MCF51JM128 eval problems - CMX USB software

By Dave Nadler

Revision 1.0 (20 August 2008)

Abstract

We've encountered serious problems while evaluating the MCF51JM128 "Flexis" 32-bit part; primarily with the USB software provided by Freescale (authored by CMX) supporting a USB thumb-drive (memory-stick). This document enumerates the problems.

The most serious problems include failing to initialize reliably and infinite looping with some thumb-drives.

Without prompt resolution of these problems, we will need to switch to an alternate vendor (probably Microchip's new PIC32 offering). Other potential Freescale customers will face the same issues.

Document revisions

Revision, Date, Author	Description
Revision 1.0 (20 August 2008), Dave Nadler	Consolidation of notes from earlier problem submissions, as requested by Pamela at Freescale support.

Table of Contents:

Background

We are adding USB thumb-drive support to an existing product. The existing product has a RS-485 LAN; we'll add a new LAN node that interfaces to the thumb drive. The processor for this new LAN node thus requires:

- host USB support
- USB software stack supporting mass storage devices
- USB thumb-drive (FAT file system) software stack
- serial port (9-bit capable for our LAN)

I am using a DEMOJM board to evaluate the MCF51JM128 for this application. Our testing is based on Freescale's "Lab 5 Host-Mass-Storage Class" demo with additional diagnostics added.

The USB software provided by Freescale (written by CMX) has serious problems. If these problems cannot be resolved promptly, the Freescale solution is unfit for our application or any application needing to support a USB thumb-drive.

Problem 1: Erratic Initialization

When a USB thumb-drive is plugged into the evaluation board, the device is not reliably initialized. Errors reported (I had to add some additional diagnostics) include:

```
Error connecting to inserted device (Tr_error = 4)
Error connecting to inserted device (Tr_error = 6)
No valid mass storage device identified
Mass-storage driver started.
```
-- This one is followed by hang (infinite loop)

Repeatedly pressing reset eventually gets the board and device into a state that works. Sometimes it works properly for hours, then does this. This behavior occurs with many different USB thumb-drives.

Things that are **not** the problem:

- The power supply was questioned (power from PC USB, the memory stick draws 150ma), so I am using an external supply for the DemoJM board. There is definitely adequate and stable power (it was suggested that USB power from a Dell Latitude PC might not be adequate for this application).
- USB pull-down drivers: in case software was not properly pulling down USB_DN and DP signals, jumpers changed on DemoJM board to always connect the pull-downs to ground (per USB specification).
- stack space: CMX's stack use subroutine was used to verify that the default stack of 0x200 was not exceeded (and then stack space was increased to 0x400 for good measure).

Problem 2: Infinite Looping on Write

I started with Freescale's "Lab 5 Host-Mass-Storage Class" demo, and added a test command to write a data file. I have a selection of several thumb-drives. For most thumb-drives, the write operation operates properly.

Trying to write data to a VisionTek 64MB USB2.0 thumb-drive, CMX code goes into an infinite loop trying to write the updated directory entry for the new file. In routine scsi_cmd_write_10 (SCSI.C), each transfer results in an ERR_IO_CHECK, and the routine loops infinitely trying to write. When device is physically removed (thumb-drive unplugged) during this infinite loop, the code continues to loop infinitely. This is easily reproducible, and the lack of an appropriate error-exit is clear in the source code.

This thumb-drive operates normally and properly with Windows computers.

Problem 3: More Infinite Looping

In numerous places in the CMX code, there are infinite loops waiting for a certain status, unprotected by a timer. Sometimes, though I have not been able to consistently reproduce, this code loops infinitely. One example I provided Freescale follows (from USB_host.c):

```
/* Wait for transaction end or an error occurs */
while((MCF_USB_INT_STAT & (MCF_USB_INT_STAT_TOK_DNE | MCF_USB_INT_STAT_STALL
| MCF_USB_INT_STAT_ERROR)) ==0) {
  // DRN: INFINITE LOOP seen here...
  if (MCF_USB_INT_STAT & MCF_USB_INT_STAT_USB_RST) {
    evt_disconnect();
    tr_error=tre_disconnected;
```

```
            return((hcc_u16)-1u);
          }
        }
```

Problem 4: FAT32 Support

FAT32 is required to support thumb-drives larger than 2GB. A 4GB memory stick now costs $20 retail. Capacity/cost will continue to grow/shrink rapidly, and shortly all new thumb-drives will be FAT32. Consequently, any new products **must** support FAT32.

The CMX library provided is documented as not supporting FAT32. CMX offers for $4600 a license for FAT32 including sources. In fact, CMX has accidentally shipped the version of their library that does support FAT32, but we cannot depend on this for a new product (and certainly cannot use this library without the source code).

Freescale needs to provide FAT32 support for USB thumb-drive support to be workable for your customers. For my application, I have removed the CMX library and replaced it with an open-source implementation that supports FAT32 – however this will be additional work for all your customers.

Problem 5: CMX code not integrated with Codewarrior "Beans"

In the CodeWarrior tool, Freescale is providing a "Bean" system for rapid code-generation of common embedded code (initialization, timers, serial, etc). Unfortunately, the CMX examples provided do not use this system (in fact including multiple versions of initialization code, some not used). If customers are to use this as a basis for real applications, things like initialization, timers, and uart drivers need to be done in a manner compatible with the CodeWarrior tool.

Problem 6: Freescale Support

Folks at Freescale support have been helpful and courteous in their replies, however we have had no progress resolving these serious problems.

I contacted CMX and inquired about the USB software stack, and was told "We do not support the version of the code distributed by Freescale; our code is newer. You have to buy a license and use our current code to get any support". While I understand that position, Freescale is currently distributing code that is unworkable.

After reviewing a number of the above problems with Freescale support, I received the following from Eduardo, suggesting I rewrite the CMX code:

> In reply to your message regarding Service Request SR 1-456168576:
>
> Thank you for your response. I'm sorry but because this is a third party vendor code I am not able to edit it. I have started requesting the CMX people to give us an answer and we are actively working towards that goal, but I am unable to provide a quick solution other than propose that you eliminate the problem either by modifying the code to work as a state machine (instead of loops, only sample the status and update flags) or adding a time out routine so that the code breaks away from the loop if a certain time or number of loop iterations has occurred.
>
> I hope you understand that because CMX is a third party, we are unable to directly edit the code, please be sure we are working towards getting

> a better version out there.
>
> Should you need to contact us with regard to this message, please see the notes below.
>
> Best Regards,
> Eduardo

Then, the problem ticket was “automatically closed because of no activity for 7 days”. The problems have not been resolved though this ticket was opened almost two months ago.

Also, I reported a CodeWarrior bug: “CodeWarrior unusable on fine-pitch computer screens”. CodeWarrior draws its dialogs using a fixed-pixel height for many fonts and gadgets (without using normal Windows dialog processing that automatically scales fonts). Consequently, as modern PCs have ever-smaller pixels for better graphics and video display, the CodeWarrior screens are becoming impossible to see. The first tech support person who responded did not understand the issue and was unfamiliar with Windows mechanics. Another tech support person told me “Get an old low-resolution screen for use with CodeWarrior”. Finally, I received a note that this was logged as a “feature request” and the issue closed – but the bug is not fixed.

Summary

Freescale needs to address these issues promptly and seriously, or the Flexis solution cannot be used for our application, or any application requiring USB thumb-drive support. Customers like ourselves need working tool-chains and cannot afford to rewrite the USB-stack to use your products. The USB-stack must initialize reliably, read and write USB thumb-drives reliably, not go into infinite loops, and support FAT32.

Please let me know how you intend to resolve these issues, or we’ll need to scrap the work we’ve done with Freescale and try the Microchip PIC32 solution.

If I can be of any help with further information or clarification, please contact me at 978-263-0097 (East Coast) or at Dave.Nadler@Nadler.com

Thanks !