

## Creating a C program for the Dragon12-Plus

1. Install GCC first, and then the EmbeddedGNU IDE. GCC, program `gnu-68hc1x-2.2.exe`, installs into `C:\usr`. EmbeddedGNU doesn't have an installer. Create the folder `C:\EmbeddedGNU` and extracting the files there. Add a link to `C:\EmbeddedGNU\EmbeddedGNU.exe` to your Start Menu or desktop for easy access.
2. When you start EmbeddedGNU, you will be given a chance to configure your serial port. In general, EmbeddedGNU works the same as AsmIDE. They both were written by the same person. For convenience in finding the header files, Start EmbeddedGNU and go to Options->Environment. In the first, Directories, pane, check the box "Add the directory below to be searched for include files" and type into the box "`c:\EmbeddedGNU\include\`".
3. Create a folder for your project off of `C:\` (not in "My Documents").
4. Start EmbeddedGNU and create a new project in your project folder. Set the Hardware Profile to "Dragon12". You will need to edit the Dragon12 profile, so click on "Edit Profile" rather than "OK" at this time. In the Linker Search Directory field, change the directory to "`c:\usr\lib\gcc-lib\m6811-elf\3.0.4\mshort`". You will only have to edit this profile when making your first project.
5. Create a new source file and save it with the ".c" extension in your project folder. Saving the file after creating is important since until then it has no name or type.
6. The function, `main`, will be executed when your program starts. Your program should not return from `main`. Do not call any C library functions. Do not use any floating point expressions.

See the textbook for an example on how to create interrupt service routines and properly initialize them.

Remember that C performs arithmetic on 16 bit integers even if you are using 8 bit values.

Be careful in the use of signed versus unsigned values. C will use the correct comparisons, shifts, and multiply/divide operations for the operands, but you must declare variables for the correct type.

Read the textbook C examples!

Here is an example program which implements last year's Lab 2 in EET363:

```

/* Lab 2 example, Tom Almy, August 2009 */
/* Note that we don't have access to the DBUG12 routines to read and
write characters like we have using assembler, so the functions
are implemented here. */

/* Always have these two lines in your program: */
#include "hcs12.h"
#include "vectors12.h"

/* A couple of useful constants */
#define true (1)
#define false (0)

/** Serial Port communication with PC */
/** Initialize the serial port */
void sciinit(void) {
    SCI0BDL = 156; // 9600 bps
    SCI0CR2 = TE | RE; // TE RE
}

/** Write a character to the serial port */
unsigned char putchar(unsigned char val) {
    while ((SCI0SR1 & TDRE) == 0) {}; // Wait for transmitter empty
    SCI0DRL = val;
    return val;
}

/** Read a character from the serial port */
unsigned char getchar(void) {
    while ((SCI0SR1 & RDRF) == 0) {}; // Wait for receiver full
    return SCI0DRL;
}

/** The code to implement the assignment goes here: */

/** The main function, from which we will never return */
int main(void) {
    int fahrenheit = -40;
    int celsius;
    unsigned char result[6];
    unsigned int value;
    int p;

    sciinit(); /* Actually, DBUG12 does this for us */

    /* First -- convert from Fahrenheit to Celsius */
    celsius = ((fahrenheit-32)*5L)/9;

    /* Then convert to the character string */

    value = celsius < 0 ? -celsius : celsius; /* get absolute value */
    p = 5;
    do {
        result[p--] = (value % 10) + '0'; /* next digit character*/
        value /= 10; /* divide value by 10 */
    }

```

```
} while (value != 0);

if (celsius < 0) {
    result[p--] = '-'; /* leading minus sign if negative */
}
while (p >= 0) {
    result[p--] = ' '; /* fill with leading space characters */
}

/* Print decimal value of ASCII character, CR, and LF */

for (p = 0; p < 6; p++) {
    putchar(result[p]);
}
putchar('\r');
putchar('\n');

/** When main returns, it goes into a loop - it won't go back
    to DEBUG12 **/
}
```