DICE II EVM

Procedures for Updating Boot Monitor, setup Initialization File, and DICE Application Firmware

CONTENTS

What's New?

This document will be familiar to those who have used the EVM previously. The significant change here is that the **setup** file in the fis file system has been moved ahead of the application file. This allows the application file to grow in size without running into the setup file. This is a common case especially when using an AV/C target firmware build that has a lot of debug features compiled in.

Overview

This document describes how to update DICEII EVM firmware with the Boot loader and Application code. These instructions assume you have target hardware that is already loaded with a RedBoot boot loader/debug monitor. If you are bringing up a board that doesn't already have a working RedBoot boot loader, you will need to use a JTAG probe for your initial bring-up (or disaster recovery). In that case, see the document ***EVM_Bring-Up.pdf*** for more information.

A complete EVM file system will typically have 5 items as shown below. The **RedBoot**, **RedBoot config** and **FIS directory** files are created when you download a RedBoot image and then run **fis init -f**. The **setup** file is added for device serialization and run-time environment variables. This file must always be called "**setup**." The **dice** file, could be any name, as long as it is the same as used by any auto-run scripts (see below).

In general, your file system should look like the list below.

```
RedBoot> fis list
Name              FLASH addr  Mem addr    Length      Entry point
RedBoot           0x04000000  0x04000000  0x00020000  0x00000000
RedBoot config    0x041E0000  0x041E0000  0x00001000  0x00000000
FIS directory     0x041F0000  0x041F0000  0x00010000  0x00000000
setup             0x04020000  0x00000000  0x00010000  0x00000000
dice              0x04030000  0x00030000  0x00080000  0x00030040
```

To decide when to update various files on your EVM, use the following information:

Update RedBoot

If you have a DICE RedBoot version older than July 2005, you should update it as described below to make use of the latest features. Developers who are making changes to the RedBoot boot monitor kernel will also wish to do this. Follow the steps in Section 2 below.

*For those who have used previous revisions of the EVM, note that the **setup** file is now placed before the **dice** application file. This allows your application firmware to grow without the setup file getting in the way.*

Update Setup and Application

If your **setup** file is placed after the application file, it is a good idea to move it and reload the application image. Section 3 below explains the steps required. If you already have moved your setup file and you are simply updating your application image, follow the steps in Section 4.

RedBoot Update

The sequence for updating the RedBoot boot loader is as follows:

Connect to the EVM with a Serial Terminal Program

1) Connect a serial cable to the DB9 connector J16. This is the Primary serial port using a straight-through (1:1) cable. Also connect the cable to a computer with a native serial port.
2) Open a serial terminal program and set it to
   115200, 8, N, 1, No Flow control.
3) Power the EVM.

If you see a prompt at this point, the device already has firmware loaded and it is possible to update the device firmware using a serial terminal.

If you do not see a prompt on the Primary Serial port of the EVM, check your terminal communications settings, and make sure you are using a straight-through serial cable.

4) Make sure you are seeing a RedBoot prompt.

In order to update RedBoot, the EVM must be at the "**RebBoot>**" prompt and not running the application prompt, "**>**". If you see the application prompt, then do the following:

a) Get to a **RebBoot>** prompt by hitting **Ctrl+C** directly after resetting the EVM.
b) Type "**fco**"
c) When prompted, replace "**true**" with "**false**"
d) Enter "**y**" to confirm the change.
e) Reset the board
f) Continue with the steps below

Load the ROM boot image via Serial Terminal and write it to FLASH

5) At the prompt enter "**load –r –m ymodem –b 0x30000**"
6) Start the transfer of ROM version of RedBoot (using "xmodem 1k")
   The file name is **redbootrom.bin**

The boot image is now in RAM. Next we write this to FLASH memory.

7) Enter "**fis write –b 0x30000 –l 0x20000 –f 0x4000000**"

Make sure to enter the flash address correctly (-f parameter). That's a 4 with 6 zero's. Otherwise you will have to reinitialize your board flash memory using a JTAG interface. Also make sure to write the correct memory offset from RAM to flash, i.e. use the same address in the –b parameter for the load and the write commands.

8) Reset the EVM to boot from the flash image.

You should now see a **RedBoot>** prompt again.

```
RedBoot(tm) bootstrap and debug environment [ROMRAM]
Non-certified release, version v2_0 - built 17:15:02, Oct 26 2005

Platform: TCAT DICE/VB (ARM7TDMI)
Copyright (C) 2000, 2001, 2002, Red Hat, Inc.

RAM: 0x00000000-0x00800000, 0x00016528-0x007ed000 available
RedBoot>
```

setup file and Serialization

If you are merely updating your board firmware, and your setup file resides before the application file, you do not need to do the steps in this section. If you have accidentally deleted or overwritten your setup flash file, you can use this section to recreate it.

1) Type "**fis init -f**" to initialize the flash

Serialize the EVM and set the clocking parameter.

Create a "setup" flash file containing the lower byes of a WWUID for the device. *The upper bytes in the EVM are set to a temporary number to be used only for evaluation, development and test. Shipping products must contain the OUI of the manufacturer.*

2) Create a text file containing the setup file contents. For example, create (or edit an existing) **config.txt**

The file should look similar this (including the last line break):

```
SERIAL_NO=54321
HPLL_CLK=50000000
```

When the application runs, it will look in this file to set up any necessary environment variables needed at run time. You can see the current environment files in a running application by entering:

> envcfg.getall

[Note that this is entered at the application prompt, not the RedBoot prompt.]

SERIAL_NO is the lower 5 digits from the yellow serial number sticker on the EVM. This is used to calculate the lower bytes of the device WWUID, and guarantees that your device will always have a unique WWUID on any 1394 bus used during development and test.

HPLL_CLK the JET-PLL is running off of the DICE crystal (**Y1**) and in order for the sample counter to be correct it needs to know which crystal frequency is used. The default is 25 MHz, but it can be overwritten with the HPLL_CLK parameter. The frequency should be two times the crystal used. In case of a 24.576MHz crystal it should be stated as: **HPLL_CLK=49152000**

Load config.txt to it to RAM, and wrie it to the setup flash file.

3) Enter "**load -r –m ymodem –b 0x30000**"
4) Transfer **config.txt** (using "xmodem 1k")
   Create the setup sector and copy the file to it
5) Enter "**fis create –b 0x30000 –l 0x40 -e 0x0 –r 0x0 setup**"
   The **–l** length parameter should be at least the length returned by the load command (end – start +1). **0x40** is sufficient here, and RedBoot will round up to the nearest sector size anyway.

You can confirm that the setup section is written to flash by examining the flash memory. If you have gone through the steps above, the **setup** section will be located in flash memory at **0x4090000**. If you have a different configuration, type "**fis list**" to see the starting address of your setup sector.

Confirm the setup sector contents

6) Enter "**x –b 0x04090000 –l 0x40 -1**"

This should show something similar to this, with an example SERIAL_NO of 54321:

```
RedBoot> x -b 0x4090000 -l 0x40 -1
04090000: 53 45 52 49 41 4C 5F 4E  4F 3D 35 34 33 32 31 0D  |SERIAL_NO=54321.|
04090010: 0A 48 50 4C 4C 5F 43 4C  4B 3D 35 30 30 30 30 30  |.HPLL_CLK=500000|
04090020: 30 30 0D 0A 54 01 03 00  78 01 03 00 A4 01 03 00  |00..T...x.......|
04090030: C0 01 03 00 00 00 00 00  C0 02 03 00 88 02 03 00  |................|
RedBoot>
```

Application Update

The sequence for updating the Application is as follows:

Load the Application image via Serial Terminal and write it to FLASH

Initialize the FLASH file system and load the **dice** application image to RAM

1) Enter "**load -r –m ymodem –b 0x30000**"
2) Transfer the application file (using "xmodem 1k")

The application file will have a **.bin** file extension, and will be something like **dice_3_0_1.bin**. If you are loading a file you have built yourself, it will be **dice.bin** in your output directory. Note that you will not be loading the "**dice**" file from that directory as it is the file used by the debugger.

Make a dice FLASH sector and copy the application from RAM into it.

3) "**fis create –b 0x30000 –l 0x70000 –e 0x30040 –r 0x30000 dice**"

If asked to overwrite the existing image, enter "**y**".

Auto-run

Configure the Board to automatically run the application image

4) At the **RedBoot>** prompt, type "**fco**"
5) Replace **false** with **true**
6) Enter the following two lines at the **>>** prompts, followed by a **return**:
   fis load dice
   go

7) Enter a value of 20 for the boot delay. This is equivalent to 2 seconds (the ms resolution indicated is not correct).
8) Enter '**y**' to confirm
9) Reset the board and verify the results

Final Checks

The EVM should run **RedBoot** and then auto-run the **dice** application at this point. Look at the splash screen. The **Board S/N** should be the same serial number entered in the **setup** section, from **config.txt**. ***Note that this value is displayed in hexadecimal in the "splash" CLI command.***

```
>splash

*********************************************************
* TC Command Line Interface System                      *
* Copyright 2003-2005 by TC Electronic A/S              *
*********************************************************
* Running Dice II 1394 Appl                             *
* Board S/N: 00002718, Appl Ver.: 2.12 RELEASE          *
*                       - built on 17:14:09, Oct 26 2005 *
*    MIDI is enabled.                                   *
*********************************************************
* Target: DICE II Evaluation Board                      *
* Driver: DiceDriver                                    *
*********************************************************
>
```

The serial number will also become the lower bytes of the WWUID for the device. *The upper bytes in the EVM are set to a temporary number to be used only for evaluation, development and test. Shipping products must contain the OUI of the manufacturer.*

```
>lal.getwwuid
WWUID: 0x0013f004 0x004 0c353
>
```

If you results are similar to this, then you have now successfully updated your DICEII EVM.

Firmware Load over 1394

Once you are using firmware Rev. 2.0.1 or greater, you can use the Windows XP Firmware Loader application to load new firmware to your DICE hardware.

See the relevant info in your DICE II Driver distribution for details. This should be included with your EVM or SDK.