TWR-K70 MSD Host Bootloader based on AN4368 source code

By: Technical Information Center

1 Introduction

The TWR-K70 MSD bootloader application is based on the Application Note source code AN4368 and has the same functionality presented in the document (for more details refer to Referenced Documents).

Some important modifications were made in order to complete the migration process of the mentioned source code to be used with the TWR-K70F102M that implements a MK70FN1M0VMJ12 device on board.

Contents

2 Important Modifications

The most important modifications done to complete the migration process are:

- Change of the *P2* (100MHz devices) device configuration for the *P3* (120MHz devices) device configuration. These configuration changes include the MCG registers to fit with the P3 characteristics, UART module signals and the BSP to support the TWR-K70.
- Use of an USB Stack example application to support 120MHz devices and the USB module requirements. To accomplish this the MSD Host example for the TWR-K70 included in the USB Stack v4.1.1 was used as the base of the application. The MSD Host example can be found in the installation folder of the USB Stack v4.1.1 in the path: *<installation_folder>\Freescale USB Stack v4.1.1\Source\Host\examples\msd\cw10\kinetis_k70*
- Change of the Flash Routines to implement 8 bytes writings supported by the K70 instead of the 4 bytes writings supported by the K60.
- Modifications on the S19 parser code in the file Loader.c of the application. This is up to the programmer considerations, in this example just simple modifications were done to meet the K70_120MHz device's requirements.
- Use of CodeWarrior 10.6 instead of CodeWarrior 10.1 as the development tool that was the tool for the AN4368 source code.

NOTE:

The example was only tested and only supports S19 files generated in a particular way. The section Generating the S19 file of the User Application. describes this process.

3 Compiling and Running the Application.

To compile and run the application the normal process is followed:

1. Open the application in the workspace, click on the arrow next to the hammer icon and compile for Flash configuration.

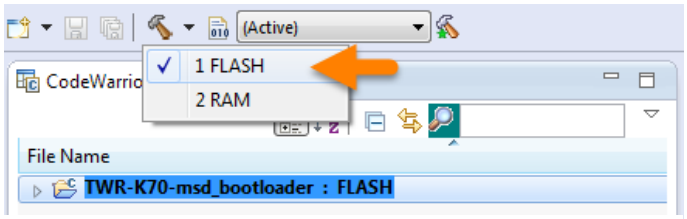

Figure 1. Compile Flash configuration

2. Debug the project setting the debug configurations by clicking the arrow next to the bug icon. For this example OSJTAG, PnE Multilink and Segger J-Link are already configured as debug tools.

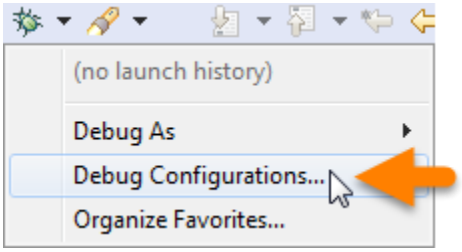

Figure 2. Debug configurations

3. In the next window choose the debug option for your board and click on debug.
4. Once the application is downloaded to the device click on the Resume button to run the application.

When running the application the user can decide whether run the bootloader mode or run the user application if there is one present. When no user application is present the bootloader mode runs automatically.

The modes are:

- Run user application: After an application is flashed to the device the user application will run automatically after reset.
- Run in bootloader mode: The code checks, just after the reset, for the SW2 switch (TWR-K70F120) pressed. When is pressed the bootloader will run showing the message in figure 3 in the terminal. This example uses the UART port of the TWR-K70 connected to the TWR-SER with a USB2SER connected. Configure the terminal:
  - 115200 baud rate.
  - 8 bits.
  - No parity mode

When running the application in bootloader mode the next information is showed in the terminal:

1. The message in the figure 3 indicates the device is in bootloader mode waiting for an attached USB MSD device.

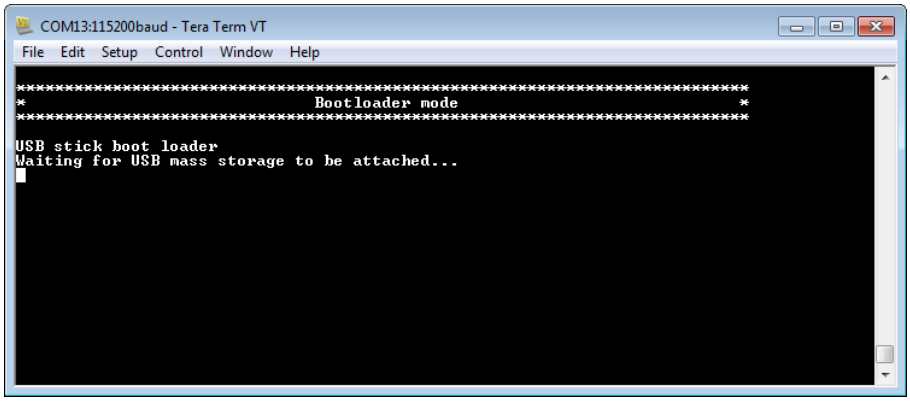

Figure 3. Bootloader mode

2. After a device is attached, containing a valid flash image according the AN4368 requirements, the process of the application is the next:

    1. Enumeration of the USB MSD device.
    2. Search the file image.S19.
    3. Found an image.S19 file.
    4. Erase memory based on the file size.
    5. Recognize a valid S19 file.
    6. Flash the application.

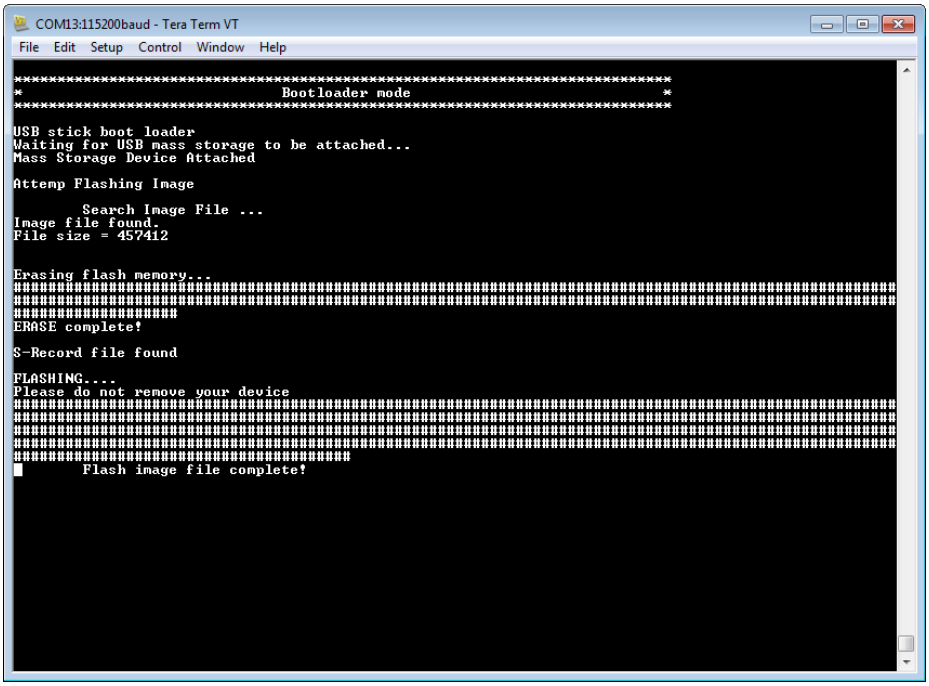

Figure 4. Flashing user application

After that press the reset button to start running the user application.

For more information and details about the application process and code refer to the Application Note AN4368 and the source code of the application. The referenced documentation is included in section Referenced Documents.

4 Generating the S19 file of the User Application.

As mentioned before this example needs that the user application S19 file is created in a particular way. The process to generate the S19 file is the next:

1. Generate the srec (S19) file, this need to be a S3 type, with the CodeWarrior's generate flash image tool. Follow the next tutorials to accomplish this:
   - S-Record Generation with GCC for ARM/Kinetis.
   - S-Record Manipulation with GNU objcopy and Burner Utility: Of this tutorial just follow the Forcing 32bit S3 Records section.
2. Open the Burner application; it can be found in the installation folder of CodeWarrior in this path: *<install_folder>\Freescale\CW MCU v10.6\MCU\prog\burner.exe*; follow the next steps:
   1. Click on Burner to open burner options.
   2. Select the S19 file created in the step above.
   3. Select the location to store the new file.
   4. Click on Content tab.
   5. Select S3 type.
   6. Change to 8 bytes per line.
   7. Set max length.
   8. Click OK.
   9. Click Execute.

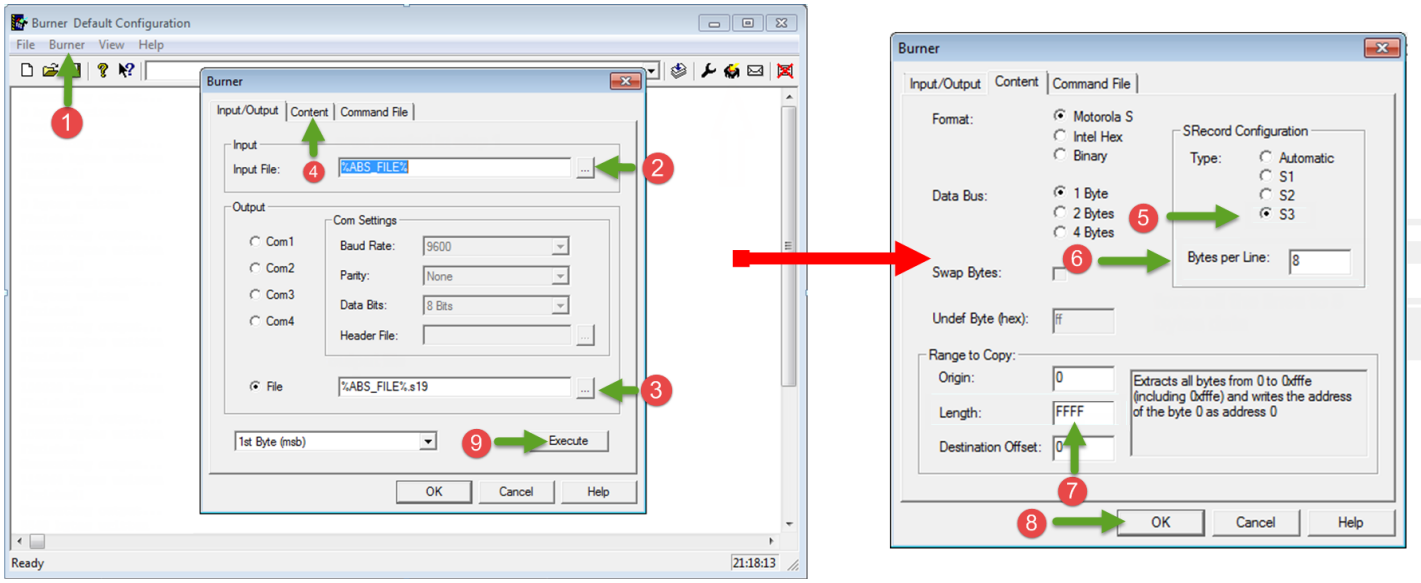

Figure 5. Burner Process

The generated file is the one that will be downloaded to the USB MSD device and has to have the name image.S19.

NOTE:

Since the K70 supports only 8 bytes writings the S19 should have all the lines forced to 8 bytes of data. This was not changed in the migration process in order to use the parser's philosophy of the original source code. This characteristic can be changed if the S19 parser code in the Loader.c file is changed with an enhanced code, this will depends on the programmer's ability.

5 Referenced Documents

- USB Mass Storage Device Host Bootloader. AN4368
- USB DFU Bootloader for MCUs. AN4370