VxWorks integration

Revision 1.7, 9rd Mar 2010
www.touch-base.com\documentation\installation

Introduction Integration VxWorks 6.4.x VxWorks 5.5.1 VxWorks 5.4.2 Contact

VxWorks from Wind River is a real-time operating system.

Within this environment, VxWorks is the embedded OS image created within the VxWorks Integrated Development Environment (IDE). The image can be build for a number of platforms. There is also WindML, otherwise known as UGL, which is the main graphics library on VxWorks and all other technologies seem to be built upon it. These include Zinc (a C++ wrapper for UGL) and the embedded Java runtime.

A good reference site for VxWorks and the IDE is http://www.xs4all.nl/~borkhuis/vxworks/vxworks.html or http://www.windriver.com/products/development_suite/

The prerequisites for running the driver are that WindML and C++ support are built into the VxWorks image.

It is our experience that every VxWorks works system is unique and therefore does not lend itself to 'packaged' software and therefore in every request for a VxWorks touch driver we need to build a new driver with the required touch controller, embed it on the target VxWorks system and test/fix until all is working.

However, due to economic considerations (cost of development tools verses return on investment) we do not have a permanent VxWorks development license and this has been supplied to us each time we have needed to undertake development. A prerequisite of any new development is the temporarily supply of the appropriate VxWorks development licenses (for supported IDEs) or the VxWorks IDE software. We also require the target VxWorks system to test the driver.

Integration procedure

Drivers can be supplied for various releases of VxWorks. In the most recent releases the code is based on UPDD version 4.0.x. Later releases are based on UPDD 3.x.x. The first release only supported serial devices but we now also offer support for specific USB devices.

General notes

1) Global Variables

VXWorks implements a shared global namespace so functions and global variables must be unique.
If unexpected crashes are observed it is possible that a name has been used in UPDD and another program in the image.

One such problem occurred with calibration where we were reading data from a msg queue to a global variable called "dummy" and the customer's image contained a 'dummy' variable. This resulted in a "page fault in msgQReceive" when spawning the tbcalibvx task until the customer changed the dummy variable name.

2) File Descriptors

VxWorks keeps track of "file descriptors" used (files, sockets, pipes...) by means of a fixed size list that has a default size of 50 entries. Most systems will use a number of these 'file descriptors' and the number changes dynamically when operations are performed. It is important to have always some of these descriptors free because otherwise system errors will occur.

The UPDD driver requires approximately 25 file descriptors (mainly pipes) in order to work. One customer did not configure sufficient "file descriptors" and reported problems with USB devices dynamic attachment.

In this particular system the problem was solved by increasing the number of "file descriptors" (NUM_FILES = 100).

VxWorks 6.4.x

WindML 5.0
IDE 2.6 – Workbench
UPDD 4.0.6

Feb 2009 release
Under this environment we have only tested ELO USB controllers:

USB layer:
-Copy usrUsbUPDDInit.c to C:\WindRiver\vxworks-6.4\target\config\comps\src\
-Copy 10usb.cdf to C:\WindRiver\vxworks-6.4\target\config\comps\vxWorks\ overwriting the copy already there
-Copy usbUPDDLib.h to C:\WindRiver\vxworks-6.4\target\h\drv\usb
-Copy Makefile to C:\WindRiver\vxworks-6.4\target\src\drv\usb overwriting the copy already there
-Copy usbUPDDLib.c to C:\WindRiver\vxworks-6.4\target\src\drv\usb
-Copy device.c to C:\WindRiver\vxworks-6.4\target\src\usb2\usbd overwriting the copy already there
-In Workbench in the "gui_image" project, open the Kernel Configuration screen
-Navigate to hardware,peripherals,usb devices and right click "UPDD" and select "include"
-Navigate to hardware,peripherals,usb devices,usb device init and right click "UPDD init" and select "include"

(*Note: If the include option is disabled (greyed out) you will have to rebuild the target/src directory first in order to make these components available*)
-Rebuild your BSP by launching a VxWorks Development shell from the start menu and then typing the following:

"cd C:\WindRiver\vxworks-6.4\target\src"
"make CPU=PENTIUM4 TOOL=diab"

UPDD Daemon:
-Copy tbupdd.reg to /ata1a/
(*Note – this is the UPDD driver settings file. Any changes to UPDD settings will need to be made to this file. Under VxWoks this file has Unix new lines at the end of each line rather then the conventional DOS newlines.* ***It is important that the Unix newlines are retained when editing and writing back the file as the driver will not be able to read its settings if they are lost and will not work. If in doubt use a conversion utility, such as "dos2unix" to convert the file (http://www.bastet.com****)).*
-Link libtbupddvx.a into your image
-Call it from usrAppInit by using the following line:

PipeDevCreate ("/ata1a/usbDataPipe", 10, 100); (required if application should function without touch screen connected)
taskSpawn("tbupddvx",100,0,48000,(FUNCPTR)tbupddvx,0,0,0,0,0,0,0,0,0,0);

You will need to provide a declaration for this function.

void tbupddvx();

TBCalib:
-Link libtbcalibvx.a into your image
-The calibration program can be invoked on demand by executing the following function.
Note: VXWorks must be in graphics mode before this call and you will need to refresh the screen after the process completes.

taskSpawn("tbcalibvx",100,0,48000,(FUNCPTR)tbcalibvx,0,0,0,0,0,0,0,0,0,0)

You will need to provide a declaration for this function.

void tbcalibvx();

WindML:
-Open C:\WindRiver\components\windml-5.0\config\wrmdb\windML_PTR_DEVICE_VIRTUAL.wrmdb in a text editor and uncomment the lines:

```
#virtualptr.ARCH=&ARMARCH5:\
#               &ARMARCH6:\
#               &MIPS32:\
#               &MIPS64:\
#               &I80486:\
#               &PENTIUM:\
#               &PENTIUM2:\
#               &PENTIUM3:\
#               &PENTIUM4:\
#               &PPC32:\
#               &SH32:\
#               &SH7750:\
#               &XSCALE
```

-In the workbench, navigate to the WindML project and open the "config.windml" file.
-Go to Display, Pointer
-Select "virtual pointer" from the device list
-Save
-Rebuild WindML

Limitations

1) ELO USB controller implementation - It is not possible to use other HID devices with a subclass=none in this configuration.

VxWorks 5.5.1

WindML 3.0
IDE 2.2.x – Tornado
UPDD 4.0.6

Mar 2010 release
Added support for another USB controller – the ELO 2216 USB controller and tested DMC serial.

Jun 2009 release
Added support for another USB controller – the Panjit (Viper) USB controller.

July 2008 release
Initial support for two USB controllers, namely EELY and DMC.

USB

Installing the USB interface module

1. Close running Tornado sessions
2. Copy usrUsbUPDDInit.c to <Tornado>\target\config\comps\src
3. Copy usbUPDDLib.c to <Tornado>\target\src\drv\usb
4. Copy usbUPDDLib.h to <Tornado>\target\h\drv\usb
5. Edit <Tornado>\target\src\drv\usb\Makefile and add "usbUPDDLib.c" to the "DOC_FILES" section
6. Edit <Tornado>\target\src\drv\usb\Makefile and add "usbUPDDLib.o" to the "OBJS" section
7. Copy 10usb.cdf <Tornado>\target\config\comps\vxWorks, overwriting the existing version
8. Rebuild the BSP

Important DMC USB controller notes:

The DMC controller is a "VENDOR_SPECIFIC" USB device and as such requires version 2.3 of the USB stack from Wind River as earlier versions are not compatible with "VENDOR_SPECIFIC" devices.

This controller also requires 3 USB requests to send initialisation data:

To use USBrequests with USB stack 2.3 then please replace your $TORNADO\target\src\usb2\usbd\device.c with the one included with the driver package and then modify the "usbdSubmitRequest" calls at line 2499 as below.

```
if(pNewDeviceInfo->uVendorID == 0xAFA && pNewDeviceInfo->uDeviceID == 0x3E8)
{
     UCHAR buf[4];
     UINT32 len;
     len = 2;
      usbdSubmitRequest(pNewDeviceInfo, 0xC0, 0x55, 0, 0, &len, buf);
      usbdSubmitRequest(pNewDeviceInfo, 0xC0, 0x5, 0x45, 0, &len, buf);
      usbdSubmitRequest(pNewDeviceInfo, 0x40, 0x31, 0, 0, &len, buf);
}
```

Project integration

1. Open your project in Tornado
2. In the "Workspace" windows, select the VxWorks tab and open "hardware\peripherals\USB Devices" in the tree
3. Right click "UPDD" in the tree and select "Include UPDD" from the menu
4. In the "Workspace" windows, select the VxWorks tab and open "hardware\peripherals\USB Devices\USB Devices Init" in the tree
5. Right click "UPDD Init" in the tree and select "Include UPDD Init" from the menu
6. Add tbcalibvx.pl and tbupddvx.pl to your project
7. Rebuild your vxWorks image file

Enabling UPDD in WindML

1. Create the directory <Tornado>\target\src\ugl\driver\pointer\updd
2. Copy the files "depend.PENTIUM3gnu", "uglupdd.c", "depend.PENTIUMgnu", "Makefile" to this directory
3. Copy the file "uglupdd.h" to the directory <Tornado>\target\h\ugl\driver\pointer
4. Edit the file <Tornado>\host\resource\windML\config\database\windML_INPUT_DB.cfg and add "updd" to the "POINTER=" line
5. Add the following section to the bottom of <Tornado>\host\resource\windML\config\database\windML_INPUT_DB.cfg

```
# UPDD

updd.NAME=UPDD Touchscreen
updd.ARCH=pentium3
updd.SELECT=INCLUDE_UPDD
updd.DEVNAME=/ata0a/tbupddvx/comReadPipe
updd.HEADER=ugl/driver/pointer/uglupdd.h
updd.DIR=updd
```

6. Run the WindML configuration program from Tornado
7. Click the "Devices" tab
8. Click the drop down box for the "pointer" and select "UPDD Touchscreen"
9. Save the configuration and exit
10. Build WindML

Invoking the driver and calibration procedure

1. Ensure that the following directories exist on the system: "/ata0a/tbupddvx", "/ata0a/tmp"
2. Copy the file "tbupdd.reg" to the "/ata0a/tbupddvx" directory

The driver daemon must be running for UPDD to function correctly. To launch it, call the following function:-
taskSpawn "tbupddvx",100,0,48000,tbupddvx
You must calibrate the touchscreen before you use it. To do this call the following function:-
taskSpawn "tbcalibvx",100,0,48000,tbcalibvx

Serial

This version has been tested with DMC TSC serial controllers. Other serial devices should work so we suggest testing this release and addressing any possible issues that arise.

Project integration

1. Open your project in Tornado
2. Add tbcalibvx.pl and tbupddvx.pl to your project
3. Rebuild your VxWorks image file

Enabling UPDD in WindML

1. Create the directory <Tornado>\target\src\ugl\driver\pointer\updd
2. Copy the files "depend.PENTIUM3gnu", "uglupdd.c", "depend.PENTIUMgnu", "Makefile" to this directory
3. Copy the file "uglupdd.h" to the directory <Tornado>\target\h\ugl\driver\pointer
4. Edit the file <Tornado>\host\resource\windML\config\database\windML_INPUT_DB.cfg and add "updd" to the "POINTER=" line
5. Add the following section to the bottom of <Tornado>\host\resource\windML\config\database\windML_INPUT_DB.cfg

```
# UPDD

updd.NAME=UPDD Touchscreen
updd.ARCH=pentium3
updd.SELECT=INCLUDE_UPDD
updd.DEVNAME=/ata0a/tbupddvx/comReadPipe
updd.HEADER=ugl/driver/pointer/uglupdd.h
updd.DIR=updd
```

6. Run the WindML configuration program from Tornado
7. Click the "Devices" tab
8. Click the drop down box for the "pointer" and select "UPDD Touchscreen"
9. Save the configuration and exit
10. Build WindML

Invoking the driver and calibration procedure

1. Ensure that the following directories exist on the system: "/ata0a/tbupddvx", "/ata0a/tmp"
2. Copy the file "tbupdd.reg" to the "/ata0a/tbupddvx" directory

The driver daemon must be running for UPDD to function correctly. To launch it, call the following function:-
taskSpawn "tbupddvx",100,0,48000,tbupddvx
You must calibrate the touchscreen before you use it. To do this call the following function:-
taskSpawn "tbcalibvx",100,0,48000,tbcalibvx

VxWorks 5.4.2

Tornado 2.0.2
WindML 2.0
UPDD 3.x.x

The version only supports serial controllers. For future serial requirements we would suggest testing UPDD VxWorks version 4 and addressing any issues that arise.

VxWords integration instructions

-Copy uglupdd.c to <tornado>\target\src\ugl\driver\pointer
-Copy uglupdd.h to <tornado>\target\h\ugl\driver\pointer
-Open <tornado>\target\h\ugl\config\uglDepend.h
-At line 201 insert the following:-

```
/* UPDD Touchscreen */
#ifdef INCLUDE_UPDD
#include <ugl/driver/pointer/uglupdd.h>
#endif /* INCLUDE_UPDD */
```

-Open <tornado>\host\resource\ugl\uglDB.cfg
-At line 63 insert the following:-

```
# UPDD

DEVICE   UPDD Touchscreen
ARCH PENTIUM
SELECT INCLUDE_UPDD
DEVNAME "/tbupddvx/comReadPipe"
HEADER   uglupdd.h
BSPHDR
```

-Open <tornado>\target\src\ugl\config\uglInit.h
-At line 249 insert the following:-

```
#undef  INCLUDE_UPDD            /* Updd touchscreen */
```

-Open <tornado>\target\src\ugl\driver\pointer\uglupdd.c
-At lines 18 and 19 modify the definitions of SCREEN_WIDTH and SCREEN_HEIGHT to match the video resolution you are using.
-In Tornado, select Tools, WindML

-Configure WindML for your hardware, selecting "UPDD Touchscreen" as your pointer device.
-Clean, Build and download WindML to your target machine
-Download "tbupddvx.out" to your target machine
-Download "tbcalibvx.out" to your target machine

VxWorks Runtime Instructions

-Create a directory "/tbupddvx"
-Copy "tbupdd.reg" to "/tbupddvx"
-Run "tbupddvx" with a stack size of at least 48000 bytes. e.g. to load it from the Tornado shell type:- "taskSpawn "tbupddvx",100,0,48000,tbupddvx" See below*****
-Run "tbcalibvx" to calibrate the driver for the first time. e.g. to load it from the Tornado shell type:- "sp tbcalibvx"

VxWorks Instructions to load and run "tbupddvx.out"

-Open Tornado
-If you do not have a target server running then do the following:-
-Click Tools, Target Server, Configure
-Select the server you want to use (we assume you will only see one)
-Click Launch
-In the dropdown box on the LAUNCH toolbar, select the target server you are using (again we assume you will only see one)
-Click Project, Download
-Select "tbupddvx.out"
-Click Tools, Shell
-Select your target machine in the drop down box, and click
OK to launch the Tornado shell
-In the Tornado shell type: taskSpawn "tbupddvx",100,0,48000,tbupddvx

At this stage the driver is running, and you should load your GUI program.

Contact

For further information or technical assistance please email the technical support team at technical@touch-base.com