OS/2 USB stack update

Gregg Young

The Team: Active Development

- Lars Erdmann
  - USB stack (usb?hcd.sys drivers, usbd.sys)
- Ruediger Ihle
  - USB human interface (usbhid.sys, usbkbd.sys, usbmouse.sys)

The Team: Source to 10.162 Level

- Gregg Young
  - usb?hcd.sys
- David Azarewicz
  - usbd.sys

The Team: Technical Support/Testing

- Wim Brul
- Steven Levine
- Richard L Walsh

The Team: Management

- Joachim Benjamins
- Roderick Klein

The Challenge

- Lack of source code for most recent drivers
- Problems related to the USB standard.
  - Windows didn't follow it exactly.
  - Device makers don't adhere to it.
  - It can be ambiguous.
- Backward compatibility and IBM design decisions

Lack of source code for most recent drivers

- Source code in the DDK is level 10.100
- The compiler in the DDK isn't the same version as the one used to build the 10.162 drivers

Process of updating source to the latest level

- Disassemble 10.162
  - IDAFW for DOS
- Build the 10.100 source
- Disassemble 10.100
- Cleanup the disassembled code
- Diff the two

Assembler Diff

```
--- USBEHCDA.asmx       2011-01-17 16:03:12.000000000 -0500
+++ USBEHCD6.asmx       2011-01-17 16:03:12.000000000 -0500
@@ -400,19 +400,17 @@
                 mov     [bp-8], ax
 
 _DecLongToASCII_b4:
-                mov     ax, [bp+8]
-                mov     dx, [bp+0Ah]
-                cmp     [bp-2], dx
-                ja      _DecLongToASCII_f5
-                jb      _DecLongToASCII_f6
-                cmp     [bp-4], ax
-                ja      _DecLongToASCII_f5
+                mov     ax, [bp-4]
+                mov     dx, [bp-2]
+                cmp     [bp+0Ah], dx
+                jb      _DecLongToASCII_f5
+                ja      _DecLongToASCII_f6
+                cmp     [bp+8], ax
+                jb      _DecLongToASCII_f5
 
 _DecLongToASCII_f6:
                 add     word ptr [bp-8], 1
                 adc     word ptr [bp-6], 0
```

```
DecWordToASCII(PSZ StrPtr, USHORT wDecVal, USHORT Option)·
{·
   BOOL  fNonZero=FALSE;·
   USHORT  Digit;·
   USHORT  Power=10000;·
·
   while (Power)·
   {·
      Digit=0;·
      while (wDecVal >=Power)                    //Digit=wDecVal/Power;·
      {·
         Digit++;·
         wDecVal-=Power;·
      }·
·
      if (Digit)·
         fNonZero=TRUE;·
·
      if (Digit ||·
          fNonZero ||·
          (Option & LEADING_ZEROES) ||·
          ((Power==1) && (fNonZero==FALSE)))·
      {·
         writeChar((char)('0'+Digit));·
      }·
·
      if (Power==10000)·
         Power=1000;·
      else if (Power==1000)·
         Power=100;·
      else if (Power==100)·
         Power=10;·
      else if (Power==10)·
         Power=1;·
      else·
         Power=0;·
   } // end while·
·
   return (StrPtr);·
}·
Loaded E:\ddk\base\src\dev\usb\misc\usbdebug.c.
```

What is going on?

- The output is binary equivalent
- Different compiler switches?
  - We tried all the combos that made sense
- Different pragmas?
  - #pragma optimize("ceglt", on)
  - #pragma optimize("eglt", off)

What is going on?

- Different compiler version.
  - We tried several
- Other oddities
  - Changing the length of a variable name could change the number it was assigned by the compiler
  - Disassembler output numbered in hex; List file output in decimal
    - Fixed (Thanks Steven).

Other Oddities

- Just because the new C code didn't make any sense didn't mean it was “wrong”.
- A smaller diff file wasn't always an improvement.
- New function had been added.
  - How to name them?
- You know you are close because it traps in the same place.

```
+_OHCIInterrupt endp
+
+
+
+
+
+
+sub_b1 proc near
+               push    bp
+               mov     bp, sp
+               mov     bx, [bp+8]
+               jmp     short sub_b1_f1
+
+
+sub_b1_b2:
+               mov     ax, [bp+4]
+               mov     dx, [bp+6]
+               cmp     [bx+10h], ax
+               jnz     sub_b1_f3
+               cmp     [bx+12h], dx
+               jz      sub_b1_f4
+
+sub_b1_f3:
+               mov     bx, [bx+1Eh]
+
+sub_b1_f1:
+               or      bx, bx
+               jnz     sub_b1_b2
+
+sub_b1_f4:
+               mov     ax, bx
+               leave
+               retn
+sub_E53_63DE   endp
+
+
+
+
+
+
0892:01 (3037:20) IA S  DIFF  E:\IDAFW\ASM1.100\USBOHCDA.diff~                  43,2B 1
```

Progress made to date

- The sync of usbehcd.sys, usbuhcd.sys and usbohcd.sys. Is as complete as it is going to get.
  - I have gone back and looked at problem areas at Lars' request but so far the problems sync with the IBM driver.
- usbd.sys is still a work in progress.

Progress made to date from Lars

- Reworked the root hub processing for all 3 HC flavors.
  - devices not properly attaching
  - hangs at bootup where calls to the root hub processing are also done (to reset ports for example)
- Improved the function of the /FS switch ("force shutdown") for all 3 HC flavors.
  - Held in reset on shutdown to eliminate hangs in bios on restart.

Progress made to date from Lars

- APM suspend/resume for all 3 HC flavors.
  - tested with memory sticks and sometimes they don't "survive" a resume but it is likely that this is related to USBMSD.ADD and not the HC drivers.
- Incorporated Martin Kiewitz fixes for USB enumeration.
  - Not working as of yet.
- Support isochronous transfers on USB 2.0
  - On going

Lars' Observation on 10-25-11

USB is one big "timing issue" which is difficult to track down. Currently it's "trial and error" and the Linux and Apple code tells me that these guys went through the very same problems (with the support of chipset vendors that would tell them how it "really works" for their HW as the specs leave plenty of room for interpretation). Let's hope for the best.

Lars from 9-25-11

I have found a precise description on how Windows 7 USB drivers (in our case it's an interaction of USBxHCD.SYS and USBD.SYS) do USB enumeration:Windows 7 Developer Blog
I definitely recommend we follow the Windows way of doing it even though it differs from what the USB specs say

USB 10.175

- This is Lars' latest release on Hobbes.
- We tried installing it on some systems that would hang after the last device driver was loaded for 70 seconds! This was caused by the USB host controller drivers or UBSD.SYS. The Quad-core Intel box now boots 66 seconds quicker!!!
- My Duel-core boots noticeably faster with these also.
- More testing is needed.

USBHID.SYS

- Rudi has updated usbhid.sys, usbmouse.sys and usbkbd.sys.
  - These are based on the 10.100 code base.
  - They fix the problem with failure of a usb mouse after a “special” key on the keyboard is pressed.
  - Should better allow multiple pointing devices
  - Fixes the Alt, Ctrl key sticking
  - Adds a spy feature

Where do I get them?

- usbhid.zip is attached to Ticket 2266
- USB 10.175 is here:
  USBHCD13.zip on Hobbes
- Testing is needed.

How can I help?

- Post bugs to: http://svn.netlabs.org/usb/report
- ALWAYS state WHAT VERSION of the driver you were using when the trap occurred !
- Especially interested in reports on older systems (also without ACPI.PSD) and on systems using ACPI.PSD in APIC mode (/APIC) and/or multi-core (/SMP). These drivers SHOULD work on the later systems

How can I help?

- Please provide as much info as you can:
  - Did it work with the IBM supplied 10.162 drivers or previous versions of these drivers? Please try combinations of IBM drivers and these drivers.
  - Kernel version: run "bldlevel OS2KRNL" from the root directory.
  - How many CPUs / CPU cores ?
  - ACPI.PSD? Version? What switches?
  - What type of USB? UHCI? OHCI? EHCI?

How can I help?

- Output of "pci.exe" is very helpful:
  pci104vka.zip on Hobbes
- If you have eCS 2.0, you can instead use:
  \ecs\install\DETECTEI\pciscan.exe
- USB relevant parts in Config.sys, in particular, order and number of driver instances:
  USBD.SYS and USB(I|O|E)HCD.SYS

How can I help?

- Traps or hangs: describe exactly when it occurs. Hit Alt-F2 when the boot blob appears in the upper left corner and note down last driver that loaded successfully.
  - What driver / daemon program loads after the last successfully loaded driver ?
  - If trap in USB(I|O|E)HCD.SYS: record the trap screen, CS:IP, SS:BP and AX, BX, CX, DX, ES, DS are especially important.
    - Take a digital picture of the trap screen and attach it to the report

•How to set up a trap dump and tracing

- For setting up a dump partition you can have a look at: comp.os.os2.bugs
- For additional information: Steven Levine's OS2 Diagnostics

Setting up a dump partition

- Create a partition > the physical size of your memory and give it a drive letter.
  - Note: if you use LVM, the partition needs to be "compatible" and "bootable". After creating it, you should however remove it from boot manager.
- Get:dumpfs.zip
  - Copy udumpfs.dll to \os2\dll, copy dumpfs.ifs to \os2\boot, backup os2dump and copy os2dump.hd to os2dump.

Setting up a dump partition

- Add "IFS=?:\OS2\BOOT\DUMPFS.IFS" to config.sys. Make it the last IFS
- Add "TRAPDUMP=R0,X:" to config.sys.
- Add tracing now to avoid multiple reboots
  - Add "TRACEBUF=512 /M=WRAP,QUEUED,NODTI /D=ALL" to config.sys
  - Add "TRACE=ON 226" to config.sys
- Reboot

Setting up a dump partition

- Run "format X: /FS:DUMPFS" to format the dump partition.
- Now you are basically set up to do a dump.

Capturing a Dump

- Now, when the system reboots and hangs, hit Ctrl-Alt-NumLock-NumLock to initiate a system dump. Let the system dump finish
- If it traps the dump will be initiated immediately.
- After the dump completes it will reboot the machine. Remember to boot from a different partition.

Analyzing a Hang

- Now you will need pmdf.exe to extract the trace info from the system dump: do a "\os2\pdpsi\pmdf.exe" to open the dump formatter
  - Select file->open go to your dump partition and select the dump file.
  - You can now enter commands in the command line at the very bottom of the pmdf window. Enter ".ts c:\trace.raw" to save the trace buffer to file
  - "c:\trace.raw". We need this file for the trace information.

Analyzing a Trap

- Load the file in PMDF.exe
- Select Synopsis -> Trap Screen from the Analyze menu.
- Select Thread -> Call Gate from the Analyze menu.
- If the thread is in a Call Gate, select Thread -> Ring 0 Stack Trace from the Analyze menu.
- Select Thread -> Ring 3 Stack Trace from the Analyze menu.

Analyzing a Trap

- Enter the commands in the PMDF command line at the bottom of the window. Press the Enter key after each command.
  - r
  - ln
  - u eip-20 eip
  - k
  - dd ebp
  - dd esp
  - db esp

Analyzing a Trap

- If the r command does not report the same cs:eip as shown on the Trap Screen, repeat the above commands substituting:
  - the numeric cs:eip value from the Trap Screen for eip
  - the numeric ss:ebp value from the Trap Screen for ebp
  - the numeric ss:esp value from the Trap Screen for esp.

Analyzing a Trap

- Select Process -> Open files the from Analyze menus
- Select Process -> Module Table from the Analyze menu.
- Select Save Output from the File menu and save the window contents to a file.
- Save the dump file to a different location since it might be needed for further analysis

Questions?

ygk@qwest.net

Thanks