Keywords: DS5250 EV Kit, RSA, MTK2

APPLICATION NOTE 4004

RSA Key Generation in DS5250

Mar 19, 2007

Abstract: The DS5250 microcontroller evaluation (EV) kit is a proven platform to evaluate the capabilities of this high-speed secure microcontroller. This application note demonstrates how to set up the EV kit and generate the RSA key-pair of the bit length needed for an application. The Keil µVision2® compiler is used to develop the library and sample application. The Microcontroller Tool Kit (MTK2) is used to load the application on the EV kit and to observe the results.

Introduction

This application note demonstrates how to generate RSA key-pair sets using the sample application binary (rsa.hex) associated with this application note. The application note describes how to load and run the software on the DS5250 Evaluation Kit (EV kit).

The EV kit and the software related to this application note are available to customers with a valid NDA for the this device. Customers without an NDA for the DS5250 can initiate the process through the DS5250 quick view Customers with an existing NDA can contact Technical Support to obtain the software referenced in this application note.

Getting Started with RSA Key-Pair Generation

Build and execute the RSA key-pair sample application program written in C using the Keil µVision2 IDE.

1. Install the Keil µVision2 IDE.
2. Open the project `rsa.uv2`.
3. Click on **Project→Rebuild All Target FILES** to generate the `rsa.hex` file.

Loading the Sample Application onto the DS5250-KIT EV Kit

Install the Microcontroller Tool Kit (MTK2_INSTALL) to load the application onto the EV kit. When MTK2 is launched, a dialog box similar to the one shown in **Figure 1** is displayed.

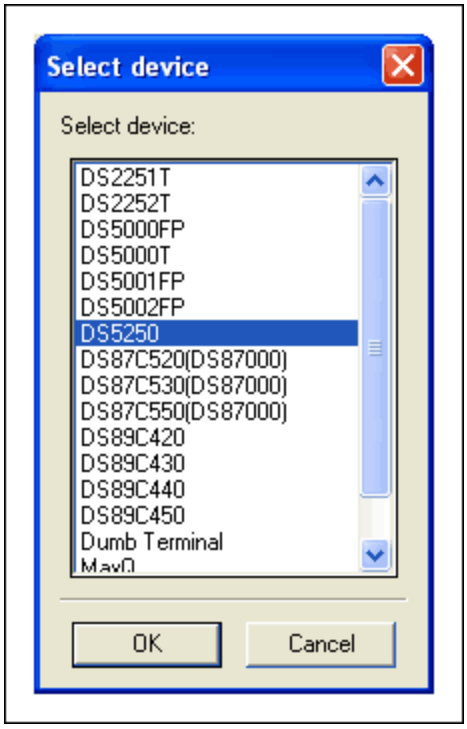

Figure 1. MTK2 options on startup.

Select the option **DS5250** to communicate to the EV kit. From the MTK2 menu, **Options→Configure Serial Port**, select the COM port that you are using and choose 115200 speed. Next select the **Target→Open COMx port at 115200 baud** option, and then **Target→Connect to Loader** to reset the EV kit. The DS5250 loader should print a message something like the following:

```
DS5250 SECURE LOADER VERSION 1.0 COPYRIGHT (C) 2002 DALLAS SEMICONDUCTOR
      LID: 62E9490700000071 8284

>
```

Configure the EV kit memory by sending the following commands to the loader.

```
W MSIZE 12[1]
W MCON 81[2]
```

[1]W MSIZE 12 identifies the external program and data memory chip size as 512kb.

[2]W MCON 81 identifies the memory as Partition Mode.

From the **File** menu, select **Load HEX File** and then the `rsa.hex` file that you just created.

Choose **Target→Disconnect from Loader** to execute the program loaded onto the EV kit. The prompt appears as seen in **Figure 2**.

Enter key length bits to be generated:

Enter the number (for example, 1024) and wait for the application to display the results. The application displays the execution status as shown in Figure 2. It takes approximately 60 seconds to generate a 1024 bit-length RSA key-pair, encrypt, and decrypt the random message. This time can vary for each execution. The minimum, maximum, and average times needed to generate an RSA key-pair for various

bit lengths are tabulated in **Table 1**.

Figure 2. Execution status and results of sample application.

Developing a Simple Application Using RSA Key-Generation Library

The library provides four easy-to-use interface functions in C to generate the key-pair and encrypt/decrypt the user message using the private/public key. Refer to the `rsalib.h` file to see the prototypes of these interfaces. The application provided with this application note demonstrates the use of these interface functions:

```
rsa_generateKeySet(...)
rsa_bignumModExp(...)
rsa_newNum()
rsa_freeNum()
```

Typical test results for different bit lengths are shown below.

Table 1. Average Time Needed for Generating an RSA Key-Pair

RSA Bit Length Generated	Number of Tests Run	Minimum Time Taken for the Test (in seconds)	Maximum Time Taken for the Test (in seconds)	Average Time Taken per Test (in seconds)
256	60	3.4	10.3	4.8
512	60	6.1	21.0	10.76
1024	60	13.5	62.0	26.6
2048	60	36.6	313.2	122.4
3072	30	102.7	731.9	369.8

Conclusion

The RSA key generation library provided by Maxim allows applications written in C to access the power and functionality of the DS5250 microcontroller hardware. RSA key-pairs can be generated up to a maximum of 4096 bits.

Relevant Links

Application note 2783, "Using the Keil C Compiler for the DS5240/DS5250"
Secure Microcontroller Family User's Guide and Supplements
DS5250 High-Speed Secure Microcontroller Data Sheet

Related Parts	
DS5250	High-Speed Secure Microcontroller
DS5250	High-Speed Secure Microcontroller

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 4004: http://www.maximintegrated.com/an4004
APPLICATION NOTE 4004, AN4004, AN 4004, APP4004, Appnote4004, Appnote 4004