```
/*

 Modified to output to 16 x 2 LCD Display - Glen Popiel KW5GP

 Based on HMC5883L_Example.pde - Example sketch for integration with an
HMC5883L triple axis magnetometer. Copyright (C) 2011 Love Electronics
(loveelectronics.co.uk)

 This program is free software: you can redistribute it and/or modify
 it under the terms of the version 3 GNU General Public License as
 published by the Free Software Foundation.

 This program is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.

 You should have received a copy of the GNU General Public License
 along with this program.  If not, see <http://www.gnu.org/licenses/>.

 */

#include <Wire.h>  // Use the I2C Library
#include <HMC5883L.h>  // Use the HMC5883L Compass Library
#include <LiquidCrystal_I2C.h>  // Use the Liquid Crystal I2C Library

const int lcd_end = 16; // set width of LCD
const int lcd_address = 0x27; // I2C LCD Address
const int lcd_lines = 2; // Number of lines on LCD

HMC5883L compass;  // Store our compass as a variable.
int error = 0;  // Record any errors that may occur in the compass.
String Direction;  // Store the Direction Text as a variable

// set the LCD I2C address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(lcd_address,lcd_end,lcd_lines);

void setup()
{
  Wire.begin(); // Start the I2C interface.

  lcd.init();  // initialize the LCD
  lcd.backlight();  // Turn on the LCD backlight
  lcd.home();  // Set the cursor to line 0, column 0

  compass = HMC5883L(); // Construct a new HMC5883 compass.

  error = compass.SetScale(1.3); // Set the scale of the compass to +/-
1.3Ga.
  if(error != 0) // If there is an error, print it out.
  {
    error = compass.SetMeasurementMode(Measurement_Continuous); // Set
the measurement mode to Continuous
  }
```

```
}  // End Setup Loop


void loop()
{
  // Retrieve the raw values from the compass (not scaled).
  MagnetometerRaw raw = compass.ReadRawAxis();

  // Retrieved the scaled values from the compass (scaled to the
configured scale).
  MagnetometerScaled scaled = compass.ReadScaledAxis();

  // Values are accessed like so:
  int MilliGauss_OnThe_XAxis = scaled.XAxis;// (or YAxis, or ZAxis)

  // Calculate heading when the magnetometer is level, then correct for
signs of axis.
  float heading = atan2(scaled.YAxis, scaled.XAxis);

  // Once you have your heading, you must then add your 'Declination
Angle', which is the 'Error' of the magnetic field in your location.
  // Find yours here: http://www.magnetic-declination.com/

  // If you cannot find your Declination, comment out these two lines,
your compass will be slightly off.
  float declinationAngle = 0.0169296937;  // Declination for Southaven,
MS
  heading += declinationAngle;

  if(heading < 0)  // Correct for when signs are reversed.
  {
    heading += 2*PI;
  }

  if(heading > 2*PI)  // Check for wrap due to addition of declination.
  {
    heading -= 2*PI;
  }

  float headingDegrees = heading * 180/M_PI;   // Convert radians to
degrees for readability.

  Output(raw, scaled, heading, headingDegrees);  // Output the data to
the LCD

  delay(1000);  // Update the LCD every second

}

// Output the data to the LCD
void Output(MagnetometerRaw raw, MagnetometerScaled scaled, float
heading, float headingDegrees)
{
  lcd.clear();
```

```
  lcd.print("^   " );  // Print an up arrow to indicate compass pointer
  lcd.print(headingDegrees,1);  // Display the Heading in degrees
  lcd.print(" Deg");
  lcd.setCursor(15,0);
  lcd.print("^");

  // Calculate Direction
  Direction = " ";
  lcd.setCursor(6,1);

  if (headingDegrees >= 348.75 | headingDegrees <11.25)  // Direction =
North
  {
    Direction = " N";
  }
  if (headingDegrees >= 11.25 && headingDegrees <33.75)  // Direction =
North North East
  {
    Direction = "NNE";
  }
  if (headingDegrees >= 33.75 && headingDegrees <56.25)  // Direction =
North East
  {
    Direction = "NE";
  }
  if (headingDegrees >= 56.25 && headingDegrees <78.75)  // Direction =
East North East
  {
    Direction = "ENE";
  }
  if (headingDegrees >= 78.75 && headingDegrees <101.25)  // Direction =
East
  {
    Direction = " E";
  }
  if (headingDegrees >= 101.25 && headingDegrees <123.75)  // Direction =
East South East
  {
    Direction = "ESE";
  }
  if (headingDegrees >= 123.75 && headingDegrees <146.25)  // Direction =
South East
  {
    Direction = "SE";
  }
  if (headingDegrees >= 146.25 && headingDegrees <168.75)  // Direction =
South South East
  {
    Direction = "SSE";
  }
  if (headingDegrees >= 168.75 && headingDegrees <191.25)  // Direction =
South
  {
    Direction = " S";
```

```
  }
  if (headingDegrees >= 191.25 && headingDegrees <213.75)  // Direction =
South South West
  {
    Direction = "SSW";
  }
  if (headingDegrees >= 213.75 && headingDegrees <236.25)  // Direction =
South West
  {
    Direction = "SW";
  }
  if (headingDegrees >= 236.25 && headingDegrees <258.75)  // Direction =
West South West
  {
    Direction = "WSW";
  }
  if (headingDegrees >= 258.75 && headingDegrees <281.25)  // Direction =
West
  {
    Direction = " W";
  }
  if (headingDegrees >= 281.25 && headingDegrees <303.75)  // Direction =
West North West
  {
    Direction = "WNW";
  }
  if (headingDegrees >= 303.75 && headingDegrees <326.25)  // Direction =
North West
  {
    Direction = "NW";
  }
  if (headingDegrees >= 326.25 && headingDegrees <348.75)  // Direction =
North North West
  {
    Direction = "NNW";
  }

  lcd.print(Direction);  // Display the direction on the LCD

}
```