/\*

"Code Blue"

PA GOVERNOR'S STEM COMPETITION

BSHS STEM Team Leader - Matt Horger

Alpha - 10/12/15 - initialized file

Beta Pre-Build 1.0 - 12/23/15 - Libraries installed, instances and declarations made

Beta Pre-Build 1.1 - 1/6/2016 - working temperature, GPRS Shield, TCP Connection

Beta Pre-Build 1.2 - 1/7/2016 - Added GPS Functionality - very low memory - will have to compress

Beta Pre-Build 1.3 - 1/12/2016 - Need to send GPS coordinates via TCP Connection - memory is compressed

Version 1.0 - 1/15/2016 - Fully Compiled and Working GPS / GPRS / Temperature - 1,496 bytes used out of 2,048 bytes (74%)

\*/

#include <Adafruit\_GPS.h>

#include <DHT.h>

#include <GPRS\_Shield\_Arduino.h>

#include <SoftwareSerial.h>

#define GPSECHO false // GPS DEBUG

#define DHTPIN 11 // temperature pin

#define DHTTYPE DHT11 // DHT11 model

#define PIN\_TX 7 // GPRS pin tx

#define PIN\_RX 8 // GPRS pin rx

#define BAUDRATE 19200 // comm. rate

char buffer[40]; // string to send our variables

char flat[10]; // variable to store latitude

char flon[10]; // variable to store longitude

void useInterrupt(boolean); // void method that returns the refresh rate

boolean usingInterrupt = false; // are we using the method?

uint32\_t timer = millis(); // timer in milliseconds

GPRS gprs(PIN\_TX,PIN\_RX,BAUDRATE); // initialize gprs shield

DHT dht(DHTPIN, DHTTYPE); // initialize temperature sensor

SoftwareSerial mySerial(3, 2); // define our GPS constructors

Adafruit\_GPS GPS = (&mySerial); // initialize GPS

void setup() {

Serial.begin(115200); // begin serial data

GPS.sendCommand(PMTK\_SET\_NMEA\_OUTPUT\_RMCGGA); // format GPS output

GPS.sendCommand(PMTK\_SET\_NMEA\_UPDATE\_1HZ); // GPS refresh rate - 1 Hertz

useInterrupt(true); // return the refresh rate!

while(!gprs.init()) {

delay(1000);

Serial.print("init error\r\n"); // if GPRS can't initialize, throw an error

}

delay(4000);

Serial.println("gprs init success"); // if GPRS can initialize, tell us

if(!gprs.connect(TCP,"198.168.1.106",32000)) {

Serial.println("connect to server success"); // if GPRS can connect to the server, tell us

}

dht.begin(); // begin temperature gathering

}

void useInterrupt(boolean v) {

if (v) {

OCR0A = 0xAF;

TIMSK0 |= \_BV(OCIE0A);

usingInterrupt = true;

} else {

TIMSK0 &= ~\_BV(OCIE0A);

usingInterrupt = false;

}

}

SIGNAL(TIMER0\_COMPA\_vect) {

char c = GPS.read();

#ifdef UDR0

if (GPSECHO)

if (c) UDR0 = c;

#endif

}

void loop(){

GPS.begin(9600); // start our GPS communication

// Wait a few seconds between measurements.

delay(3000);

// Read temperature as Fahrenheit (isFahrenheit = true)

int f = dht.readTemperature(true);

// Check if any reads failed and exit early (to try again).

if (isnan(f)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

if (! usingInterrupt) {

// read data from the GPS in the 'main loop' above

char c = GPS.read();

if (GPSECHO) // if debug is true, print debug statement

if (c) Serial.print(c);

}

if (GPS.newNMEAreceived()) { // if the attempt to get new coordinates fails

if (!GPS.parse(GPS.lastNMEA())) // this takes our last set and parses it through our program

return;

}

Serial.print("Temperature: "); // output temperature

Serial.print(f);

Serial.print(" \*F\n");

Serial.print("Fix: "); Serial.print((int)GPS.fix); // output GPS satellite fix

Serial.print(" Quality: "); Serial.println((int)GPS.fixquality); // output satellite quality

if (GPS.fix) { // if we have a fix of our GPS OR if(f<32) = temperature is lower than freezing

Serial.print("Location (Lat, Long): "); // print coordinates

Serial.print(GPS.latitudeDegrees, 4);

dtostrf(GPS.latitudeDegrees, 12, .01, flat); // parse our float to our char array

Serial.print(", ");

Serial.println(GPS.longitudeDegrees, 4);

dtostrf(GPS.longitude, 12, .01, flon); // parse our float to our char array

Serial.println("Sending coordinates..."); // send the coordinates to the server.

gprs.send(flat, sizeof(flat)-1);

gprs.send(flon, sizeof(flon)-1);

}

delay(4000); // wait roughly 15 seconds before information is gathered again

}