Future Technology Devices International Ltd.

Application Note AN_148

Vinculum-II Using USB Still Image Capture Devices

Document Reference No.: FT_000294

Version 1.0

Issue Date: 2010-06-14

This application note illustrates the capabilities of the Vinculum II VNC2 by using a simple application example called "Still Image" that takes photographs on a digital still camera and copy the images taken onto a flash drive.

The techniques used in this application note are applicable to the whole range of VNC2-based modules

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
E-Mail (Support): support1@ftdichip.com Web: http://www.ftdichip.com

Table of Contents

1 Introduction 2
1.1 Overview 2
2 "Still Image" Application Layout 3
3 Hardware Setup 4
4 Software 5
4.1 Header Files 5
4.2 FTDI Files 5
4.3 Still Image Application Code 6
4.3.1 Drivers Includes Section 6
4.3.2 Setup Section –Main Function 6
4.3.3 Thread Section 7
5 Opening the Application 9
6 Building the Application 13
7 Flashing VNC2 Memory 14
8 Operation 15
9 Contact Information 16
Appendix A – List of Figures 18
Appendix B – References 19
Appendix C – Revision History 20

1 Introduction

This application note demonstrates the capabilities of the Vinculum-II VNC2 by using a simple application example called "Still Image". The application illustrates how to connect a USB still image capture device and a BOMS class device to a Vinculum-II VNC2. The application allows the user to take photographs on a digital still camera which is connected to USB1 (USB Host 1) of the VNC2 daughterboard plugged on the V2-Eval board. It then transfers the images taken to a USB flash drive which is connected to USB2 (USB Host 2) of the VNC2.

The supplied example code is used with the VNC2 V2-EVAL customer evaluation board with a 64-pin VNC2 based daughterboard.

Any software code examples given in this document are for information only and are not supported by FTDI.

1.1 Overview

This application note gives details of how to interface the Vinculum-II VNC2 device to, and how to transfer images from, a digital still image camera to a USB flash drive (BOMS class). The document firstly describes the layout of the "Still Image" application. It then describes the hardware setup and software structure of the "Still Image" application. Additionally, it illustrates the libraries (VOS Kernel Services and device drivers) and outlines the general structure of the application. This is followed by opening and building the "Still Image" application and loading the resulting ROM file onto the VNC2 device. Lastly there are instructions on how to run the application.

2 "Still Image" Application Layout

Figure 2.1 shows the layout of the "Still Image" application which consists of the file support block (FAT File System), BOMS class driver block and USB Host block (USB1 and USB2). The camera is connected to the USB1 (USB Host 1) and the USB flash drive is connected to the USB2 (USB Host2) of the 64-pin VNC2 daughterboard plugged on the V2-Eval board.

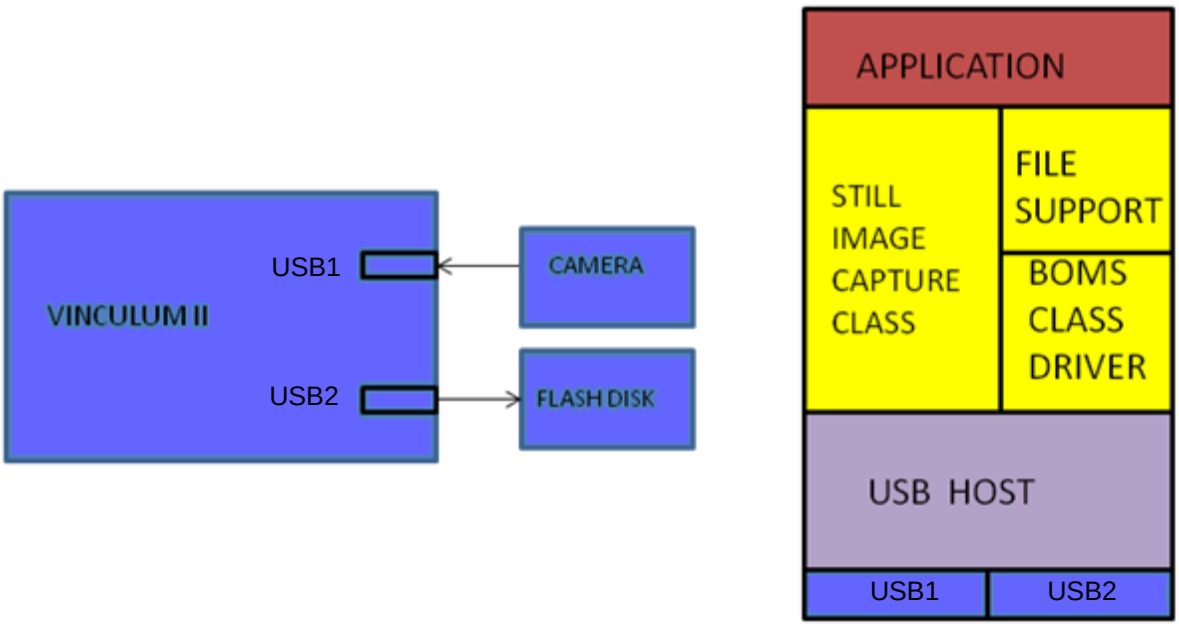

Figure 2.1 – Application Layout

The application requires the following device drivers:

The File Support block is layered on top of the BOMS driver. The File support communicates with the device file structure and provides file, directory and disk functions (format, open, read, write and close) to an application.

BOMS class driver is layered on top of the USB Host driver. The BOMS driver communicates with a mass storage class of USB device and provides access to the data on a USB drive. It also manages reading and writing data to sectors and clusters on disk.

The Still Image Capture class within the application is used to connect the camera to a USB Host.

The USB Host acts as an interface to the USB flash disk. It performs device enumeration and adds transactions and reports transaction status.

3 Hardware Setup

The "Still Image" application demonstration requires a Host PC, VNC2 customer evaluation board V2-Eval, VNC2 64-pin QFN daughterboard, a FAT-formatted USB flash drive, digital still image camera (PIMA 15740 compatible). The camera used for this example is a USB2.0 compliant Canon PowerShot SX110 IS. The demonstration also requires a USB A-Mini B cable to connect the camera to the V2-Eval USB1 and a USB 2.0 compliant A-B cable to connect the Host PC to the V2-Eval debugger port. This demonstration also works with the VNC2 48-pin QFN daughterboard.

Figure 3.1 shows the hardware set-up of the "Still Image" application demonstration:

Figure 3.1 – Hardware Setup

4 Software

This section explains how the "Still Image" application source code is structured. The application C code is available as a free download with the Vinculum-II Toolchain Installer from the FTDI website. The Vinculum-II Toolchain IDE is designed to support the development of user firmware applications for the FTDI Vinculum-II programmable USB Host / Slave controller devices. The tool suite is a royalty-free, 'C' based Integrated Development Environment (IDE), which comes complete with compiler, linker and hardware debug tools.

The code for this application must be compiled with the Vinculum-II Toolchain IDE Compiler (VinIDE compiler) before being programmed into the VNC2. The compiler translates source code from C program (high-level programming language) to Vinculum-II instructions (assembly language).

On the Host PC with the Vinculum-II Toolchain, installation of the FTDI device driver is required for the FT4232H which is connected to the VNC2 debugger port. Please refer to FTDI website for driver installation.

For further details please see AN_142 Vinculum-II Toolchain Getting Started Guide and AN_145 Vinculum-II Toolchain Installation Guide.

4.1 Header Files

The "Still Image" application consists of "StillImageApp.h" file that contains declarations of functions, constant values and all global variable declarations that are used throughout the application.

The StillImageApp.h file defines:
The size of the stack memory the application thread is going to need.

```
#define SIZEOF_FIRMWARE_TASK_MEMORY 0x1000
```

Number of device interfaces that are used within the application.

```
#define NUMBER_OF_DEVICES       5         total number of device interfaces

#define VOS_DEV_USB_HOST1       0         devices serviced on order 0, 1,2,3,4, 0, 1, 2....

#define VOS_DEV_BOMS            1

#define VOS_DEV_USB_HOST2       2

#define VOS_DEV_STILL_IMAGE     3

#define VOS_DEV_GPIO            4
```

4.2 FTDI Files

Device drivers, Kernel Services and Runtimes libraries come with the Vinculum-II Toolchain Installer in the form of archive files. Each archive file has a corresponding header file that defines its API, providing information on functions and data structures that are contained within the archive files. Both the header files and archive files are included in the "Still Image" application.

The Kernel Services provides overall control of the device drivers. The Device drivers are: USBHost driver, the BOMS driver, FAT driver and the GPIO driver is used for the V2-Eval board LEDs.

Please refer to AN_142 Vinculum-II Toolchain Getting Started Guide for further details.

4.3 Still Image Application Code

The application consists of StillImageApp.c file that contains the Drivers includes section which has declarations of the Kernel services and Driver header files. It also contains the Setup section which includes a main function. The main function acts as an entry point each time application is run. It initializes the Kernel and drivers, configures Ios, creates a thread and starts the scheduler. And lastly, a thread section which contains firmware code to control the VNC2 device.

4.3.1 Drivers Includes Section

Include statements for all header files.

```
#include "vos.h"

#include "devman.h"

#include "USBHost.h"

#include "USB.h"

#include "BOMS.h"

#include "FAT.h"

#include "GPIO.h"

#include "StillImageApp.h"

#include "StillImageApp.h"
```

Declarations.

```
VOS_HANDLE    hUsb1,

              hUsb2,

              hUart,

              hBoms,

              hCamera,

              hGpio;

vos_tcb_t    *tcbFirmware;

fat_context   fatContext;

void firmware (void);
```

4.3.2 Setup Section –Main Function

The Setup section takes care of initialization of the Kernel, CPU clock frequency, Device drivers, IO configuration, create application thread and start kernel scheduler.

The Setup Section takes care of initialization of the kernel for the number of devices being used, the time slice for each thread (Quantum) and the interval for timer interrupts (tick)

```
vos_init(10, VOS_TICK_INTERVAL, NUMBER_OF_DEVICES);
```

Initialization of the CPU clock frequency, the default clock frequency for the CPU is 48MHz.

```
vos_set_clock_frequency(VOS_48MHZ_CLOCK_FREQUENCY);
```

The device can also operate at 24MHz or 12Mhzfor lower power apps.

VNC2 IOMux configuration – this is matching signals to physical pins

```
vos_iomux_define_output(12,IOMUX_OUT_GPIO_PORT_A_1);//LED3,GPIO port A bit 1 to pin 12

vos_iomux_define_output(13,IOMUX_OUT_GPIO_PORT_A_2); //LED4,GPIO port A bit 2 to pin 13

vos_iomux_define_output(29,IOMUX_OUT_GPIO_PORT_A_5); //LED5, GPIO port A bit 5 to pin 29

vos_iomux_define_output(31,IOMUX_OUT_GPIO_PORT_A_6); //LED6, GPIO port A bit 6 to pin 31

vos_iomux_define_output(39,IOMUX_OUT_UART_TXD); //UART Tx, // UART to V2EVAL board pins

vos_iomux_define_input(40,IOMUX_IN_UART_RXD); //UART Rx

vos_iomux_define_output(41,IOMUX_OUT_UART_RTS_N); //UART RTS#

vos_iomux_define_input(42,IOMUX_IN_UART_CTS_N); //UART CTS#
```

Initialization of the USBHost driver

```
usbhost_init(VOS_DEV_USB_HOST1, VOS_DEV_USB_HOST2, &usb_ctx);
```

Initialization of the BOMs driver

```
boms_init(VOS_DEV_BOMS);
```

Initialization of the Still Image driver

```
stillimage_init(VOS_DEV_STILL_IMAGE);
```

Initialization of the GPIO driver

```
gpioCtx.port_identifier = GPIO_PORT_A;

gpio_init(VOS_DEV_GPIO,&gpioCtx);
```

Create the application thread

```
tcbFirmware = vos_create_thread(29, SIZEOF_FIRMWARE_TASK_MEMORY, firmware, 0);
```

Start the scheduler

```
vos_start_scheduler();
```

Every application includes a vos_start_scheduler and should be the last instruction after all configuration is done.

Please refer to AN_142 Vinculum-II Toolchain Getting Started Guide for further details.

4.3.3 Thread Section

This section contains firmware code to control the VNC2 device.

Opens the handle to the devices

```
hUsb1 = vos_dev_open(VOS_DEV_USB_HOST1);

hUsb2 = vos_dev_open(VOS_DEV_USB_HOST2);

hGpio = vos_dev_open(VOS_DEV_GPIO);

hBoms = vos_dev_open(VOS_DEV_BOMS);

fatContext = fat_open(hBoms, 0, NULL);

hCamera = vos_dev_open(VOS_DEV_STILL_IMAGE);
```

Finds BOMS class device

```
hc_iocb_class.dev_class = USB_CLASS_MASS_STORAGE;

hc_iocb_class.dev_subclass = USB_SUBCLASS_MASS_STORAGE_SCSI;

hc_iocb_class.dev_protocol = USB_PROTOCOL_MASS_STORAGE_BOMS;
```

Finds first hub device

```
hc_iocb.ioctl_code = VOS_IOCTL_USBHOST_DEVICE_FIND_HANDLE_BY_CLASS;

hc_iocb.handle.dif = NULL;

hc_iocb.set = &hc_iocb_class;

hc_iocb.get = &ifDev2;
```

Attach BOMS class device

```
boms_att.hc_handle = hUsb2;

boms_att.ifDev = ifDev2;

boms_iocb.ioctl_code = MSI_IOCTL_BOMS_ATTACH;

boms_iocb.set = &boms_att;

boms_iocb.get = NULL;
```

Find Still Image class device

```
hc_iocb_class.dev_class = USB_CLASS_IMAGE;

hc_iocb_class.dev_subclass = USB_SUBCLASS_IMAGE_STILLIMAGE;

hc_iocb_class.dev_protocol = USB_PROTOCOL_IMAGE_PIMA;
```

Attach Still Image class device

```
camera_att.hc_handle = hUsb1;

camera_att.ifDev = ifDev1;

camera_iocb.ioctl_code = STILLIMAGE_IOCTL_ATTACH;

camera_iocb.set = &camera_att;

camera_iocb.get = NULL;
```

Write to the flash drive

```
fat_fileWrite(&FILE, obj_info.len, NULL, hCamera, NULL);
```

Closes the handle to the devices

```
status = fat_fileClose(&FILE);

fat_close(fatContext);

vos_dev_close(hCamera);

vos_dev_close(hBoms);
```

Please refer to AN_142 Vinculum-II Toolchain Getting Started Guide for further details.

5 Opening the Application

Following the Vinculum-II Toolchain installation process, by default, the Vinculum-II Toolchain IDE is located within the *Program Files/FTDI/Vinculum-II Tool Chain* directory on the local hard disc; there will also be a start menu shortcut created, under the *FTDI/Vinculum-II Tool Chain* folder heading. The Vinculum-II Toolchain Installer can be downloaded free from FTDI website

After the Vinculum-II Toolchain is installed successfully, launch the application by double clicking on Vinculum-II Toolchain icon. The "Still Image" application will be saved within *My Documents* folder of your PC.

Select *File* tab within the toolbar and click *Open* under the *Project* group (see Figure 5.1).

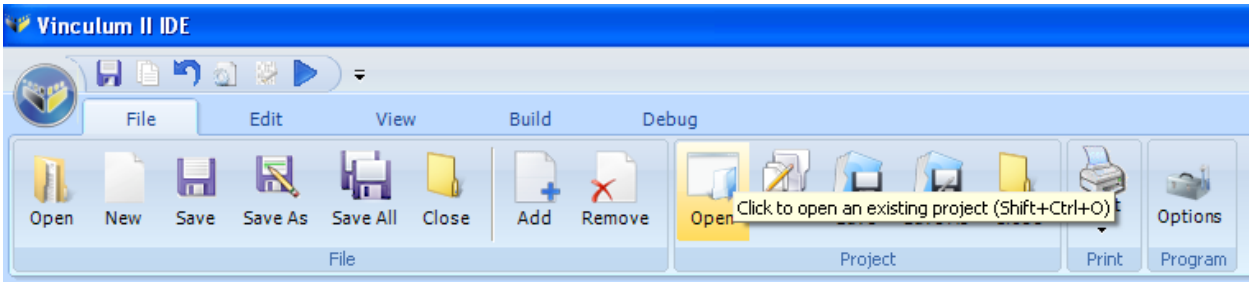

Figure 5.1 – Opening Project Using File Tab

Alternatively, go to the circular Vinculum button and click *Open->Project*

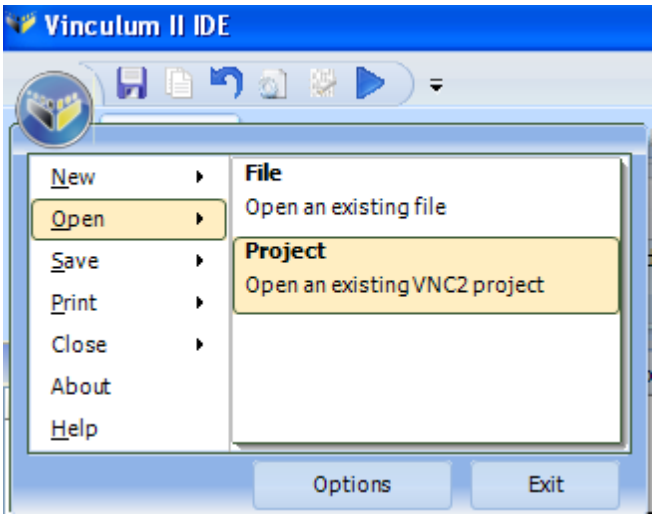

Figure 5.2 – Opening Project Using Circular Vinculum Button

This will result in the *Open Existing Project* dialog box (see Figure 5.3) appearing. Browse to *My Documents* and find the folder *FTDI/Firmware/Samples/1.0.4/General.* Within this is a folder called *StillImageApp* containing a file *StillImageApp.vproj (*vproj is the file extension used by all VNC2 project files) double click the file to open the "Still Image" Application within the Vinculum-II Toolchain IDE.

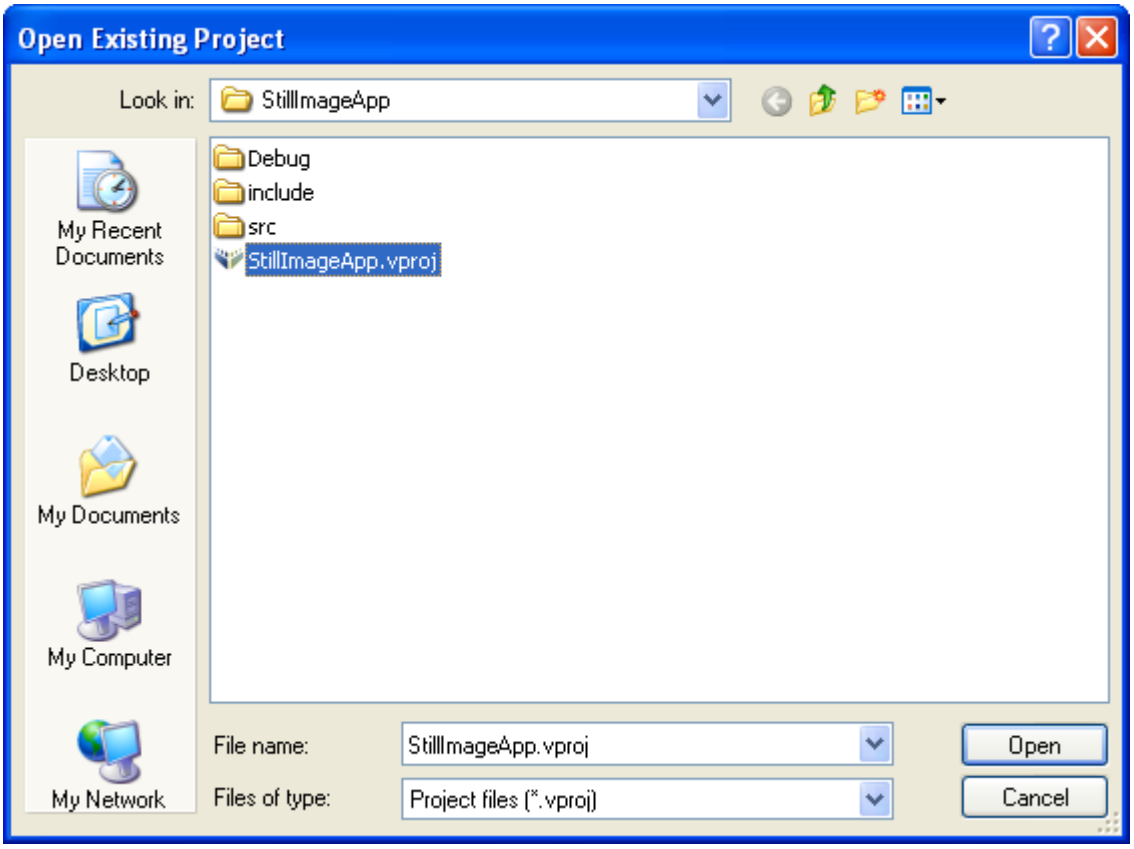

Figure 5.3 – Opening Existing Project Window

To open the "Still Image" application project click the *Open* button and the application project opens within the VNC2 Toolchain IDE (

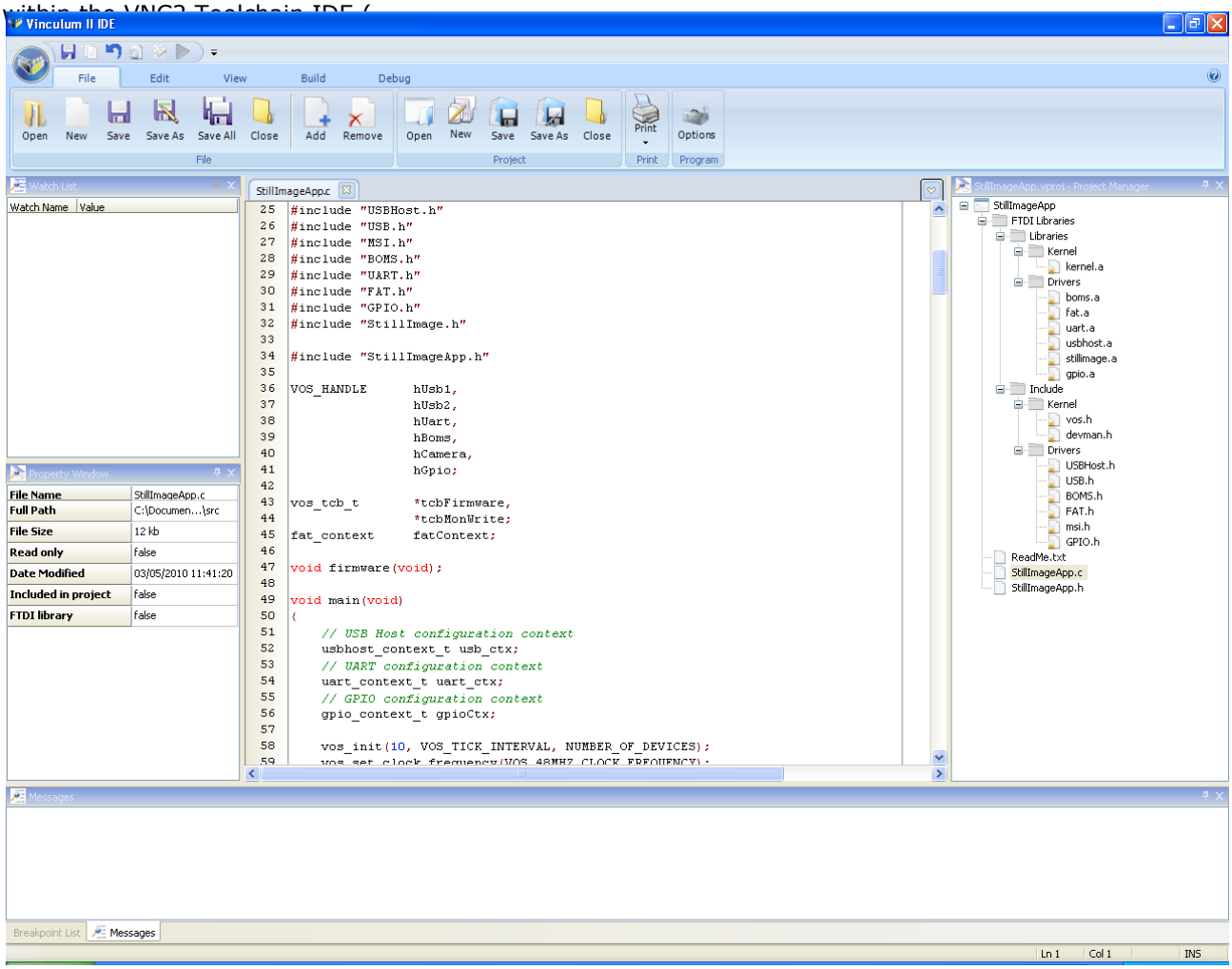

Figure **5.4**).

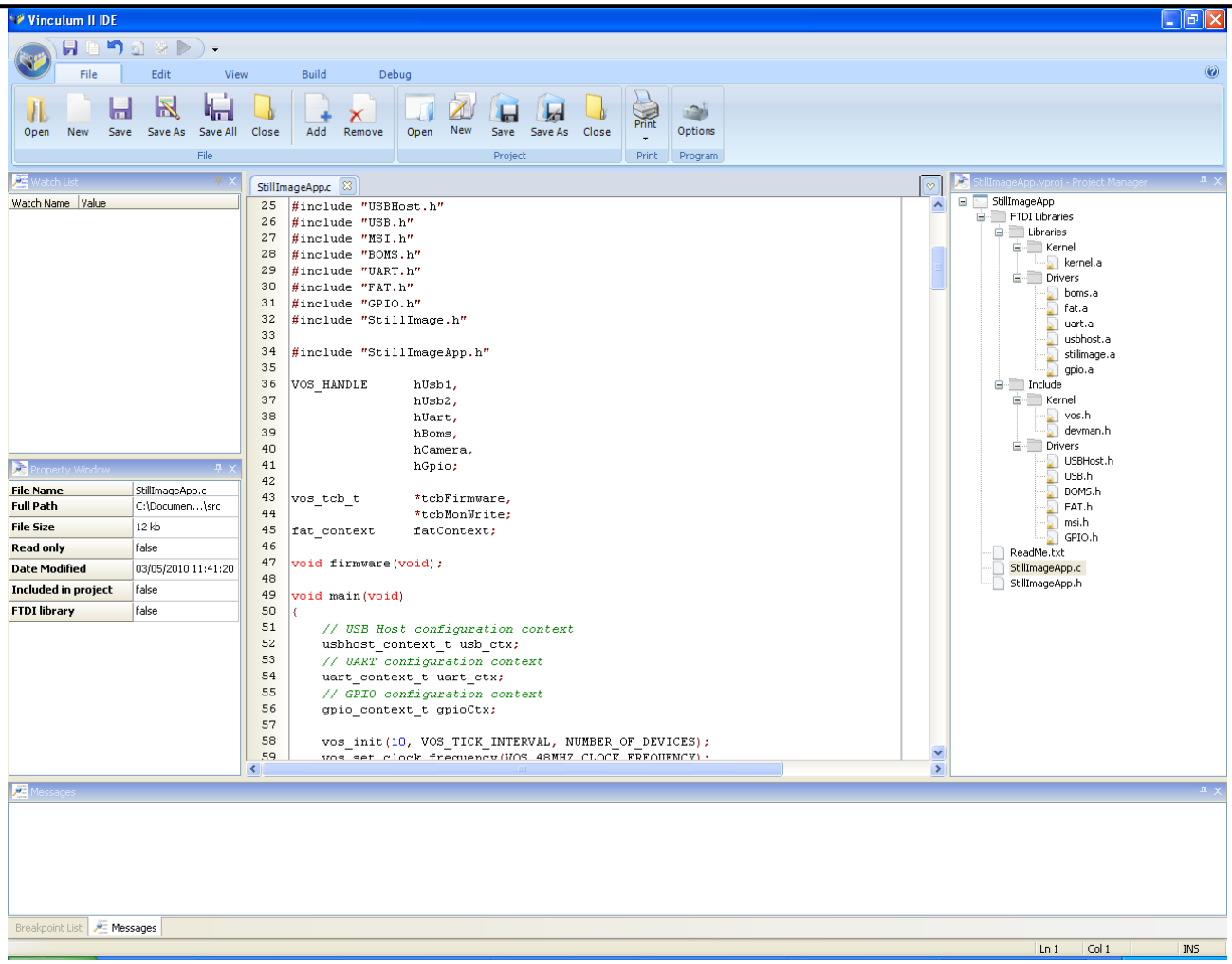

Figure 5.4 – VNC2 Toolchain IDE Window

6 Building the Application

This section contains step-by-step instructions for building the "Still Image" application. Once an application is opened, it is necessary to clean and build the application to delete files from a previous build. This build also creates an executable file for VNC2.

To build the application, select *Build* tab within the toolbar and click *Clean* and then *Build* under the *Build* group (see Figure 6.1).

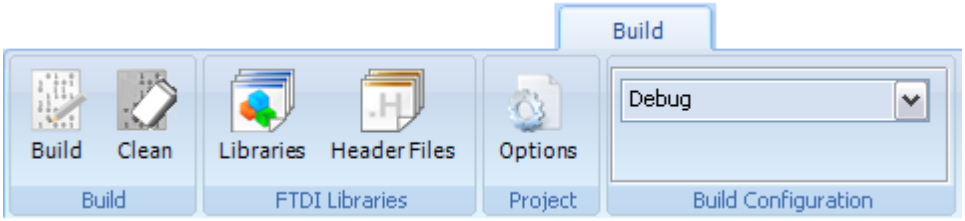

Figure 6.1 – Building Project

Clicking *Clean* will delete the object and ROM file to ensure only new files are generated. Following *Clean, click Build* which invokes the Vinculum-II Toolchain IDE Compiler compile, assemble and links the source code into a file format (ROM file) that can be loaded onto the VNC2 device. This pops up a *Compiling* dialog box (Figure 7.2) which shows a number of warnings and compilation errors (if there are any).

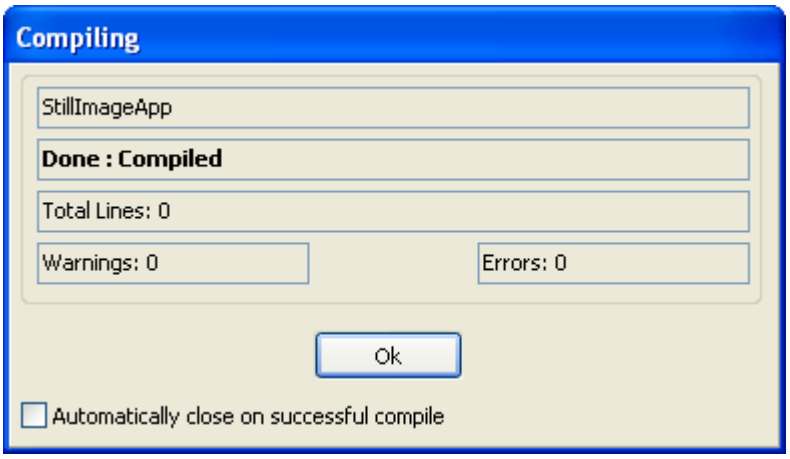

Figure 6.2 – Compiling Window

If the "Still Image" application source code is not altered, the application will successfully build with no compilation errors or warnings (see Figure 7.2).

This shows that the StillImageApp.ROM file has been successfully created. Additional information is also sent by the compiler and linker to the *Message Window* at the bottom of the IDE main window (see Figure 6.3). Ensure there are no errors.

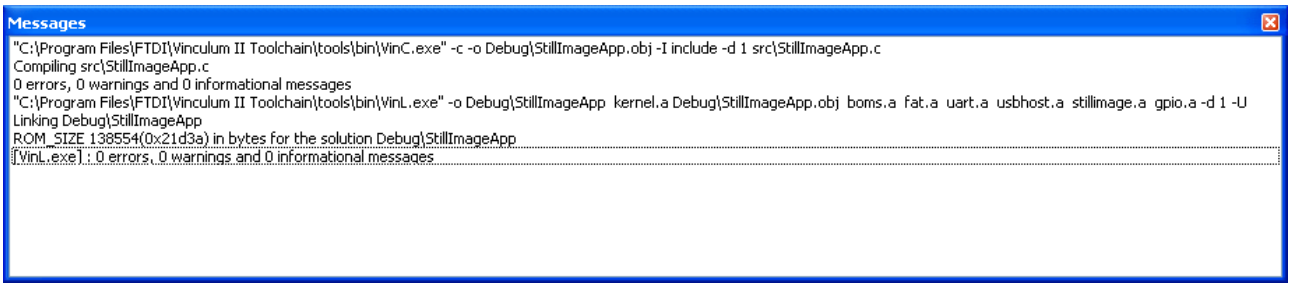

Figure 6.3 – Message Window

7 Flashing VNC2 Memory

In order to program the "Still Image" application into the VNC2 device, it is necessary to plug the 64 pin QFN daughterboard onto the VNC2 daughterboard connector – J1, J2, J3 and J4 and connect the V2-Eval board debugger port to the Host PC using the USB A-B cable.

When the V2-Eval board is connected, switch it turn ON the using SW1 (located just below the black power supply socket). The Host PC may attempt to install FTDI drivers for the FT4232H connected to the debugger port. Please visit FTDI website and download the necessary driver which matches your operating system.

Go back into the IDE and select *Debug* tab within the toolbar and click drop-down menu in the *Debugger Interface* which initiates the IDE to search for connected devices. This will automatically select the debugger interface of the FTDI V2-Eval board (see Figure 7.1). Port B of the FT4232H on the V2-Eval board is connected to the VNC2 debug port.

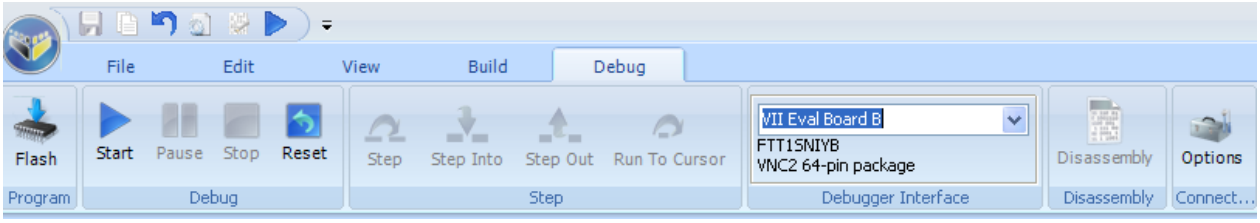

Figure 7.1 – Debugger Interface

Program the flash memory of the VNC2 by selecting the *Flash* button from the *Debug* tab*, Select ROM file to program* dialog box (see Figure 7.2) will open showing the StillImageApp.ROM image file that was built earlier, select this file and click open.

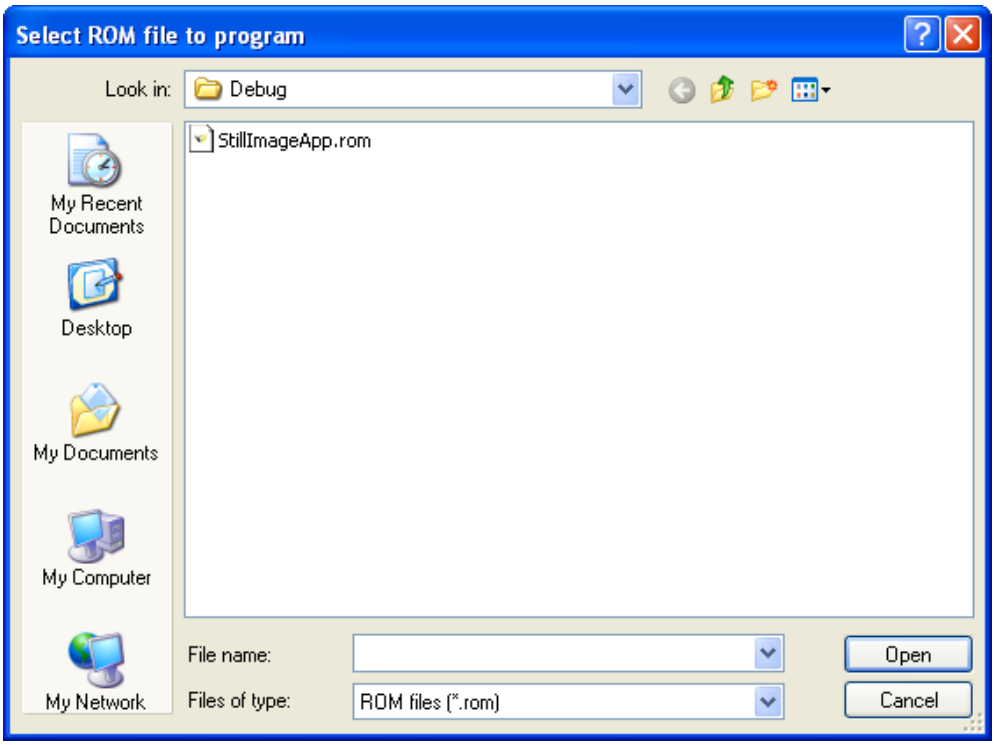

Figure 7.2 – Select ROM File to Program Window

The LED9 and LED10 located on the V2-Eval board will flash. The VinIDE will now program the VNC2 flash and a *Flashing Rom* progressing bar will appear to indicate progress and disappear after successfully completing the VNC2 programming. Additional information is shown within the *Message Window* at the bottom of the IDE main window.

8 Operation

The following sequence is required for operation of "Still Image" application demonstration:

1. Apply power to the V2-Eval board
2. Connect the USB flash drive to USB2 of the V2-Eval board
3. Connect the digital still camera to the USB1 of the V2-Eval board using the USB A-Mini B cable
4. LED3 located on the V2-Eval board should flash sequentially. This indicates that the USB device connected to USB2 has not been enumerated. The VNC2 device waits for the RESET button to be pressed using the SW2 switch on the V2-Eval board.
5. The VNC2 will detect the RESET and the USB Flash connected to the V2-Eval board will be enumerated.
6. The LED3 on the V2-Eval board should now stay ON which indicates the completion of the USB device enumeration
7. The camera should take a photograph
8. The first image is transferred to the flash drive
9. LED4 and LED5 should light up to show image transferred completed
10. The application will continue taking and transferring photographs until the V2-Eval board or camera is turned OFF or the V2-Eval board is unplugged from the Host PC or the USB flash and camera are removed from the V2-Eval board. Ensure that the V2-Eval board is turned OFF before removing the flash drive or the camera. Remove the flash disk from the board and insert it into the PC to view the photographs

9 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
Web Site URL http://www.ftdichip.com
Web Shop URL http://www.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
Web Site URL http://www.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
Web Site URL http://www.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 URL http://www.ftdichip.com

Distributor and Sales Representatives

Please visit the Sales Network page of the FTDI Web site for the contact details of our distributor(s) and sales representative(s) in your country.

Appendix A – List of Figures

Figure 2.1 – Application Layout 3
Figure 3.1 – Hardware Setup 4
Figure 5.1 – Opening Project Using File Tab 9
Figure 5.2 – Opening Project Using Circular Vinculum Button 9
Figure 5.3 – Opening Existing Project Window 10
Figure 5.4 – VNC2 Toolchain IDE Window 12
Figure 6.1 – Building Project 13
Figure 6.2 – Compiling Window 13
Figure 6.3 – Message Window 13
Figure 7.1 – Debugger Interface 14
Figure 7.2 – Select ROM File to Program Window 14

Appendix B – References

AN_145: Vinculum-II Toolchain Installation Guide

AN_142: Vinculum-II Tool Chain Getting Started Guide

AN_137: Vinculum-II IO Cell Description

AN_138: Vinculum-II Debug Interface Description

AN_139: Vinculum-II IO Mux Explained

AN_140: Vinculum-II PWM Example

AN_144: Vinculum-II IO Mux Configuration Utility User Guide

AN_147: Vinculum-II Tool Using BOMS Class devices

Appendix C – Revision History

Revision	Changes	Date
1.0	Initial release	2010-06-14