Getting Started with DeepCover Secure Microcontroller (MAXQ1050) Evaluation (EV) Kit and the Rowley CrossStudio Compiler for the MAXQ30

May 14, 2012

Abstract: This application note describes how to create, build, and debug applications targeted for the DeepCover® Secure Microcontroller (MAXQ1050) with RISC architecture. The example presented in this article uses the Rowley CrossStudio for the MAXQ30 compiler to demonstrate how the MAXQ1050 can read a smart card.

Introduction

The DeepCover® Secure Microcontroller (MAXQ1050) is a low-power microcontroller, designed for USB secure token and smart card reader applications that require certificate-based or other public key cryptographic schemes. The device uses the 32-bit, pipelined, MAXQ30 microcontroller CPU. The MAXQ1050 has 128KB flash memory, 12KB of volatile SRAM, 4KB of battery-backed NV SRAM, and 256B of battery-backed secure zeroization NV SRAM.

The microcontroller is powered either from the USB bus or by a separate 3.3V supply. A battery connection is provided for applications that need to maintain data in the NV SRAMs. In battery-backed mode, the NV SRAM and security sensors consume less than 240nA (typ).

The MAXQ1050 has a number of important features, including:

- High-performance, low-power, 32-bit MAXQ30 RISC core
- Operates from USB power or a single 3.3V supply
- Runs from 20MHz (typ) internal oscillator
- Supports external 12MHz/24MHz crystal oscillator for microcontroller and USB operation
- On-chip 2x/4x clock multiplier
- 16-bit instruction word, 32-bit internal data bus
- 16 x 32-bit accumulators
- 16 x 32-bit general-purpose working registers
- Up to 20 general-purpose I/O pins
- 5V tolerant I/O
- Memory:
  - 128KB flash memory
  - 256B of secure NV SRAM
  - 4KB battery-backed NV SRAM
  - 12KB SRAM
- Security:
  - Unique 64-bit serial number
  - Tamper detection with rapid key/data destruction
- Secret key destruction on tamper events
- Permanent loader lockout option
- Hardware accelerators for AES, RSA, DSA, ECDSA, DES, 3DES, SHA-1, SHA-224, SHA-256
- True hardware random-number generator
- Temperature and voltage sensors to detect attacks
- Two self-destruct input pins

- Additional peripherals
  - Power-fail warning
  - Power-on-reset/brownout reset
  - JTAG I/F for system programming and accessing on-chip debugger
  - Full-speed USB device with six endpoint buffers and integrated transceiver
  - ISO 7816 smart card UART with FIFO
  - 16-bit programmable timers/counters with prescaler, capture/compare, and PWM
  - SPI master/slave hardware
  - Programmable watchdog timer
  - Up to 20 general-purpose I/O pins with eight external interrupts

The MAXQ1050 evaluation (EV) kit provides a proven, reliable platform for developing low-power applications for the MAXQ1050 processor. The kit includes a RS-232 serial port, a native USB port, 4 pushbuttons for user input, 4 LEDs for application usage, and headers for accessing all the MAXQ1050's I/O pins. Jumpers are provided that allow configuration of the smart card interface. A separate MAXQ622 is also included and provides the JTAG interface to the MAXQ1050 and the host loader/debugger.

**Setting Up the MAXQ1050 EV Kit**

A photograph of the MAXQ1050 EV kit board is provided in Figure 1. The following hardware components are contained in the EV kit. These components are used for implementing and verifying the demonstration program in this application note:

1. MAXQ1050 EV kit board
2. USB cable (Type A to Mini-B)
3. Regulated power supply (5V, ±5%, 300mA, center positive)
4. RS-232 serial cable (9-pin male to 9-pin female)
5. Smart card

The MAXQ1050 has a number of jumpers to configure. The jumpers should be configured as shown in Table 1. See Figure 2 for jumper locations.
Table 1. Jumper Configurations for the MAXQ1050 EV Kit Board

<table>
<thead>
<tr>
<th>Jumper(s)</th>
<th>State</th>
<th>Purpose if Jumper is Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>JH1</td>
<td>Present</td>
<td>Smart card V50 power - from V50</td>
</tr>
<tr>
<td>JH2</td>
<td>Present</td>
<td>VDD LED on</td>
</tr>
<tr>
<td>JH3</td>
<td>Removed</td>
<td>CLKDIV1 (pull up/down)</td>
</tr>
<tr>
<td>JH4</td>
<td>Removed</td>
<td>CLKDIV2 (pull up/down)</td>
</tr>
<tr>
<td>JH5</td>
<td>Removed</td>
<td>5V_3V (pull up/down)</td>
</tr>
<tr>
<td>JH6</td>
<td>Removed</td>
<td>1.8V (pull up/down)</td>
</tr>
<tr>
<td>JH7</td>
<td>Present</td>
<td>VDD power</td>
</tr>
<tr>
<td>JH8</td>
<td>Present</td>
<td>JTAG reset</td>
</tr>
<tr>
<td>JH9</td>
<td>Present</td>
<td>Smart card VDD power - from VDD</td>
</tr>
<tr>
<td>JH10</td>
<td>VBAT+VDDIO</td>
<td>Connects VBAT to VDDIO</td>
</tr>
<tr>
<td>JH11</td>
<td>Present</td>
<td>TDO - JTAG</td>
</tr>
<tr>
<td>JH12</td>
<td>Present</td>
<td>TMS - JTAG</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>JH13</td>
<td>Present</td>
<td>TDI - JTAG</td>
</tr>
<tr>
<td>JH14</td>
<td>Present</td>
<td>TCK - JTAG</td>
</tr>
<tr>
<td>JH15</td>
<td>Present</td>
<td>CLKDIV1 - P1.7 - smart card</td>
</tr>
<tr>
<td>JH16</td>
<td>Present</td>
<td>DUT_VDD</td>
</tr>
<tr>
<td>JH17</td>
<td>Present</td>
<td>SC_IO - P1.3 - smart card</td>
</tr>
<tr>
<td>JH18</td>
<td>Present</td>
<td>RXD - P1.0</td>
</tr>
<tr>
<td>JH19</td>
<td>Present</td>
<td>SC_CLK - P1.2 - smart card</td>
</tr>
<tr>
<td>JH20</td>
<td>Present</td>
<td>TXD - P1.1</td>
</tr>
<tr>
<td>JH21</td>
<td>Present</td>
<td>CLKDIV2 - P0.7 - smart card</td>
</tr>
<tr>
<td>JH22</td>
<td>Present</td>
<td>5V_3V - P0.6 - smart card</td>
</tr>
<tr>
<td>JH23</td>
<td>Present</td>
<td>RS 232 PWR</td>
</tr>
<tr>
<td>JH24</td>
<td>Present</td>
<td>1_8V - P0.5 - smart card</td>
</tr>
<tr>
<td>JH25</td>
<td>Present</td>
<td>AUX2IN - P0.4 - smart card</td>
</tr>
<tr>
<td>JH26</td>
<td>Present</td>
<td>VBUS/V45</td>
</tr>
<tr>
<td>JH27</td>
<td>Present</td>
<td>AUX 1 IN - P0.3 - smart card</td>
</tr>
<tr>
<td>JH28</td>
<td>Present</td>
<td>SC_OFF - P0.2 - smart card</td>
</tr>
<tr>
<td>JH29</td>
<td>Present</td>
<td>SC_RST - P0.1 - smart card</td>
</tr>
<tr>
<td>JH30</td>
<td>Present</td>
<td>P0.0 LED enable</td>
</tr>
<tr>
<td>JH31</td>
<td>Present</td>
<td>P0.1 LED enable</td>
</tr>
<tr>
<td>JH32</td>
<td>Present</td>
<td>P0.2 LED enable</td>
</tr>
<tr>
<td>JH33</td>
<td>Present</td>
<td>P0.3 LED enable</td>
</tr>
<tr>
<td>JH34</td>
<td>Removed</td>
<td>SDI1 closed</td>
</tr>
<tr>
<td>JH35</td>
<td>Removed</td>
<td>SDI2 closed</td>
</tr>
<tr>
<td>JH36</td>
<td>Present</td>
<td>CMDVCC - P0.0 - smart card</td>
</tr>
</tbody>
</table>
Setting Up the Serial-to-USB Driver

The MAXQ1050 evaluation kit is designed to interface to the host PC over a USB interface, which allows software running on the host to communicate with the JTAG loader/debuger on the MAXQ1050 microcontroller.

Connecting the USB cable between the PC and the EV kit will cause Windows 7 to start the driver installation. The USB cable is plugged into "CN1 JTAG USB" on the EV kit. The driver creates a virtual COM port on the PC. In the event that the PC does not detect your driver, you can manually browse and select the driver in the resource library package ‘install’ folder.

To determine the virtual COM port connected to the USB cable, click on the Windows Start menu, then Control Panel, and then Device Manager. In the Device Manager window, open the section of the tree display labeled Ports (COM & LPT). You should see a listing for MAXQ USB Serial Port (COMX), where X is the number of the COM port that has been assigned. Figure 3 has an image of the Device Manger window showing the COM9 serial port.

This COM port number may vary depending on previously installed devices and previous USB-to-serial drivers that have been installed. Make note of the virtual COM port that is connected to the MAXQ1050 USB so that you can reference it later if and when you use the Rowley CrossWorks Compiler for development.

Figure 2. Location of jumpers on the MAXQ1050 EV kit.
Installing and Using the Microcontroller Tool Kit (MTK2)

The code in this application note outputs data through the RS-232 connector. To view the data, we need a terminal or a terminal emulator. One option for the terminal emulator is to use the Maxim-provided Microcontroller Tool Kit (MTK2) utility. If you have not already installed MTK2, you can do so by following the link provided in the resource library package.

After installing MTK2, run it from the Start menu. In the initial Select device dialog box, select Dumb Terminal and click OK (Figure 4). This opens MTK2 in a simple text terminal emulator mode (as opposed to a mode designed to communicate directly with the loader of a specific device).
Next, you will need to configure and open the serial port. From the MTK2 menu, select **Options > Configure Serial Port**. In the dialog box that appears, select the physical COM port to which your serial cable is connected (or the virtual COM port if you are using a USB-to-serial adapter) and select 115200 for the Speed setting (*Figure 5*). (Note: if the COM port number you want to use is not shown in the drop-down list, you can enter it directly in the Port field).

*Figure 5. Configuring the Serial Port in MTK2.*

Finally, select **Target > Open COMx at 115200 baud** from the menu. The main section of the MTK2 window will turn white to indicate that the port has been opened.

**Getting and Installing the Rowley Compiler**

Go to [www.rowley.co.uk/maxq30/index.htm](http://www.rowley.co.uk/maxq30/index.htm) in your browser (*Figure 6*). The page that appears may look different than the one displayed in Figure 6, due to browser compatibility and future Rowley page updates.
Figure 6. Rowley’s MAXQ download page.

Click on Version 2.2.0 for Windows or the latest version that is available. The browser will ask you where to save this file. Select a directory and save the file (Figure 7 and Figure 8).

Figure 7. Request by browser to save compiler installation program.
Figure 8. Saving the Rowley setup program to a temporary directory.

Using Windows Explorer, go to the directory to which you downloaded the install file and double-click on the file name. Your screen will dim under Windows 7 and you will be asked **Do you want to allow the following program to make changes to your computer?** Click **Yes**. The software will begin extracting files and you will see a Welcome window (Figure 9). Click on **Next**.
A License Agreement window will be displayed (Figure 10). Accept the agreement and click Next>.

A new window will ask for a place to install the compiler (Figure 11). Select a directory and click Next>.
The next window asks for the program folder name to be used in the Start menu (Figure 12). Select a name and click Next>.

The Associate Files window will follow (Figure 13). Click Next>.
Now we get the Start Installation window (Figure 14). Click Install.

The last window to come up is the Installation Complete. Click Finish.
Click the Windows 7 Start icon and then All Programs > Rowley Associates > CrossWorks for MAXQ30 2.2 > CrossStudio for MAXQ30 2.2 (Figure 15).

![Figure 15. Searching through the All Programs list to find the compiler.](image)

The compiler is up, but not licensed (Figure 16). Notice in the Update pane that there are updates to install. After activating the compiler, install the update.
Figure 16. The opening CrossStudio window.

Under the Tools button, click License Manager (Figure 17).
We intend to 'evaluate' the compiler. After selecting this option, the request shown in Figure 18 appears. Click Send e-mail to use your built-in email program. You can also select and copy the data in the Activation Request window, paste it into a different email program, and then send it off to license@rowley.co.uk.
An activation key will be returned to you in a few hours. Copy the activation key from the email and paste it into the indicated window of the Activate CrossWorks window (Figure 19). These keys are unique to your hardware. Click on Install License.
Figure 19. An activation key pasted into the Activate CrossWorks window.

The evaluation license will now appear, along with the days remaining that you can use it (Figure 20).
Close this window and exit the compiler.

Getting Started with the Rowley Compiler

This demonstration program is a simple application that will open the serial port and display a request to insert the smart card. Once the card is detected, it is powered and the answer-to-reset (ATR) data is printed. The ATR has information about communication parameters and the cards nature and state. It must conform to the ISO/IEC 7816 standard.

From the MAXQ1050 EV Kit resource library package, copy the directory from \Sample Code\C\isouart to the CrossWorks projects directory. On Windows 7, this directory is located under \Users\<username>\My Documents\CrossWorks_Projects. (This user guide is using <username> as a generic name.)

Start the Rowley CrossStudio for MAXQ30 compiler (Figure 21). Notice that this CrossStudio window has the time left on the evaluation and update status.
Figure 21. A licensed compiler with an update to be installed.

In the Updates pane, click Install. The Package Manager will bring up the Summary window (Figure 22).
Click on the Next button to install the update (Figure 23).

Click on the Finish button and start a new project.

Under File, select New Project (Figure 24). In the New Project window, under Project Templates, select A C executable. Enter ISOUart into the Name field. Append isouart to the location path in the Location field. Click Next.
Figure 24. New Project window.

Under Code Generation Options (Figure 25), find Target Processor, and select MAXQ1050 from the pull-down list. Leave all the other options with the default value. Click Next.
Figure 25. Selecting the MAXQ1050 and other code generation options.

The next window (Figure 26) has some of the files to compile. The file main.c is already in the source directory, so just press the Next button.
Figure 26. New project files.

The next window (Figure 27) lets us create directories for both a debug version and a release version of our project. Both should be selected. Click Finish.
Figure 27. Selecting the debug and release directories.

The project now requests to create a new main.c (Figure 28). Use the existing one, so click No.

Figure 28. Compiler request to create a new main.c.

On right-hand side of the window and a few lines down, you will find the Project Explorer (Figure 29). Click the + symbol in front of Project 'ISOUart'. Click the + symbol in front of Source Files to reveal main.c.

Highlight and right-click on Source Files. Click Add Existing File.... Highlight delay.c, MAXQ30_7816_LIB.C, and serial.c, and press Open. If you hold the Ctrl key, you can select all three files at once. Repeat the process, and select delayasm.asm. Click Open.

It is now time to power up your MAXQ1050 EV KIT. Plug in the JTAG USB cable to CN1 and your computer. Connect the 9-pin serial cable to the 9-pin connector on the EV kit and to the computer's serial port. This application note assumes that the 9-pin serial cable is connected to COM1, and the serial JTAG USB connection is on COM9.
Go further down to the **Target** list and click **Maxim Serial JTAG Adapter**. This will open the **Connection** entries. Set the COM port to which you are connected. In this example, we are connecting to COM9. Now go to the middle section titled **Targets**, then connect to the JTAG port. This can be done in three different ways: 1) Double-click on **Maxim Serial JTAG Adapter**; 2) Right-click on **Maxim Serial JTAG Adapter** and select **Connect**; or 3) Press the most left icon below **Targets**. The status bar at the bottom of the window will go from **Disconnected** to **Maxim Serial JTAG Adapter** (Figure 29).

![CrossWorks is connected to the Maxim Serial JTAG Adapter on COM9.](PDF, 1.9MB)

**Figure 29.** CrossWorks is connected to the Maxim Serial JTAG Adapter on COM9.

Now highlight **Project 'ISOUart'** and right-click. Select **Build**. In the **Output** pane, you will see the project building.

Note: Make sure the Microcontroller Tool Kit (MTK2) is running and connected to the MAXQ1050 EV kit at 115200 baud via the 9-pin serial connector.

Go to the top buttons and click **Project** and **Start Without Debugging**. In the output pane, you can watch as the compiler checks the configuration, erases the MAXQ1050 flash, downloads the program, and verifies the program (Figure 30).
Once the program is loaded, it will start running. You will see a banner and a request to Insert Card Now. Put the supplied smart card in to the card reader, and watch as the card is found and powered up, and the ATR message is printed (Figure 31). Pressing the RST_N (SW7) button on the EV kit will make the program run again.

The ISOUart Program

This demonstration program comprises several source files. The files delay.c and delayasm.asm demonstrate how to call assembly code from 'c.' These particular routines implement a callable microsecond delay in the assembly file, and a callable millisecond delay in the c source.

Serial.c implements the int __putchar(int ch) and char __getchar(void) routines. Once these two routines are written, the higher-level stdin and stdout routines can be used with the RS-232 serial port. Serial.c also implements the void serial_init(void) routine, which initializes the serial port to operate at 115200 baud.

MAXQ30_7816_LIB.C implements the ISOUART functions.

The main program is listed in Figure 32.
ISO uart example. Retrieve and Display ATR on LCD

This example only uses Slot 1. The SIM Card slot is not used.

/*
#include "MAXQ30.h"
#include "MAXQ30_7816_LIB.H"
#include "delay.h"
#include "serial.h"
#include <stdio.h>

void main(void)
{
    int status, i;
    uint8_t buff[128];
    serial_init();
    printf("\nMAXQ1050 ISOUART Ex.\n");
    // Initialize the Smart Card interface
dssc_init();
    dssc_selectcard(0);
    if (!dssc_checkpresence())
    {
        printf("Insert Card Now\n");
        // Wait forever until card is present
        while (!dssc_checkpresence());
    }
    // Wait for user to see output
delay_ms(1000);
    printf("\nCard found. Power up\n");
    // Set up for transaction and get an Answer To Reset
    status = dssc_powerup(POWERUP_5V);
    if (status == ERR_DO_WARM_RESET)
    {
        printf("Warm Reset\n");
        status = dssc_warmreset();
    }
    // Wait for user to see output
delay_ms(1000);
    if (status != STATUS_SUCCESS)
    {
        printf("\nPower up Error: %d\n", status);
    }
    else
    {
        status = dssc_getATRbuffer(buff, 128);
        printf("\nATR Received\n");
        // Wait for user to see output
delay_ms(1000);
        for(i=0;i<status;i++)
        {
            printf("%02X", buff[i]);
            if ((i % 20) == 19)
                printf("\n"); // Need to shift to next line
            else
                printf(" ");
        }
        printf("\n");
    }
    // Power down card interface
dssc_powerdown();
while (1){};
}

Figure 32. Listing of ISOUart's main.c.

For More Information

The source code for this application and many others is available in the resource library package distributed with the MAXQ1050 EV kit. All the Rowley project files are also included in the distribution.

Maxim has software libraries, application notes, and reference designs available for further design assistance. Search Maxim's microcontroller product line pages or contact Technical Support for the latest information on available libraries and tools.

DeepCover is a registered trademark of Maxim Integrated Products, Inc.
MAXQ is a registered trademark of Maxim Integrated Products, Inc.
Windows is a registered trademark and registered service mark of Microsoft Corporation.

Related Parts

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXQ1050</td>
<td>DeepCover Secure Microcontroller with USB and Hardware Cryptography</td>
</tr>
</tbody>
</table>

More Information

For Technical Support: [http://www.maximintegrated.com/support](http://www.maximintegrated.com/support)
For Samples: [http://www.maximintegrated.com/samples](http://www.maximintegrated.com/samples)
Other Questions and Comments: [http://www.maximintegrated.com/contact](http://www.maximintegrated.com/contact)

Application Note 5393: [http://www.maximintegrated.com/an5393](http://www.maximintegrated.com/an5393)
APPLICATION NOTE 5393, AN5393, AN 5393, APP5393, Appnote5393, Appnote 5393
© 2013 Maxim Integrated Products, Inc.