```
#ifndef LCDG_H
#define LCDG_H
//*****************LCDG.h*************************************
// implementation of the driver for the AGM1264F MODULE
// Jonathan W. Valvano 11/21/09

//  This example accompanies the books
//   "Embedded Microcomputer Systems: Real Time Interfacing",
//         Engineering, copyright (c) 2006,
//   "Introduction to Embedded Microcomputer Systems:
//    Motorola 6811 and 6812 Simulation", Thomson, copyright (c) 2002

// Copyright 2009 by Jonathan W. Valvano, valvano@mail.utexas.edu
//    You may use, edit, run or distribute this file
//    as long as the above copyright notice remains

// Hardware:
// gnd    =  1- AGM1264F ground
// +5V    =  2- AGM1264F Vcc (with 0.1uF cap to ground)
// pot    =  3- AGM1264F Vo  (center pin of 10k pot)
// PP2    =  4- AGM1264F D/I (0 for command, 1 for data)
// gnd    =  5- AGM1264F R/W (blind cycle synchronization)
// PP3    =  6- AGM1264F E   (1 to latch in data/command)
// PH0    =  7- AGM1264F DB0
// PH1    =  8- AGM1264F DB1
// PH2    =  9- AGM1264F DB2
// PH3    = 10- AGM1264F DB3
// PH4    = 11- AGM1264F DB4
// PH5    = 12- AGM1264F DB5
// PH6    = 13- AGM1264F DB6
// PH7    = 14- AGM1264F DB7
// PP0    = 15- AGM1264F CS1 (controls left half of LCD)
// PP1    = 16- AGM1264F CS2 (controls right half of LCD)
// +5V    = 17- AGM1264F RES (reset)
// pot    = 18- ADM1264F Vee (-10V)
// 10k pot from pin 18 to ground, with center to pin 3
// references   http://www.azdisplays.com/prod/g1264f.php
// sample code  http://www.azdisplays.com/PDF/agm1264f_code.pdf
// data sheet   http://www.azdisplays.com/PDF/agm1264f.pdf

// BUG NOTICE 11/11/09 -Valvano
// When changing from right to left or from left to right
//   the first write with data=0 goes to two places
// One can reduce the effect of this bug by
// 1) Changing sides less often
// 2) Ignore autoincrement, and set column and page address each time
// 3) Blanking the screen then write 1's to the screen

//*********************************************************************

// to use it as an 8-line by 21-character display
// initialize it once using
//      LCD_Init
// clear screen with
//      LCD_Clear
// set cursor position using
//      LCD_GoTo
// place ASCII on the screen using
//      LCD_OutChar
//      LCD_OutString
//      LCD_OutDec
//      LCD_OutFix1
//      LCD_OutFix2
//      LCD_OutFix3

// ********* LCD_Init************
// Initialize AGM1264F 128-bit by 64-bit graphics display
// activates TCNT at 1.5 MHz, assumes PLL active
// Input: none
// Output: none
// does not clear the display
void LCD_Init(void);


// ********* LCD_Clear************
// Clear the entire 1024 byte (8192 bit) image on the
//    AGM1264F 128-bit by 64-bit graphics display
// Input: value to write into all bytes of display RAM
```

```
 79: // Output: none
 80: // e.g., LCD_Clear(0);  // makes all pixels off
 81: void LCD_Clear(unsigned char data);
 82:
 83: //-----------------------LCD_GoTo-----------------------
 84: // Move cursor
 85: // Input: line number is 1 to 8, column from 1 to 21
 86: // Output: none
 87: // errors: it will ignore legal addresses
 88: void LCD_GoTo(int line, int column);
 89:
 90:
 91: // ********* LCD_OutChar***********
 92: // Output ASCII character on the
 93: //    AGM1264F 128-bit by 64-bit graphics display
 94: // Input: 7-bit ASCII to display
 95: // Output: none
 96: // letter must be between 32 and 127 inclusive
 97: // execute LCD_GoTo to specify cursor location
 98: void LCD_OutChar(unsigned char letter);
 99:
100:
101: //---------------------LCD_OutString--------------
102: // Display String
103: // Input: pointer to NULL-terminationed ASCII string
104: // Output: none
105: void LCD_OutString(char *pt);
106:
107: void LCD_DrawGrid(unsigned char field[10][10]);
108:
109: #endif
```