Application Note

AN_159

Vinculum-II Firmware Flash Programming

Document Reference No.: FT_000358

Version 2.0

Issue Date: 2011-12-08

This document describes three ways to program and update firmware on the Vinculum-II (VNC2) USB host controller.

- Programming a ROM file via the debugger interface.
- Programming a ROM file over UART.
- Re-flashing a ROM file using a flash disc drive.

Future Technology Devices International Limited (FTDI)

Unit 1, 2 Seaward Place, Glasgow G41 1HH, United Kingdom

Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758

Web Site: http://ftdichip.com

Table of Contents

1 Introduction

The VNC2 device is a programmable SoC device with a powerful embedded microprocessor core and dual USB host or slave interfaces, large RAM and flash capacity and the ability to develop and customise firmware using the VNC2 Toolchain.

This application note describes how to program or re-flash a ROM file into the flash memory of the VNC2 device.

VNC2 firmware in a ROM file format can be programmed into the flash of the device by three methods:

- Programming a ROM file via the VNC2 debugger/programmer moduleinterface.
- Programming a ROM file over UART.
- Re-flashing a ROM file using a flash drive.

VNC2 devices are produced as blank devices. These blank devices can only be programmed over UART or the debug interface. Re-flashing or upgrading over USB can only be performed after an initial firmware is programmed to the VNC2.

VNC2 devices can also be programmed before being assembled in a system using the VPROG1 VNC stand alone programmer – see (www.ftdichip.com)

Note: Any sample code provided in this note is for illustration purposes and is not guaranteed or supported.

2 IDE Tool Suite

The Integrated Development Environment (IDE) is FTDI's Toolchain that allows the user to open and edit sample C programs from FTDI or develop their own C programs. The IDE is used to build or compile the C program into a ROM file. This ROM file is programmed to the flash of the VNC2 device. The IDE is a free download from FTDI web site with sample projects including two Vinculum-I (VNC1L) firmware versions that have been re-written for VNC2 (V2DAP and V2DPS).

2.1 IDE Tool Suite – Create a ROM file

C programs can be compiled using the IDE build as shown in Figure 2.1 IDE Build. In this example the V2DAP project is opened from the samples folder and is being built. Full details on opening a C project and building the ROM file are available in the getting started guide found within the IDE help pages (Figure 2.2 IDE Getting Started Guide).

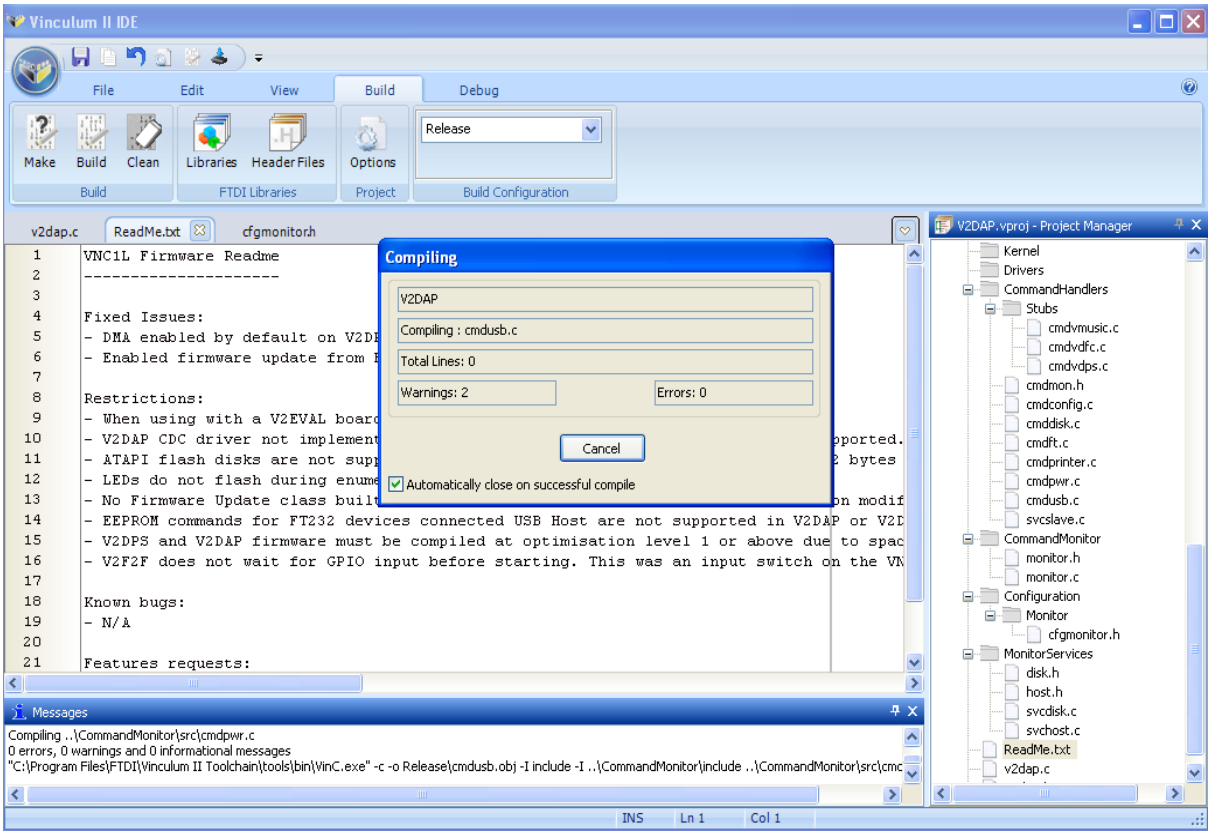


Figure 2.1 IDE Build

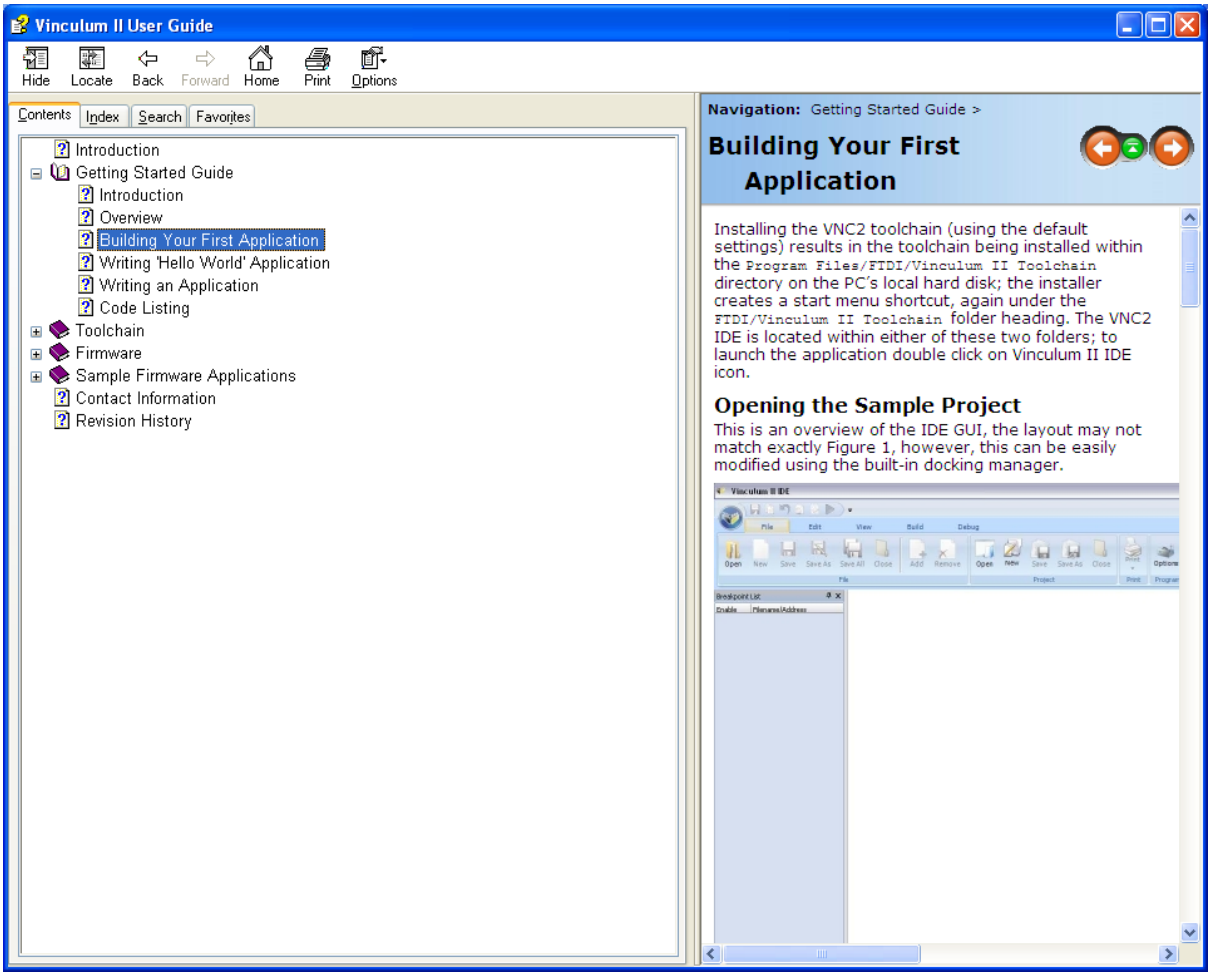


Figure 2.2 IDE Getting Started Guide

2.1.1 Debug or Release Build

The C code can be compiled as either a release or a debug build, the option is a pull down selection in the Build tab as shown is Figure 2.3. If further debug and testing of the code is required it is recommended to select the debug option. After an application has been successfully tested it is recommended to subsequently build the application in Release mode. This will have the effect of reducing the overall ROM file size but will now not allow any further debugging using the IDE.

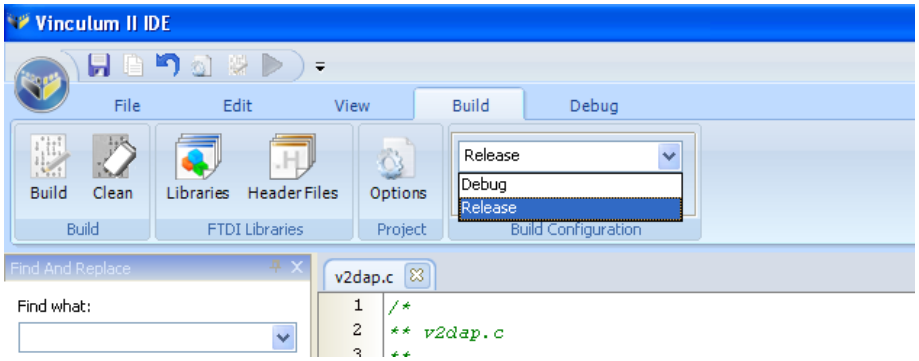


Figure 2.3 Debug or Release Build

2.1.2 Build Completion

On completion of a successful build of the code without errors, the IDE messages window will display:

[VinL.exe] : 0 errors, 0 warnings and 0 informational messages

The ROM file will be located within either the debug or release folders:

e.g. \My Documents\FTDI\Firmware\Samples\1.2.2-SP1\Firmware\VNC1L\V2DAP\Debug

3 Programming a ROM File via the Debugger Interface

The IDE is used to program the VNC2 with a ROM file for the first time (blank device) or to reprogram a previously installed firmware.

3.1 Hardware Requirements

There are several options to provide the debug interface:

- VNC2 debug port as defined in AN_138 Vinculum-II Debug Interface Description
  - Available on Vinco and V2DIP modules
  - Included on a customer target design
- VNC2 Debugger/Programmer Module
- USB A to mini-B cable

Note that the V2-EVAL includes the debug interface that connects directly to the host PC running the VNC2 Toolchain.

3.2 Hardware Example

Figure 3.1 shows an example of the V2DIP1-48 module connected to the Debugger/Programmer module. A USB A to Mini-B Cable is then required to connect the debugger/programmer to the PC.

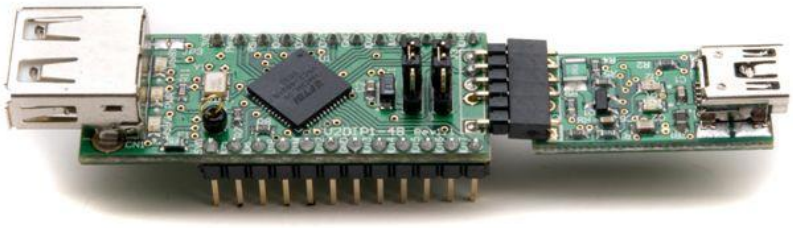
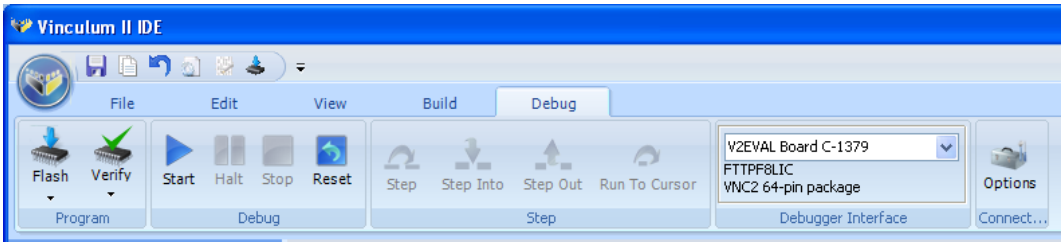


Figure 3.1 V2DIP1 connected to the VNC2 Debugger/Programmer Module

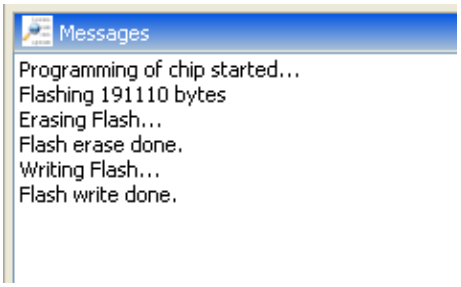
3.3 Procedure – Program Using IDE

When a ROM file has been successfully generated, as described within section 2.1, it can be programmed into the flash of the VNC2.

- Connect the Debugger/Programmer module into V2DIP1 module's debug port.
- Connect a USB A to Mini-B Cable from the PC to the Mini-B connector on the Debugger/Programmer module.
- In the IDE Debug tab, the VNC2 that is connected will be shown in the interface area. The "Flash" button can be pressed to start the programming the most recently compiled ROM file. A different ROM file can be selected by clicking the small down-arrow below "Flash".



- Completion of the programming will be display within the messages window.



The "Verify" button may be used to check the contents of the flash. This requires Prog_Loader V1.7 or above. If this version of Prog_loader is not in the flash then the message "Cannot verify flash contents! Use latest version of the program loader." will be displayed. Contact support for more information.

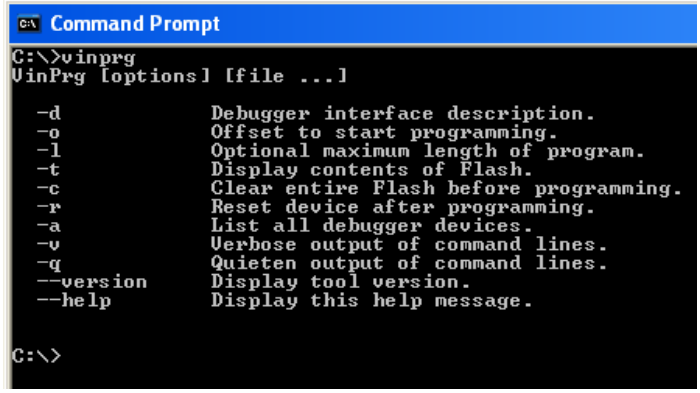
3.4 Procedure – Program using VINPRG.exe

The ROM file can be programmed to the VNC2 using the command line programming tool – VINPRG.exe.

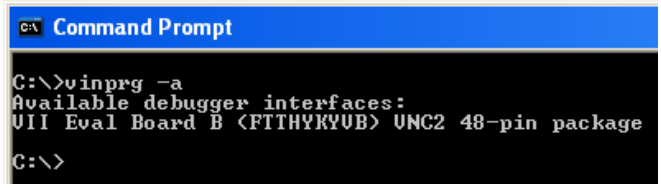
The tool is installed with the IDE setup file and is located at:

C:\Program Files\FTDI\Vinculum II Toolchain\Tools\bin\VinPrg.exe

- Open a command prompt and type vinprg to view the available commands.

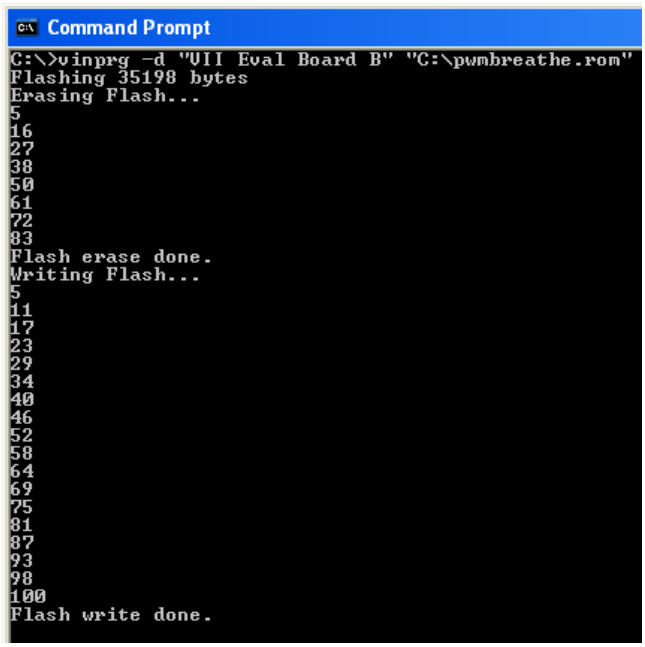


- Perform the vinprg –a command to determine what hardware is connected to the debugger module. In this case, the debug interface is identified as "VII Eval Board B"



- Perform the vindprg -d command to program the ROM file. In this example the file C:\pwmbreathe.rom is programmed to the VII Eval Board via channel B.

  vinprg -d "VII Eval Board B" "C:\pwmbreathe.rom"



4 Programming a ROM file over UART

A blank VNC2 can be programmed over UART using the FTDI utility FT_Prog with a ROM file. FT_Prog can also be use to upgrade previously installed firmware. VNC2 support requires FT_Prog version 1.12 or newer.

4.1 Hardware Requirements

VNC2 module or customers designed VNC2 PCB with UART pins connected to a PC via either a TTL-232R-3V3 cable or customers designed serial bridge.

4.2 Hardware Example

Figure 4.1 is an example where an FTDI TTL-232R-3V3 cable is used to program a V2DIP1 module over UART. To enable the bootloader mode, the PROG# pin must be driven low and VNC2 must then be reset by driving the RESET# pin low then high. The UART connections of the cable are connected to the default UART pins on the module. In this case, a short was made by connecting PROG# to GND prior to applying power to the module. This is shown at the bottom right of Figure 4.1

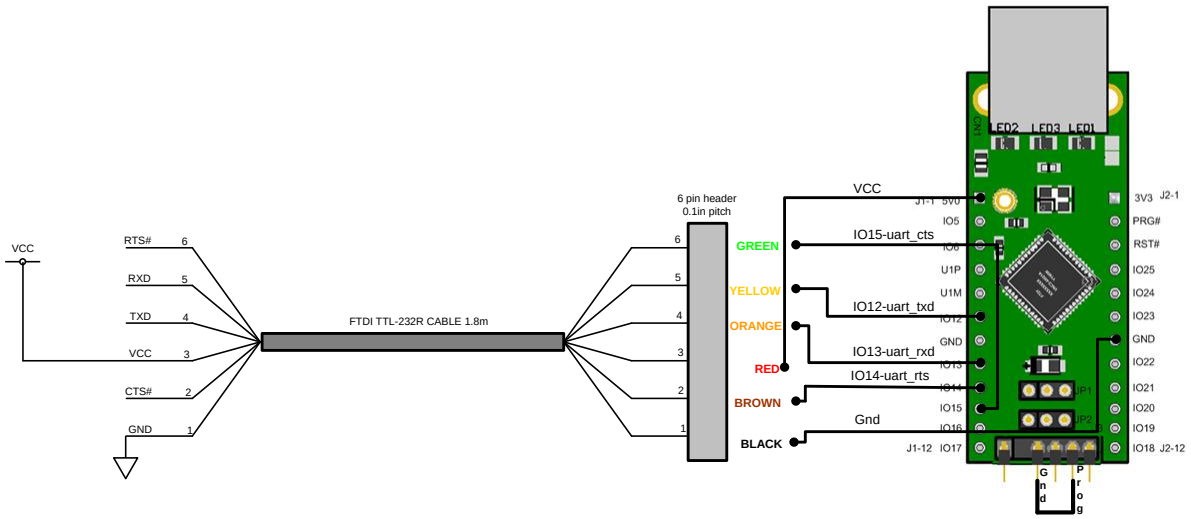


Figure 4.1 TTL-232R-3V3 Cable to V2DIP1-48 Module via UART

4.3 Procedure

FT_Prog is used to program the VNC2 with a ROM file. FT_Prog is available from the FTDI website utilities page (version 1.12 or later supports VNC2).

- Select the flash ROM tab at the top of the window.
- Select VNC2 from the pull down tab.
- Select interface.
- Select the location where the ROM file resides.
- Press the program button.
- Perform a hard reset (power cycle) prior to running the firmware.

Figure 4.2 is an example of programming the VNC2 Evaluation board revision 2 with the V2DAP firmware.

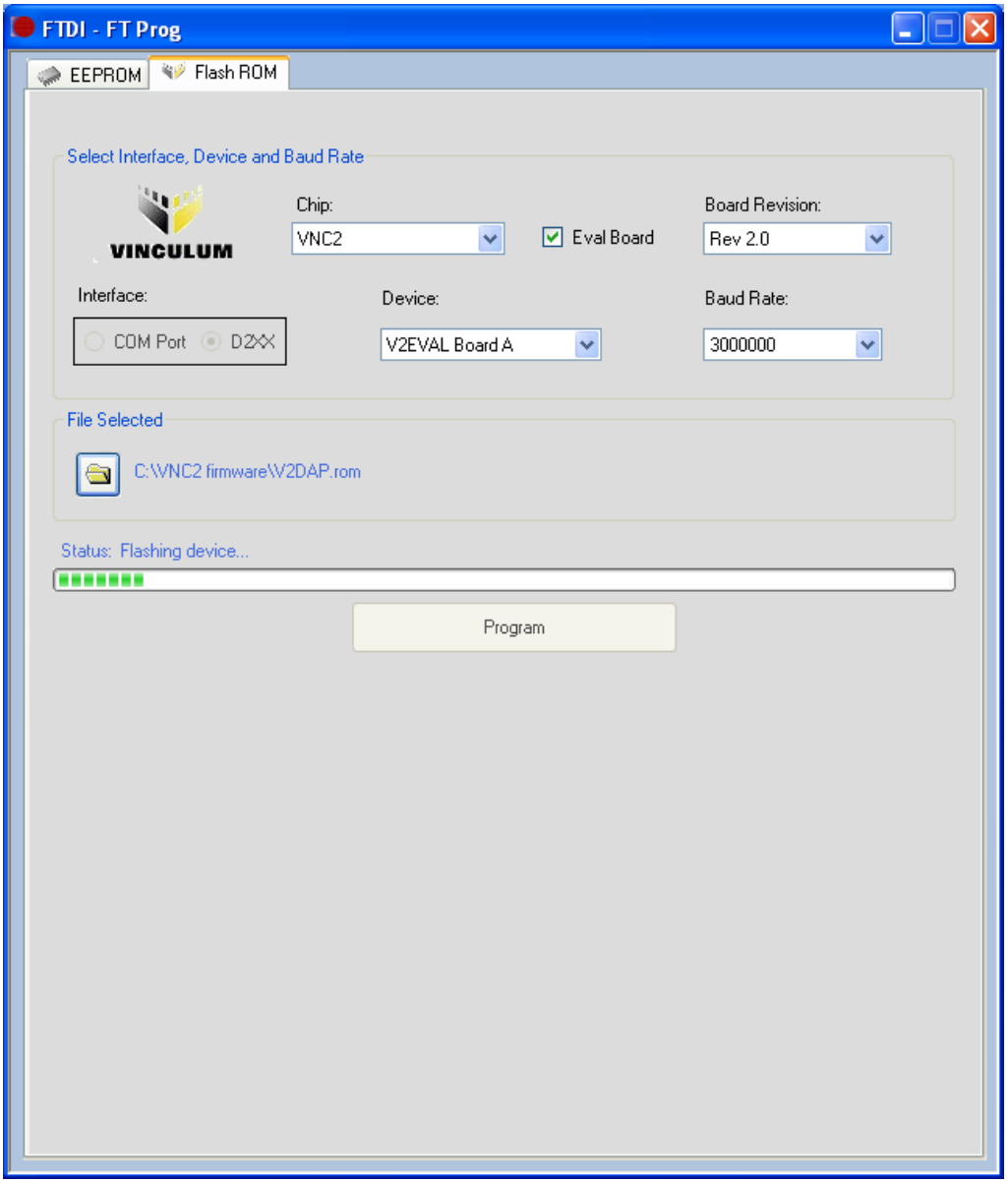


Figure 4.2 FT_Prog Programming Utility

5 Re-flashing a ROM File Using a Flash Drive

The VNC2 is distributed as a blank device which can be programmed with an application by the two methods (UART or debug interface) previously described in this document.

The VNC2 device can be programmed with re-flasher firmware that allows subsequent application firmware to be upgraded from a USB memory stick. The re-flasher firmware code is separate from the application code and is not overwritten when the application firmware is update.

The re-flasher firmware must be programmed into the VNC2 flash over UART or debug interface .This firmware is programmed into an area of flash of the VNC2 that will not be overwritten by the user firmware.

To permit the firmware update an application needs to include a Firmware Update library which can read a file from a FAT format disk and instruct the re-flasher firmware to program that into the flash memory of the VNC2.

The "FirmwareUpdate" sample in the General sample section provides an example of how to create an application that will find an update file on a USB memory stick and program that into flash.

The source code for the application is provided as an example and is neither guaranteed nor supported by FTDI.

Currently the only re-flasher firmware is ReflashFATFile.rom which will program an application ROM file from a FAT format disk.

5.1 Hardware Requirements

- VNC2 module or customers designed PCB with USB Type A connector on a USB channel of the VNC2 chip configured as host.
- Memory stick / Flash Drive formatted in a supported file system (FAT16 or FAT32)

5.2 Software Requirements

- VNC2 IDE
- Command prompt – cmd.exe (In windows select Start – Run cmd.exe)
- V2Eval Board Terminal
- Vinprg – included within IDE (C:\Program Files\FTDI\Vinculum II Toolchain\Tools\bin)

5.3 Principles

The principle of operation is that the application selects a file on a disk which contains a valid ROM image. The file must be accessible by the FAT file system driver or API and must be opened by the application before it calls the Firmware Update library.

The Firmware Update library will validate the ROM file before calling the re-flasher firmware. The re-flasher firmware is stored at the end the flash memory on the VNC2. This essentially forms a second application on the VNC2 which performs the actual update of the flash ROM on the VNC2 and overwrites the main application.

The Firmware Update library is called with a handle to a FAT file and the address of the Re-flasher code. There is an option to instruct the reflasher firmware to provide feedback through the UART or a GPIO pin.

During a firmware update the re-flasher code can output a dot trail to UART or toggle a GPIO line.

A fail will be a constant low GPIO line or an exclamation mark (!) on the UART.

A pass will result in the GPIO line set high or a carriage return on UART.

These options are set when calling the Firmware Update library. The parameters for the feedback options are shown in the FirmwareUpdate.h file. They can be bitwise ORed together to have UART and multiple GPIO pins provide feedback.

When complete, the CPU will reset and run the new program.

5.4 Components

The Re-flasher code is supplied as both an object file which must be linked to an address that will not be overwritten by the main application, and as a ROM file which is linked to address 0x1c000. The recommended location of the re-flasher firmware is 0x1f000 which is 8kB from the top of flash, leaving 248kB of flash available for the main application.

Addresses of data in flash are all word addresses.

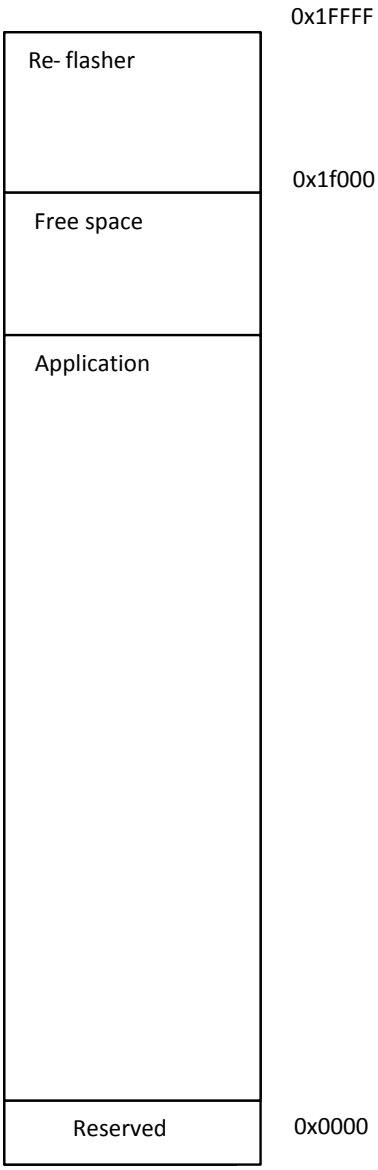
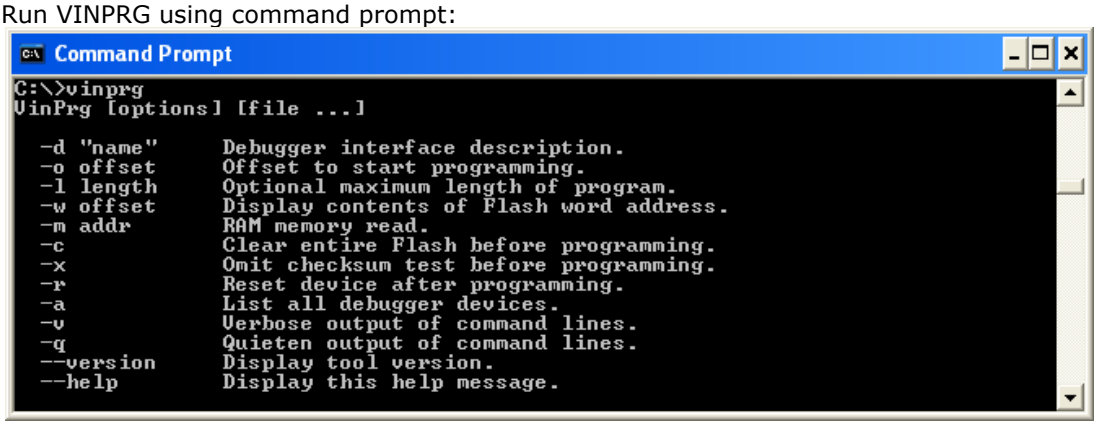


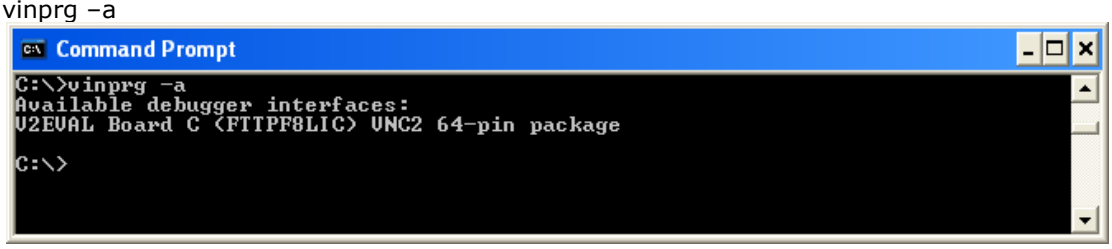
Figure 5.1 Flash Memory

5.5 Firmware Update Sample Procedure

- VINPRG.exe is used to program the Re-flasher ROM code to an area within flash that is not used by the user application.
- Run VINPRG using command prompt:



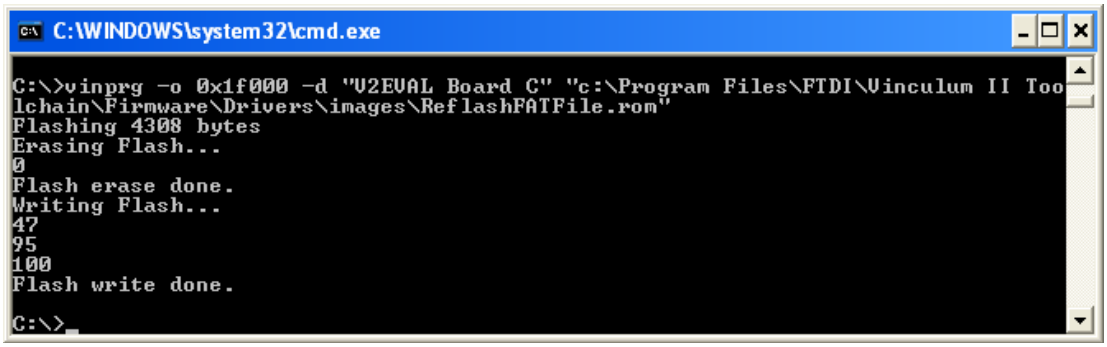
- Firstly detect the debugger interface name using the list all debugger interfaces vinprg –a



- Use the debugger interface name within the following command:

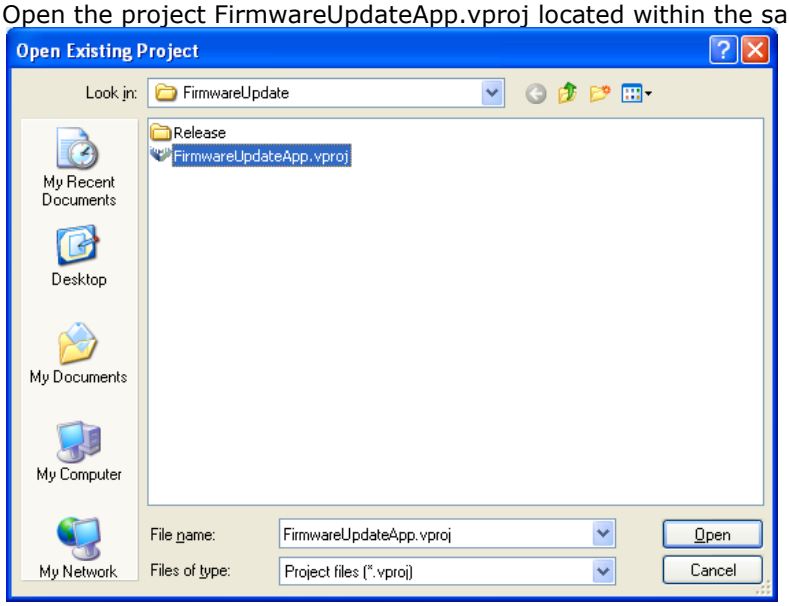
vinprg -o 0x1f000 -d "V2EVAL Board C" "c:\Program Files\FTDI\Vinculum II Toolchain\Firmware\Drivers\images\ReflashFATFile.rom"

Parameters:	-o 0x1f000	offset to start programming at 0x1f000
	-d " V2EVAL Board C "	debugger interface description

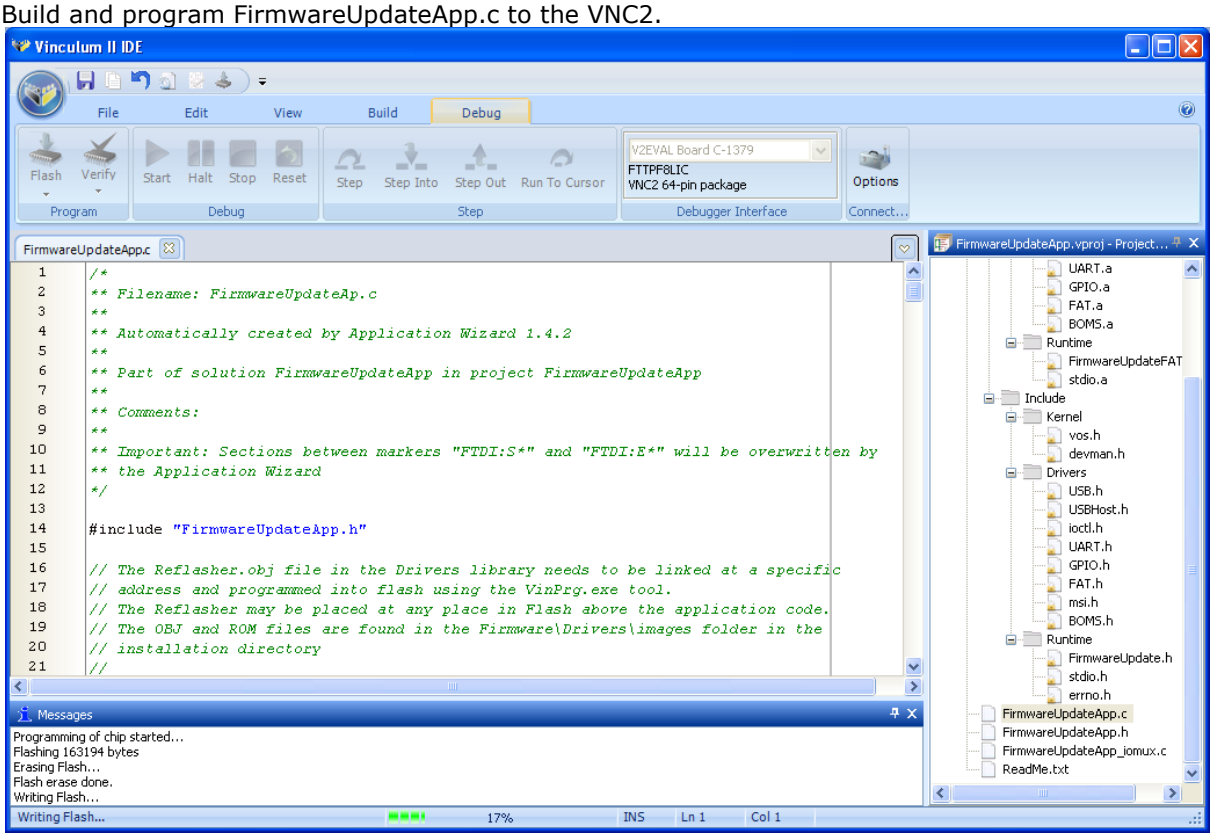


5.5.1 Program FirmwareUpdate

- Open the project FirmwareUpdateApp.vproj located within the samples section of the IDE.



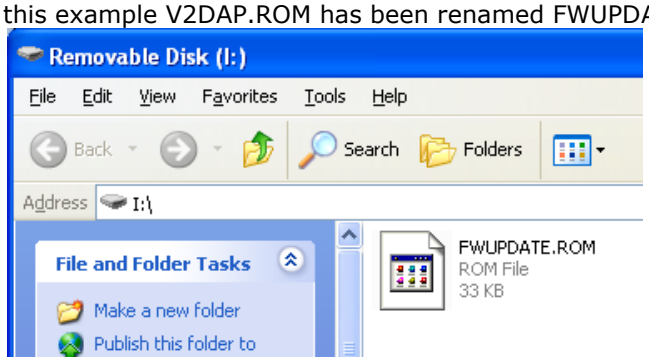
- Build and program FirmwareUpdateApp.c to the VNC2.



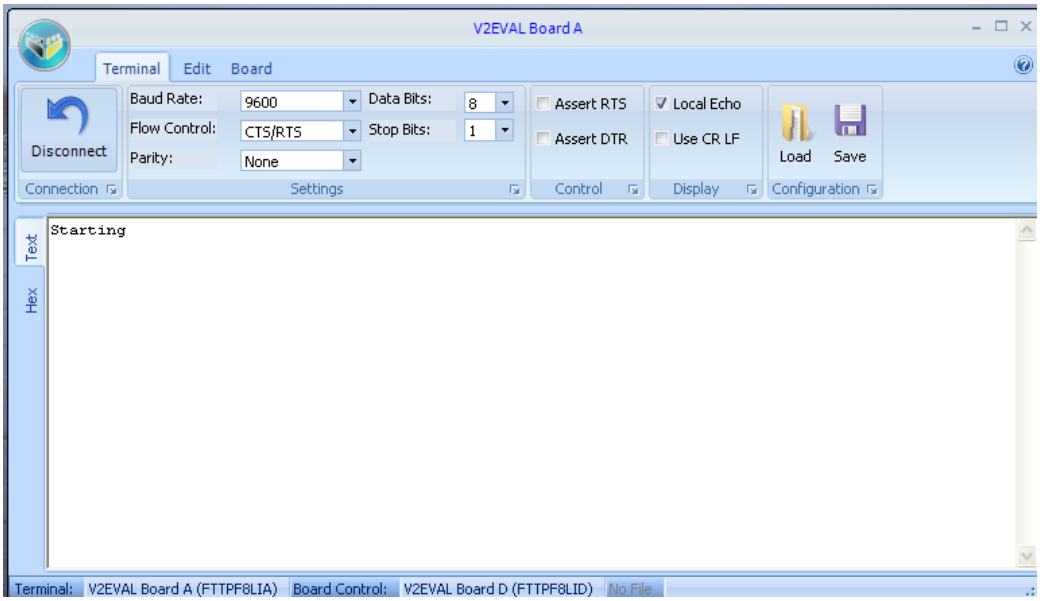
5.5.2 Store User Firmware on USB Memory Stick

- Select the ROM file to be programmed and put it onto the flash disk and rename it as "FWUPDATE.ROM". This can be the ROM file of any of the samples, however, if the ROM file for this sample (FirmwareUpdate) is used then the update will run over and over. In

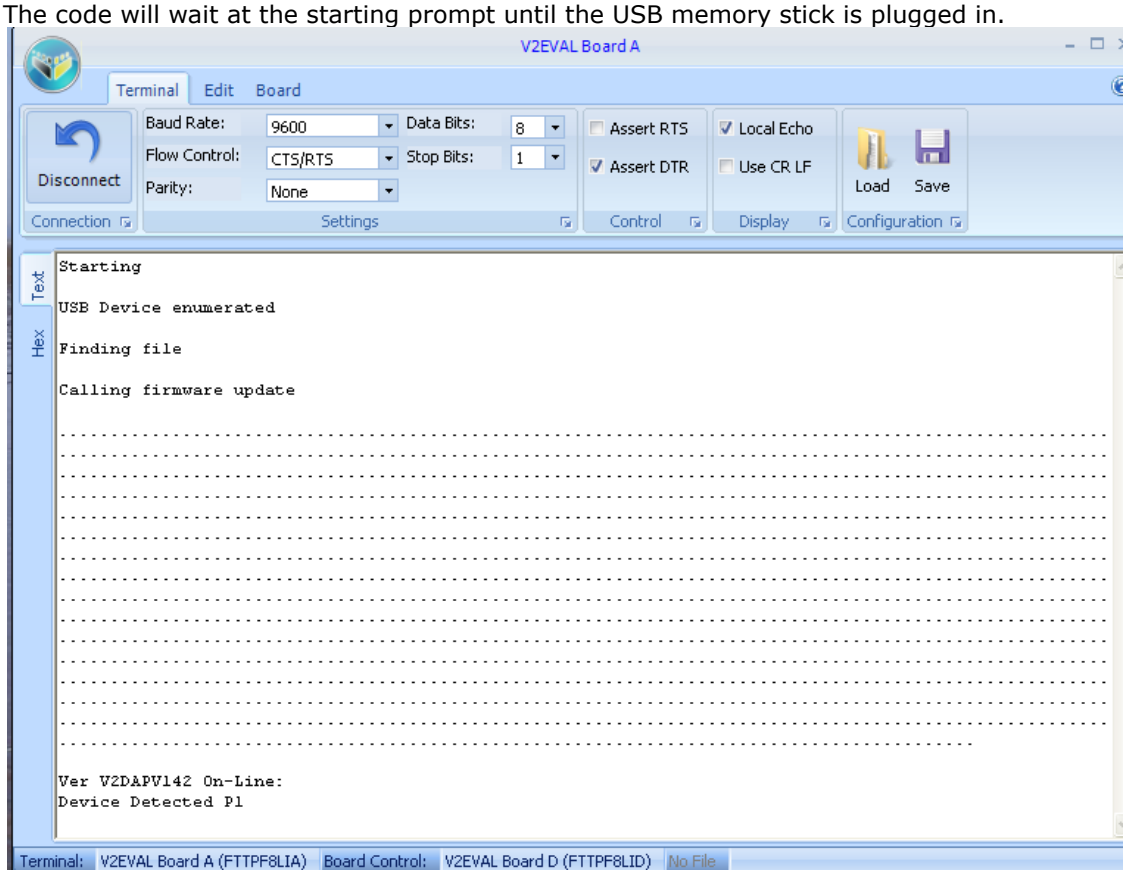
this example V2DAP.ROM has been renamed FWUPDATE.ROM.



- Open V2Eval Board Terminal (or similar e.g. TTY, HyperTerminal), and press reset on the board.



- The code will wait at the starting prompt until the USB memory stick is plugged in.



- During programming, a trail of dots will be sent to the UART interface and (if using a V2EVAL board) LED0 will flash.
- When programming is complete, a carriage return ("\r") will be sent to the UART and the LED will be turned off.
- If an error occurs then an exclamation mark will be sent to the UART and the LED will be set on permanently.
- After successful programming the VNC2 will be reset and run the updated code. In the example shown the V2DAP prompt can now be observed.

5.6 Building the Flash Update Into a Custom Design

A custom design must include two key components in order to allow future flash updates (e.g. updating ROM in the field using the Flash disk).

- FAT file system and required libraries, typically BOMS and USBHost.
- Firmware Update library

These libraries must be included by the App Wizard or by using the Manage Libraries and Manage Headers method.

The "Hello World" application example available within the VNC2 IDE was edited to describe an example of building the flash update into a VNC2 project.

The "GPIOKitt" application example also found in VNC2 IDE was used as updated project.

5.6.1 FAT File System

The FAT file system must be initialised and able to access the disk where the firmware update ROM file is stored. The stdio library, FAT file system driver or FAT file system API may be used to open the firmware update ROM file in the application before calling the firmware update library.

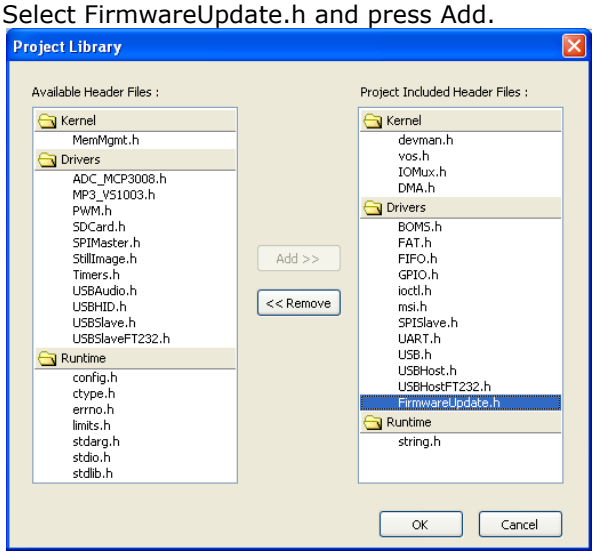
5.6.2 Firmware Update Library

The header file for the firmware update library must be included and the location (in ROM) of the firmware re-flasher noted.

The "Hello World" application example was edited as follows:

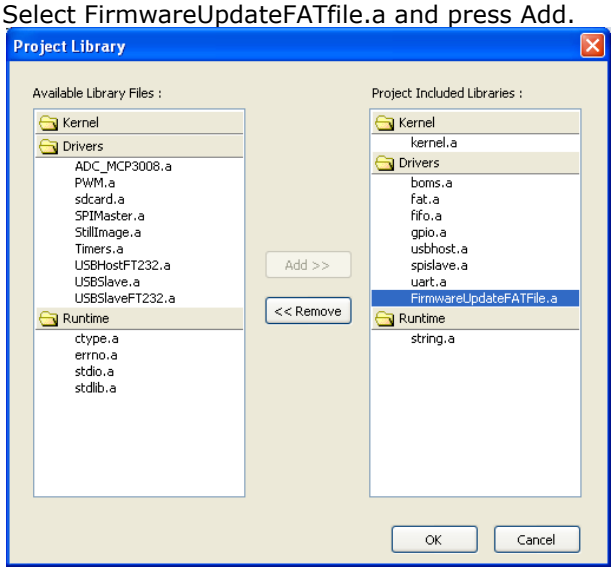
- Right click the Project Library window and select manage header files.

- Select FirmwareUpdate.h and press Add.



- Right click the Project Library window and select manage libraries.

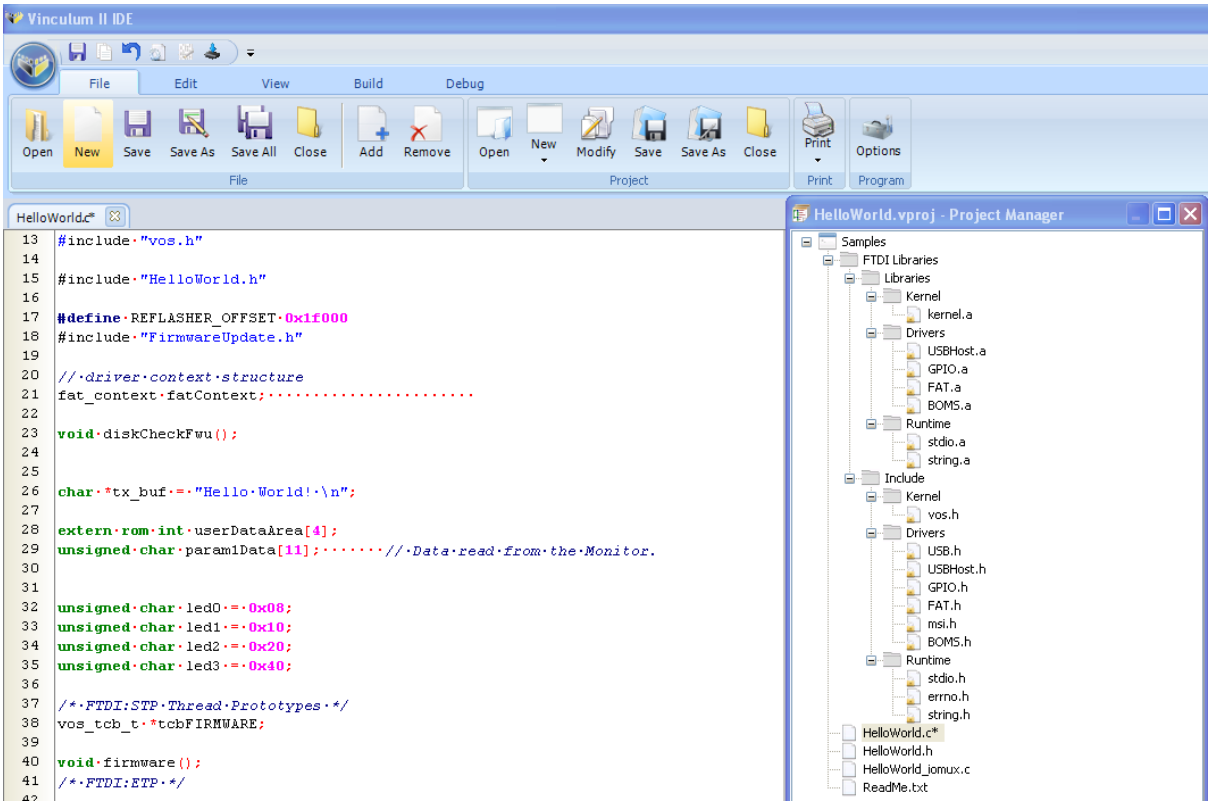
- Select FirmwareUpdateFATfile.a and press Add.



- Edit HelloWord.c

#define REFLASHER_OFFSET 0x1f000

#include "FirmwareUpdate.h"



5.6.3 "Hello World" – Firmware Update File Code Example

Firmware Update File code was added into the "Helloworld.c" file to check a disk for a firmware update.

```
//Global Variables
extern rom int userDataArea[4];
unsigned char param1Data[11];

** Checks a disk for a firmware update file.
** Parameters: param1: filename of update
** Returns: none
** Comments: Checks for BOMS disk and valid FAT file system*/

void diskCheckFwu()

{
        char *strChangeMain = "Change MAIN\r";
        unsigned char status;
        int verFile, verROM;
        FILE *file;
        // check for firmware update file: "FTRFBV2.FTD"
        file = fopen((char *) param1Data, "r");
        if (file != NULL)
        {
                status = fseek(file, 0x20, SEEK_SET);
                if (status == 0)
                {
                        status = fread(&verFile, 4,1, file);
                        if (status == 4)
                        {
                                verROM = userDataArea[0];
                                verROM |= userDataArea[1] << 8;
                                verROM |= userDataArea[2] << 16;
                                verROM |= userDataArea[3] << 24;
                                if (verFile != verROM)
                                {
                                        status = fseek(file, 0, SEEK_SET);
                                        status = FirmwareUpdateFATFileFeedback(file, 0x1f000,
                                        FIRMWARE_UPDATE_FEEDBACK_UART);
                                }
                        }
                }
                fclose(file);
        }
}
```

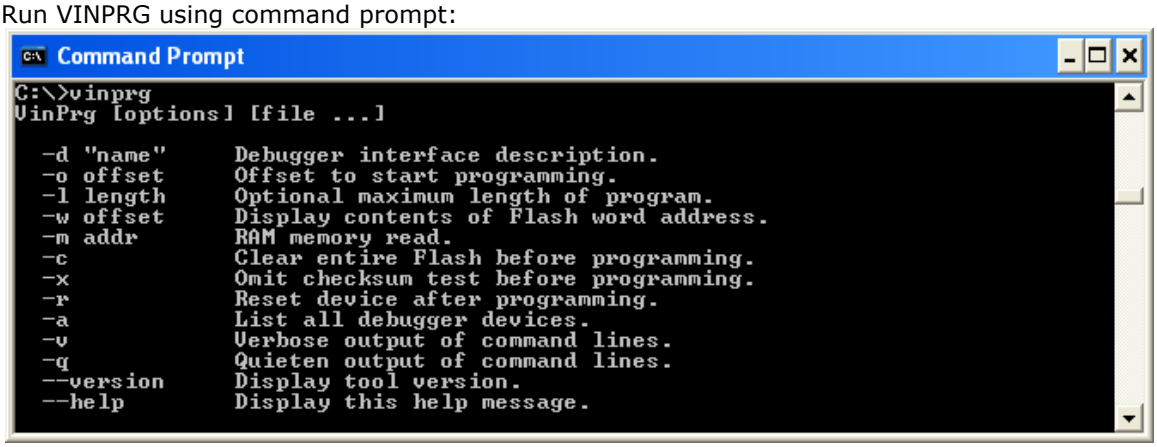
The function "diskCheckFwu()" was then called from the "firmware ()" function as shown below:

```
char *updateFile = "FTRFBV2.FTD";
vos_memcpy(param1Data, updateFile, 11);
diskCheckFwu();
```

5.6.4 "Hello World" Sample – Re-Flash Steps

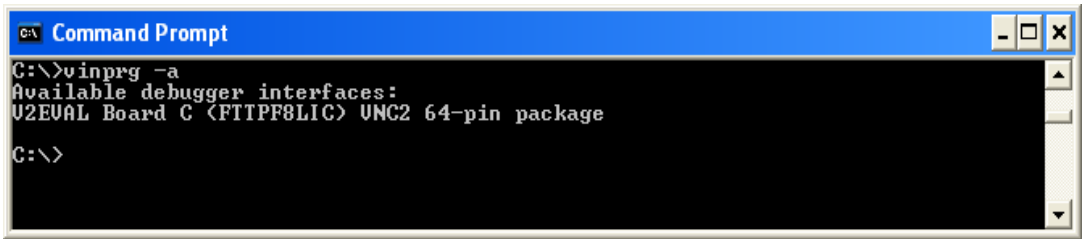
VINPRG.exe is used to program the Re-flasher ROM code to an area within flash that is not used by the user application.

- Run VINPRG using command prompt:



- Firstly detect the debugger interface name using the list all debugger interfaces

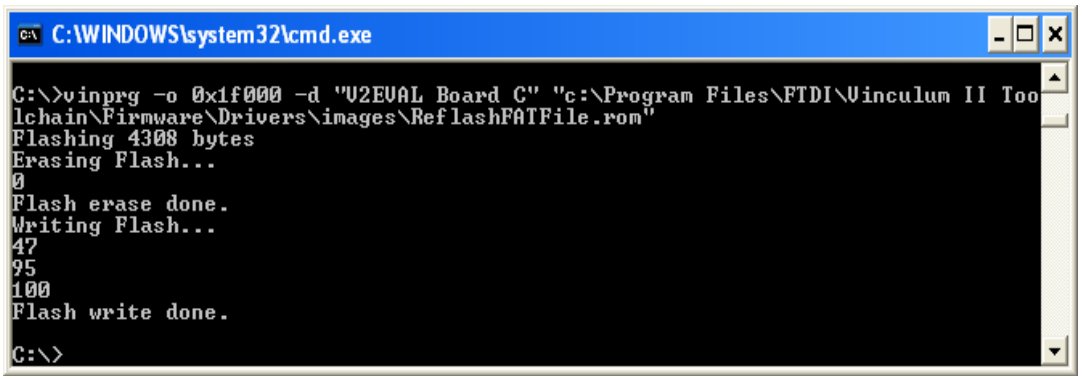
  vinprg –a



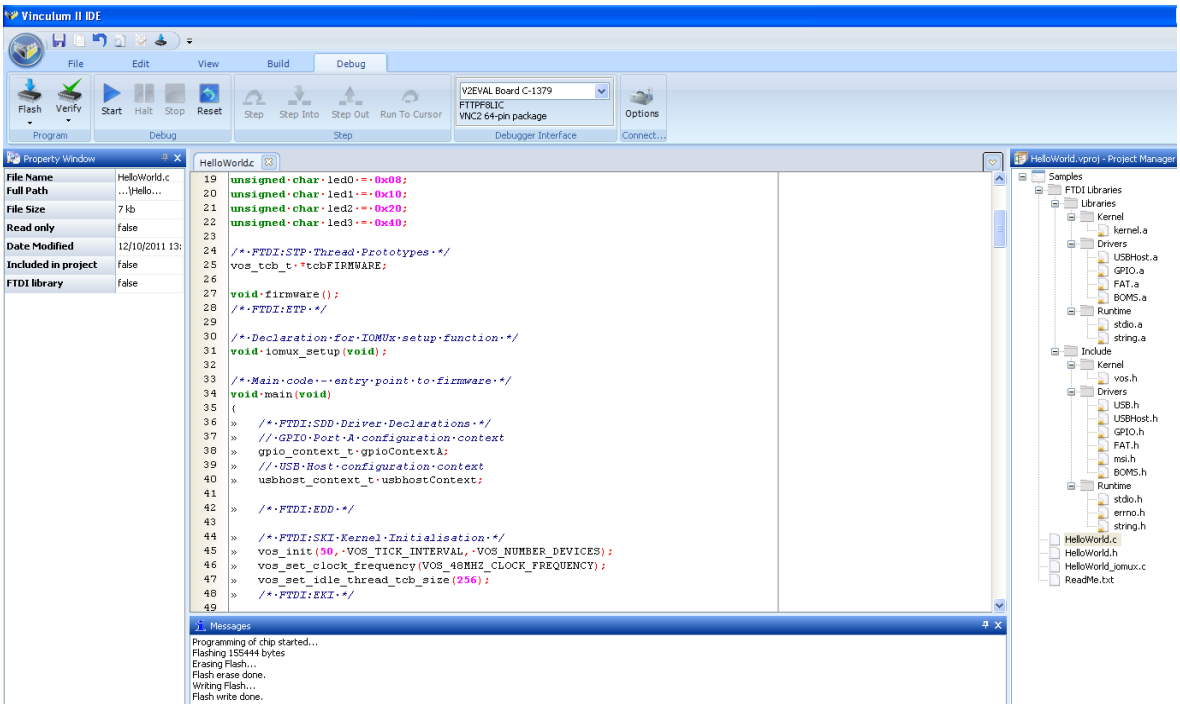
- Use the debugger interface name within the following command:

  vinprg -o 0x1f000 -d "V2EVAL Board C" "c:\Program Files\FTDI\Vinculum II Toolchain\Firmware\Drivers\images\ReflashFATFile.rom"

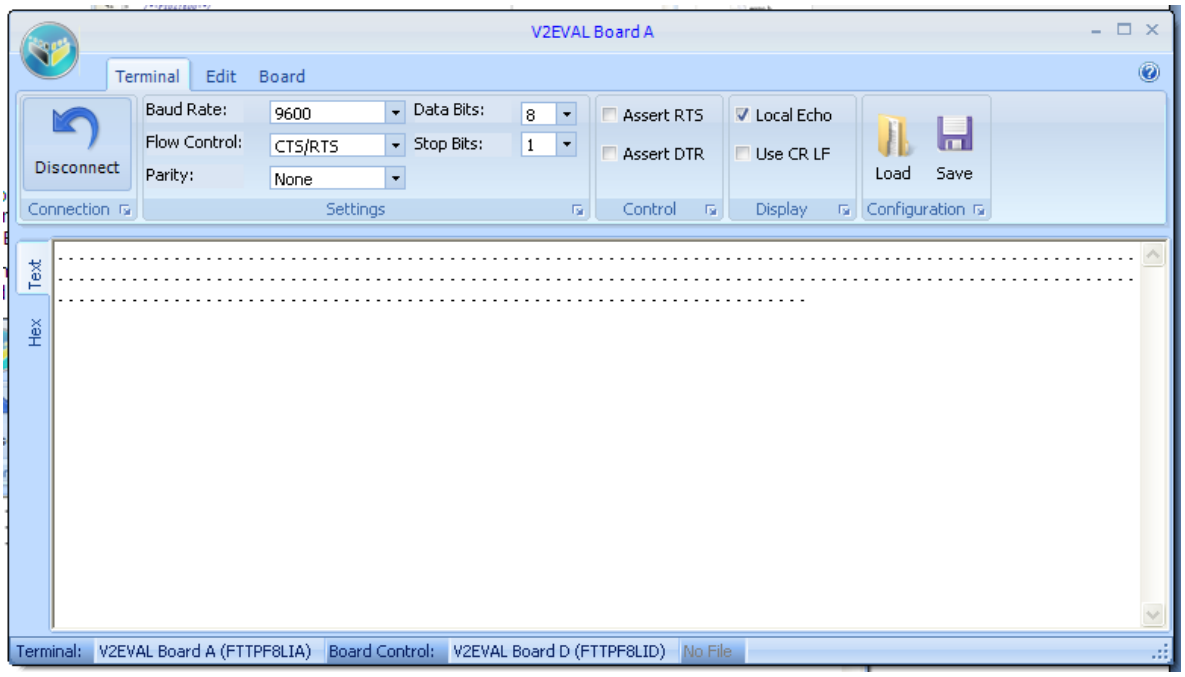
  Parameters: -o 0x1f000 offset to start programming at 0x1f000
  -d " V2EVAL Board C " debugger interface description



- Use the IDE to program "Hello World" application to the chip



- Choose a ROM file that you want to flash to the VNC2, put it onto the flash disk and rename it as "FTRFBV2.FTD". In this example GPIOKitt.ROM has been renamed FTRFBV2.FTD
- Open a COM port terminal program e.g. TTY, HyperTerminal and press reset on the board. In this example the V2-EVAL Terminal Utility was used.
- The code will wait at the starting prompt until the USB memory stick is plugged in.



- During programming, a trail of dots will be sent to the UART interface and (if using a V2EVAL board) LED0 will flash.
- A successful update will take approximately 10 to 20 seconds after which activity on the flash disk will cease and the dots will end. The VNC2 CPU will be halted upon completion so a power cycle is required before the new firmware will be activated.

6 Contact Information

Head Office – Glasgow, UK

Future Technology Devices International Limited
Unit 1, 2 Seaward Place, Centurion Business Park
Glasgow G41 1HH
United Kingdom
Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758

E-mail (Sales) sales1@ftdichip.com
E-mail (Support) support1@ftdichip.com
E-mail (General Enquiries) admin1@ftdichip.com

Branch Office – Hillsboro, Oregon, USA

Future Technology Devices International Limited
(USA)
7235 NW Evergreen Parkway, Suite 600
Hillsboro, OR 97123-5803
USA
Tel: +1 (503) 547 0988
Fax: +1 (503) 547 0987

E-Mail (Sales) us.sales@ftdichip.com
E-Mail (Support) us.support@ftdichip.com
E-Mail (General Enquiries) us.admin@ftdichip.com

Branch Office – Taipei, Taiwan

Future Technology Devices International Limited
(Taiwan)
2F, No. 516, Sec. 1, NeiHu Road
Taipei 114
Taiwan , R.O.C.
Tel: +886 (0) 2 8791 3570
Fax: +886 (0) 2 8791 3576

E-mail (Sales) tw.sales1@ftdichip.com
E-mail (Support) tw.support1@ftdichip.com
E-mail (General Enquiries) tw.admin1@ftdichip.com

Branch Office – Shanghai, China

Future Technology Devices International Limited
(China)
Room 408, 317 Xianxia Road,
Shanghai, 200051
China
Tel: +86 21 62351596
Fax: +86 21 62351595

E-mail (Sales) cn.sales@ftdichip.com
E-mail (Support) cn.support@ftdichip.com
E-mail (General Enquiries) cn.admin@ftdichip.com

Web Site

http://ftdichip.com

Appendix A – References

Document References

VNC2 Data sheet

Vinculum-II Toolchain

Vinculum-II Tool Chain Getting Started Guide

V2DIP1-48 Data sheet

Vinculum-II Debug Interface Description

Migrating Vinculum Designs From VNC1L to VNC2-48L1A

Acronyms and Abbreviations

Terms	Description
FAT	File allocation Table
FT_Prog	FTDI Programming application.
IDE	Integrated Development Environment
SoC	System on chip
USB	Universal Serial Bus
USB-IF	USB Implementers Forum
VNC1L	Vinculum-I First Generation Embedded Dual USB Host Controller IC
VNC2	Vinculum-II Second Generation Embedded Dual USB Host Controller IC

Appendix B – List of Figures

List of Figures

Appendix C – Revision History

Document Title: AN_159 Vinculum-II Firmware Flash Programming

Document Reference No.: FT_000358

Clearance No.: FTDI# 198

Product Page: http://www.ftdichip.com/Products/ICs/VNC2.htm

Document Feedback: Send Feedback

Revision	Changes	Date
1.0	Initial Release	2011-02-04
2.0	Updated sections 5.4, 5.5 and 5.6. Also added more description to other sections. Added "Hello World" application example to demonstrate the reflasher in a custom build Edited Figure 5.1 Changed the re-flasher firmware location from 0x1c000 to 0x1f000. Updated format to new template.	2011-12-08

Revision Record Sheet

Authors	Gavin Moore
Filename	AN_159_Vinculum-II_Firmware_Flash_Programming.docx

Revision	**Date**	**Details**
1.0	2011-02-04	Initial Release
2.0	2011-10-14	Updated to new template

Sign Off

Signatory	**Signature**	**Date**
Managing Director		
Principal Hardware Engineer		
Principal Software Engineer		
Senior Marketing Manager		
Sales Manager		

Clearance Approval

(Please delete unnecessary conditions upon purposes!)

- This Document is cleared for Future Technology Devices International use and unrestricted circulation.

FTDI# 198
Clearance Number
(Where applicable for external communications)