ECE 4175
Project Three
QwikBug Checksum

Complete by:
Thursday Sept. 4th for an A+

References:
Chapter 6 PC Monitor Use

Overview

For this project, begin with the template code from your last project. Before you are done, strip out the parts (most of it) that do not pertain to this project.

A question has arisen in conjunction with Microchip Technology's production of the Qwik&Low board. How can we tell that the QwikBug kernal, located in the PIC18LF4321's program memory beginning at address 0x1A00 and ending at address 0x1FFF, has been loaded with the correct version? The answer that you will develop here is to form a checksum of the content of these addresses as a sixteen-bit number and then to send the resulting four-hex-character number to the PC for display followed by a carriage return code, 0x0D, and a line feed code, 0x0A. Then sleep.

Accessing Program Memory

Define a pointer into program memory with
```
const char rom *RomPtr;        // Pointer to one-byte ROM elements
```

and a two-byte checksum variable with
```
unsigned int CHECKSUM;         // Sixteen-bit checksum
```

Add the byte pointed to by RomPtr into CHECKSUM and then increment RomPtr with
```
CHECKSUM  += *RomPtr++         // Add into checksum
```

Note that when you have completed the formation of CHECKSUM, you can break out the nibbles as exemplified by the following line to move the upper four bits to NIBBLE:
```
NIBBLE = (CHECKSUM & 0xF000) >> 12;  // Get upper four bits; leave CHECKSUM unchanged
```

Nibble-to-ASCII conversion

Define a constant table of ASCII characters in program memory with:
```
const char rom FormHex[] = "0123456789ABCDEF";
```
and unsigned byte variables NIBBLE and ASCII with
```
unsigned char NIBBLE;
unsigned char ASCII;
```

If NIBBLE contains a value between 0 and 15, then ASCII will contain the corresponding ASCII-coded character after executing
```
ASCII = FormHex[NIBBLE];
```