/\*

\* This file is responsible for drawing sensor bar graphs.

\*/

#include <pololu/3pi.h>

#include <avr/pgmspace.h>

// Data for generating the characters used in load\_custom\_characters

// and display\_readings. By reading levels[] starting at various

// offsets, we can generate all of the 7 extra characters needed for a

// bargraph. This is also stored in program space.

const char levels[] PROGMEM = {

0b00000,

0b00000,

0b00000,

0b00000,

0b00000,

0b00000,

0b00000,

0b11111,

0b11111,

0b11111,

0b11111,

0b11111,

0b11111,

0b11111

};

// This function loads custom characters into the LCD. Up to 8

// characters can be loaded; we use them for 7 levels of a bar graph.

void load\_custom\_characters()

{

lcd\_load\_custom\_character(levels+0,0); // no offset, e.g. one bar

lcd\_load\_custom\_character(levels+1,1); // two bars

lcd\_load\_custom\_character(levels+2,2); // etc...

lcd\_load\_custom\_character(levels+3,3);

lcd\_load\_custom\_character(levels+4,4);

lcd\_load\_custom\_character(levels+5,5);

lcd\_load\_custom\_character(levels+6,6);

clear(); // the LCD must be cleared for the characters to take effect

}

// This function displays the sensor readings using a bar graph.

void display\_readings(const unsigned int \*calibrated\_values)

{

unsigned char i;

for(i=0;i<5;i++) {

// Initialize the array of characters that we will use for the

// graph. Using the space, an extra copy of the one-bar

// character, and character 255 (a full black box), we get 10

// characters in the array.

const char display\_characters[10] = {' ',0,0,1,2,3,4,5,6,255};

// The variable c will have values from 0 to 9, since

// calibrated values are in the range of 0 to 1000, and

// 1000/101 is 9 with integer math.

char c = display\_characters[calibrated\_values[i]/101];

// Display the bar graph character.

print\_character(c);

}

}

// Local Variables: \*\*

// mode: C \*\*

// c-basic-offset: 4 \*\*

// tab-width: 4 \*\*

// indent-tabs-mode: t \*\*

// end: \*\*