build a digital voltmeter.

The MAXQ2010's integrated ADC accepts a voltage from 0V to 3.3V and translates it into a bit value from 0 to 4095 (12 bits of precision). The microcontroller then scales the value to a range from 0 to 3.3000 and writes it on the board's LCD. The application code repetitively samples all eight ADC channels and uses the 5-way switch to determine which channel to display.

Aside from the EV kit, no additional parts are required. Included in the EV kit is a limited version of the IAR Embedded Workbench® software. The full version of the software can be purchased from the IAR™ website. The sample application was written and compiled on version 2.20I.

Getting Started

Set up the board according to the EV kit documentation. You should also install jumpers J30 and J31, which will connect the photocell (R17) to ADC input 6 (AN6) and connect the thermistor (R20) to ADC

input 7 (AN7). You can also add voltages on ports 0–5, as long as the voltages stay within the 0V to 3.3V range.

Next, run the IAR software. In the top left corner click File → Open → Workspace. Navigate to the directory containing the voltmeter project and open voltmeter.eww. The project settings will already be set up. To ensure that the JTAG is set to the correct COM port, right click on the bold-faced project name on the left side of the screen and select Options (**Figure 1**). Go to Categories, click JTAG under Debugger, and enter the COM port to which your programmer is connected (**Figure 2**). To check which COM port is being used, open Device Manager. The EV kit documentation details how to get there.
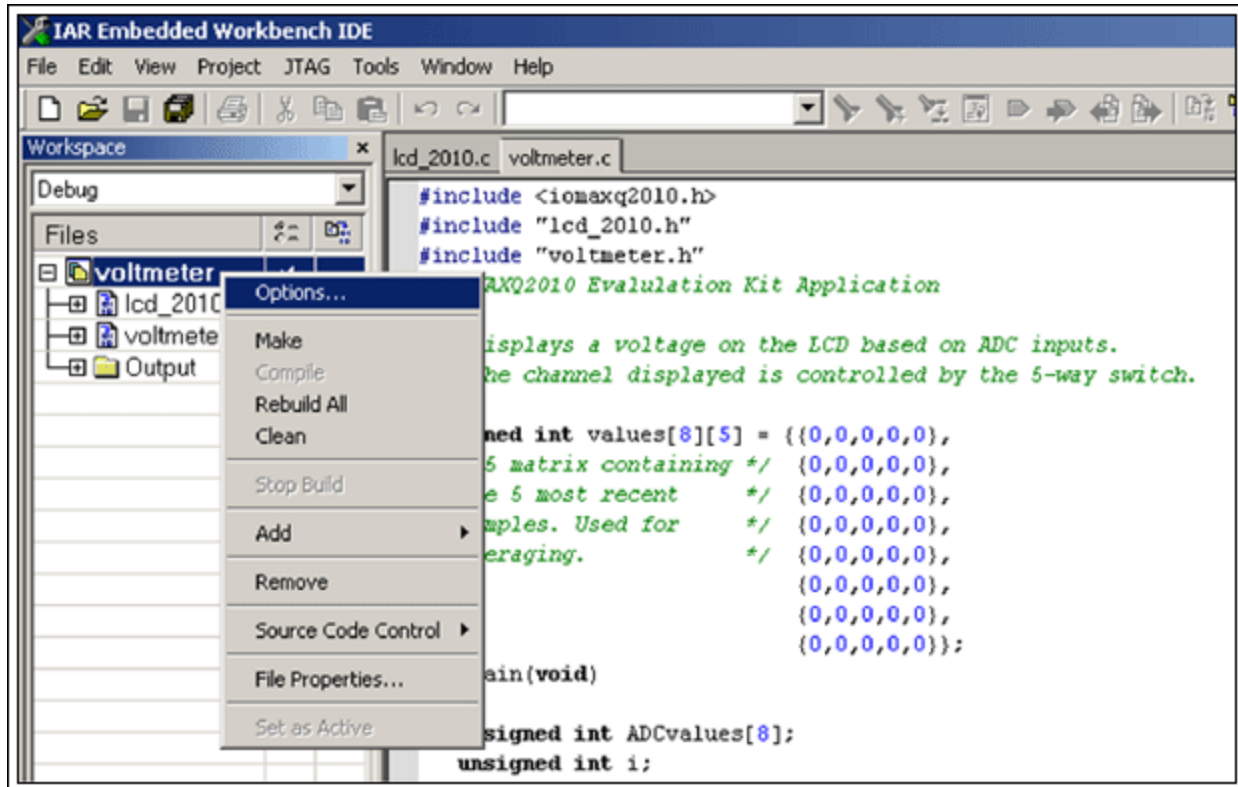

Figure 1. To verify that the JTAG is set to the correct COM port, start with the Options menu.
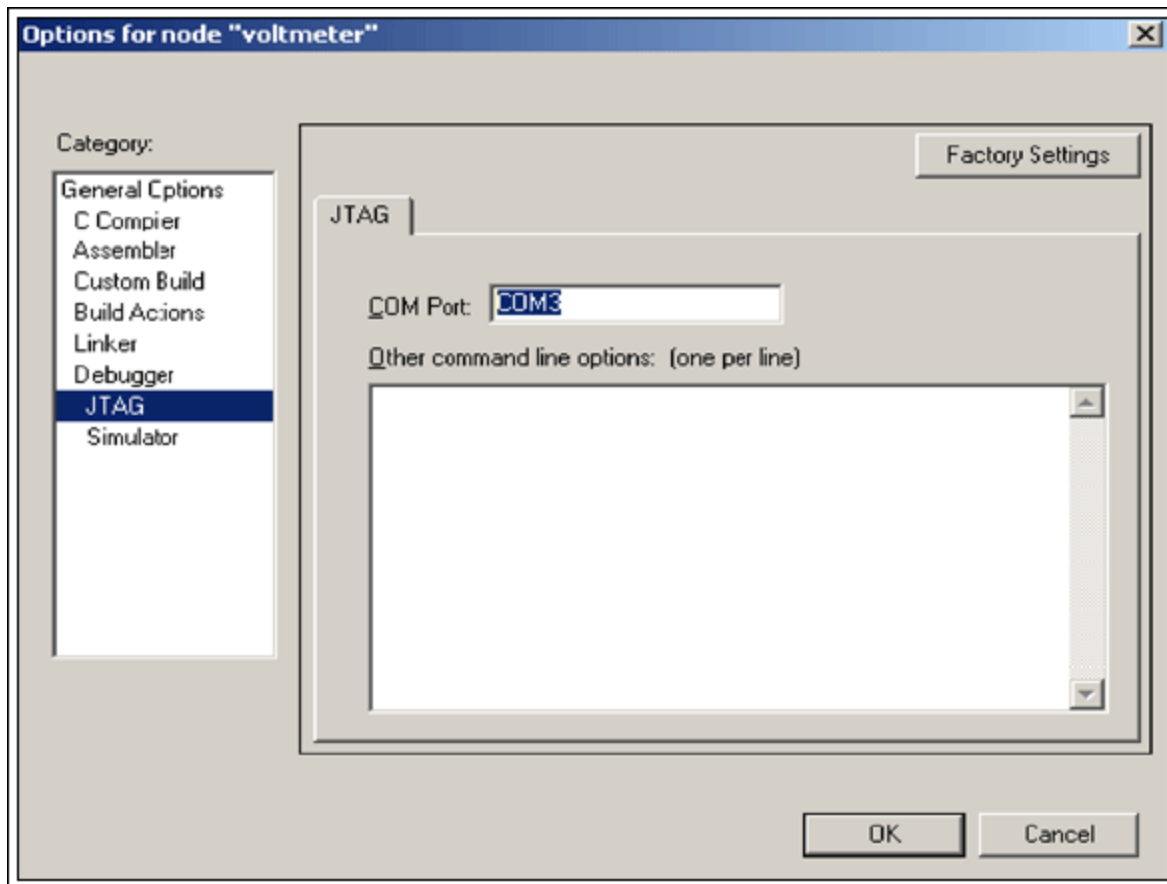
Figure 2. Find the JTAG option and enter the COM port to which your programmer is connected.

Next, click Debug on the far right side of the toolbar (**Figure 3**). This will compile, link, and program your board. The debug menu will be brought up momentarily. A green arrow will be pointing to the main function. Run the program by pressing F5 or clicking the button on the toolbar that looks like a flat piece of paper with three arrows above it.
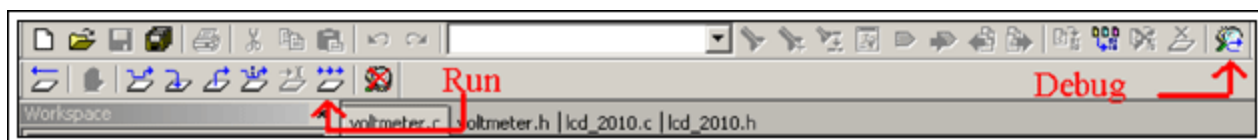

Figure 3. Use the Debug command to compile, link, and program your EV kit board.

Application Results

Featured on the EV kit board are two analog inputs, the thermistor and photocell, which continuously output voltage signals. Both of these inputs are connected to ADC pins so their voltage can be measured. The sample application reads eight ADC values (on channels 0 through 7) at a time. The values are sent to a software lowpass filter that removes high-frequency noise. The selected ADC channel is displayed on the LCD. On the lower right-hand corner of the board is a switch. Pressing this switch down or to the right increments the ADC channel to be monitored. If the channel selected is greater than seven, the software selects channel zero instead. Conversely, pressing the switch up or to the left decreases the ADC channel; channel seven is monitored if the channel selected is less than zero. To display the current channel being monitored, press in the switch. To stop the ADC from sampling, hold the button SW3 down. While SW3 is

held down, the channel can still be changed.

After reset, the microcontroller displays the photocell voltage on channel six. The more light the cell receives, the lower the voltage is across it. Effectively, the voltage at the ADC pin is directly related to the amount of light present. This is because the photocell is part of the voltage-divider on AN6 (see **Figure 4**). In fluorescent lighting, the voltage will sit around 2V. If you shade the corner of the board that contains the cell, the voltage will drop a bit; completely covering the photocell with your finger will drop the voltage to approximately 0.5V. Removing your finger will return the voltage to 2V.
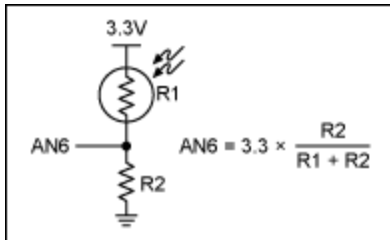

Figure 4. The voltage-divider and the equation that determines the voltage.

$$AN6 = 3.3 \times \frac{R2}{R1 + R2}$$

Note: you may notice that without any input connected to one of the AN pins, the voltage displayed is close to 1.4V. This results from internal bias circuitry and does not affect measurement.

The Operation of the Code

When the code begins executing, three things are initialized: the LCD, the I/O pins for the 5-way switch, and the ADC.

The MAXQ2010 features an internal LCD driver which has two registers to control its operation. The values written to them set up the default mode for this board.

```
LCRA = 0x1860; /* 1/4 duty, 32kHz clock, ground VADJ internally */
LCFG = 0x7F;   /* Activate segments on P0-P3, enable LCD, normal mode */
```

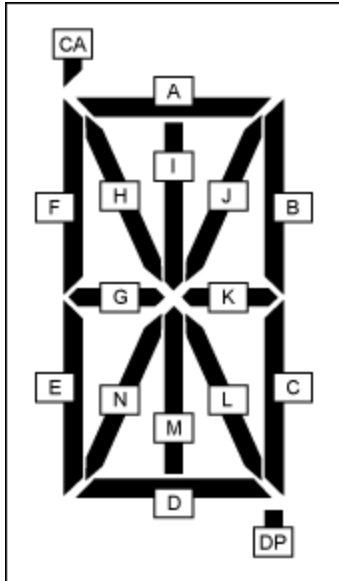In this default mode, the segments are set up as shown in **Figure 5**.

Figure 5. LCD display configuration.

A character is written by sending the high byte of the hex code (CAfedabcDP) to an LCD register, and the low byte (ijklhgnm) to the consecutive LCD register. This application includes a constant character table that can be used to look up the hex codes for the decimal digits. The character table makes displaying digits on the LCD very easy.

The 5-way switch (SW1) uses port pins P4.0 through P4.5. The common pin of SW1 is P4.0 which is set to output low. The other pins are connected to weak pullups by setting the direction to input and writing a 1 to the output. Thus, if a pin has a logic 0 on it when it is sampled, the program knows that the switch is pressed. After each ADC sample, the microcontroller polls the switch and changes the channel accordingly.

It is important to know that all the ADC configuration and data registers are accessed through the Conversion Sequence Address Register (ADDATA), and that the Status Register (ADST) points to the current register. Write instructions to ADDATA will write to the configuration registers and read instructions will read from the data registers.

The ADC is initialized by setting up the ADC Control Register (ADCN) and the configuration registers. First, we must load ADST with the configuration register to which we want to write first. After writing to ADDATA, ADST will automatically increase. Thus eight consecutive writes to ADDATA, when the initial register is Configuration Register 0, will write to registers 0 through 7. Each configuration register controls the configuration for that channel. Writing a 0x07 to the ADADDR instructs the microcontroller to start the conversion sequence with channel 0 and end with channel 7. This will cause the ADC to sample each channel once before raising a flag signaling that the data is ready.

Each result placed in a Data Register is accessed through ADDATA. To start at Data Register 0, ADST is set to 0. Eight reads from ADDATA retrieve all eight values of the conversion sequence. Here, the data register also increases with consecutive reads from ADDATA.

Each value read from ADDATA ranges from 0 to 4095. For the MAXQ2010, 4095 represents the ADC reference voltage, AVDD, which is typically 3.3V. The program scales the 12-bit value up to a range from 0 to 33000 instead of down to 3.3 because integer division would cause truncation. For example, an ADC value of 2036 would be displayed as 1.0000 instead of 1.6407.

```
unsigned int voltage = ADCvalue * 3.3 * 10000 / 4095;
```

The result of this code is a value from 0 to 33000. The digits are masked off and displayed on the LCD with the decimal point added in the correct place. Essentially the conversion of the bit value is to 10^{-4} volts. Adding the decimal place later reverts the value back to volts.

Conclusion

This application note has demonstrated how to implement a digital voltmeter on the MAXQ2010 microcontroller. Using the peripherals of the MAXQ2010-KIT, voltage data from the internal ADC is easily written on the LCD. For different applications, the EV kit LCD can display alphanumeric characters. The 5-way switch can be used to create an extensive menu interface. The ADC supports eight channels hooked up to parts such as potentiometers, infrared distance sensors, resistor ladders, and any other device that returns an analog voltage.

IAR Embedded Workbench is a registered trademark of IAR Systems AB.
IAR is a trademark of IAR Systems AB.

Related Parts		
MAXQ2010	16-Bit Mixed-Signal Microcontroller with LCD Interface	Free Samples
MAXQ2010-KIT	Evaluation Kit for the MAXQ2010	

More Information
For Technical Support: http://www.maximintegrated.com/support
For Samples: http://www.maximintegrated.com/samples
Other Questions and Comments: http://www.maximintegrated.com/contact

Application Note 4716: http://www.maximintegrated.com/an4716
APPLICATION NOTE 4716, AN4716, AN 4716, APP4716, Appnote4716, Appnote 4716
Copyright © by Maxim Integrated Products
Additional Legal Notices: http://www.maximintegrated.com/legal