The Embedded Muse 4

Editor: Jack Ganssle (jack@ganssle.com) August 7, 1997

Dumb Mistakes

No lesson is learned more powerfully than those that come from making mistakes. The bigger the mistake the more meaningful the lesson. Bob Fehrenbach, a loyal Embedded Muse reader, suggested that I start passing along stories of dumb mistakes we've all made, in a spirit of sharing our pain to hopefully spare others the same. So, send your dumb mistakes to me (jack@ganssle.com) and I'll pass them along.

As one who has spent too much time in the abyss of disaster, here's a couple of my own stories:

A front panel switch on an analytical instrument was to activate the COPY button on a thermal printer. An associate enthusiastically sketched in wiring from the switch to an input port on the computer. Firmware would read the switch closure and then activate an output bit that pulled in a relay which commanded the printer to do its thing. I'll never forget the look on his face when I suggest just wiring the switch directly to the printer, bypassing the computer altogether…

But, I have to plead guilty to making far too many dumb mistakes myself. One that still hurts was in my life as a consultant in the early 80s. My partner and I were designing a Z80 based thickness gauge. Our customer had hired another consultant for a different portion of the project. At one meeting this individual suggested - strongly - that we design our code around an RTOS. Our egos were too fragile to take suggestions from a competitor (only through harsh experience do we learn some humility!), so we resisted and created mythical reasons for writing procedural code.

Without an RTOS the system met no performance targets and was impossible to maintain. We later spent far too much (of our own money) shoehorning an RTOS into the completed code. I still feel like an idiot when thinking about that experience.

Bob Fehrenbach had two stories of his own to pass along:

I once did a whole Z8 project using absolute register addresses rather than descriptive labels. The product worked fine. Maintenance was impossible.

(Editor's note: The Univac 1108's Exec 8 OS was even worse: I still shudder to think of the 100,000 lines of assembly code with label-less jumps - like JMP $+57).

Bob also added:

Several years ago I designed a unit that had a 16 digit LED display. The unit also included 16 keys. I used the digit select lines to scan the keys. A key press would raise the 'key' input to the processor if the corresponding digit was selected. So far nothing special.

Originally, when the software detected any key press, e.g. #5, it would initiate the debounce routine. The scan then continued and if it subsequently detected a different key, e.g. #9, the debounce routine started over. If TWO keys were pressed simultaneously, the software thought that the input was changing and never produced a valid key.

This was not a problem until we needed a function that required simultaneous key pressed. BUSTED!

The solution, of course, was to look for changes only after ALL keys were scanned. Seems obvious looking back. Why was this not obvious when the code was first written?

Moral: Scan ALL keys and only then look for changes.

Thought for the Week

Steve Litt (whose web site - www.troubleshooters.com - carries lots of neat hints about debugging systems) submitted an improved version of the World's Last C Bug code:

```
launches = 0;
 while (1)
 {
   status = GetRadarInfo();
   if (status = 1)
   {
     LaunchMissiles();
     launches++;
   }
   if (launches > 1)
     apologize();
 }
```

About The Embedded Muse

The Embedded Muse is an occasional newsletter sent via email by Jack Ganssle. Send complaints, comments, and contributions to him at jack@ganssle.com.

To subscribe, send a message to majordomo@ganssle.com, with the words "subscribe embedded *your-email-address*" in the body. To unsubscribe, change the message to "unsubscribe embedded *your-email-address*".

The Embedded Muse is supported by The Ganssle Group, whose mission is to help embedded folks get better products to market faster. We offer seminars at your site offering hard-hitting ideas - and action - you can take now to ***improve firmware quality and decrease development time***. Contact us at info@ganssle.com for more information.