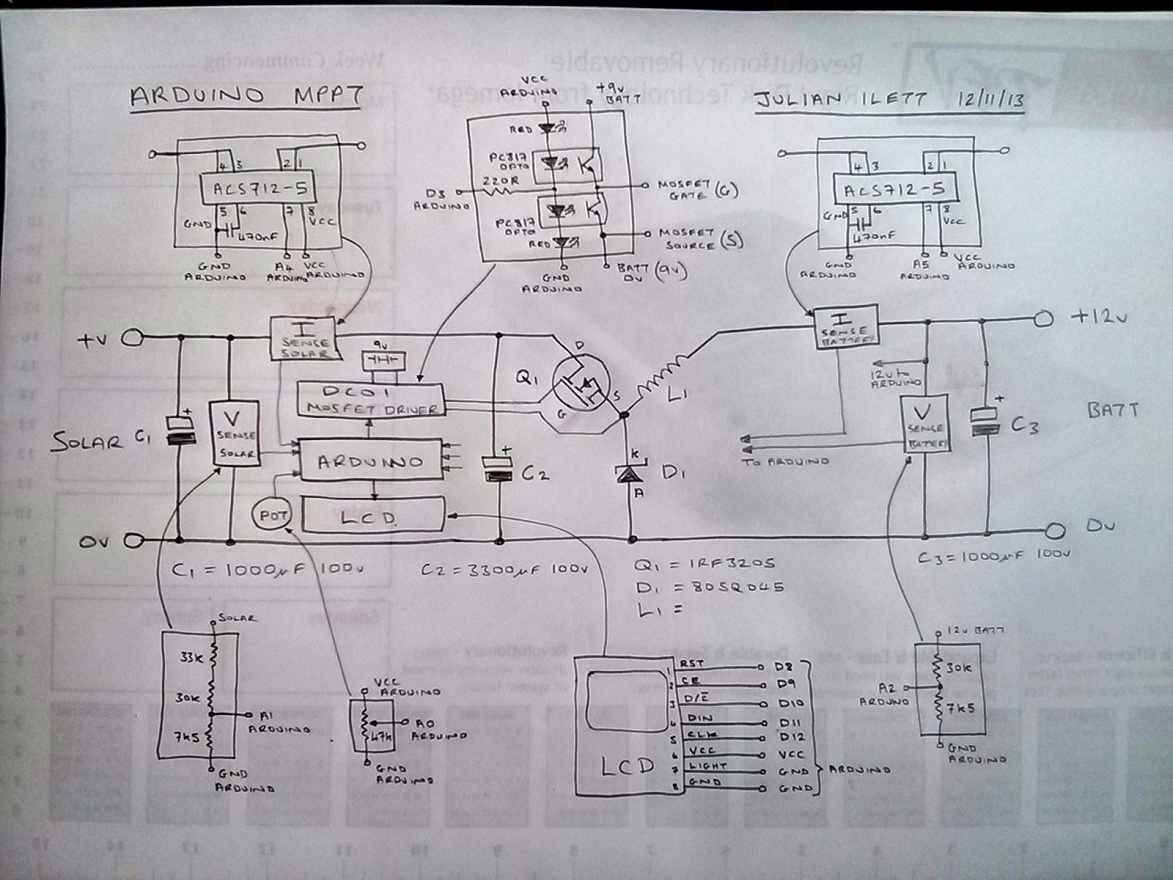
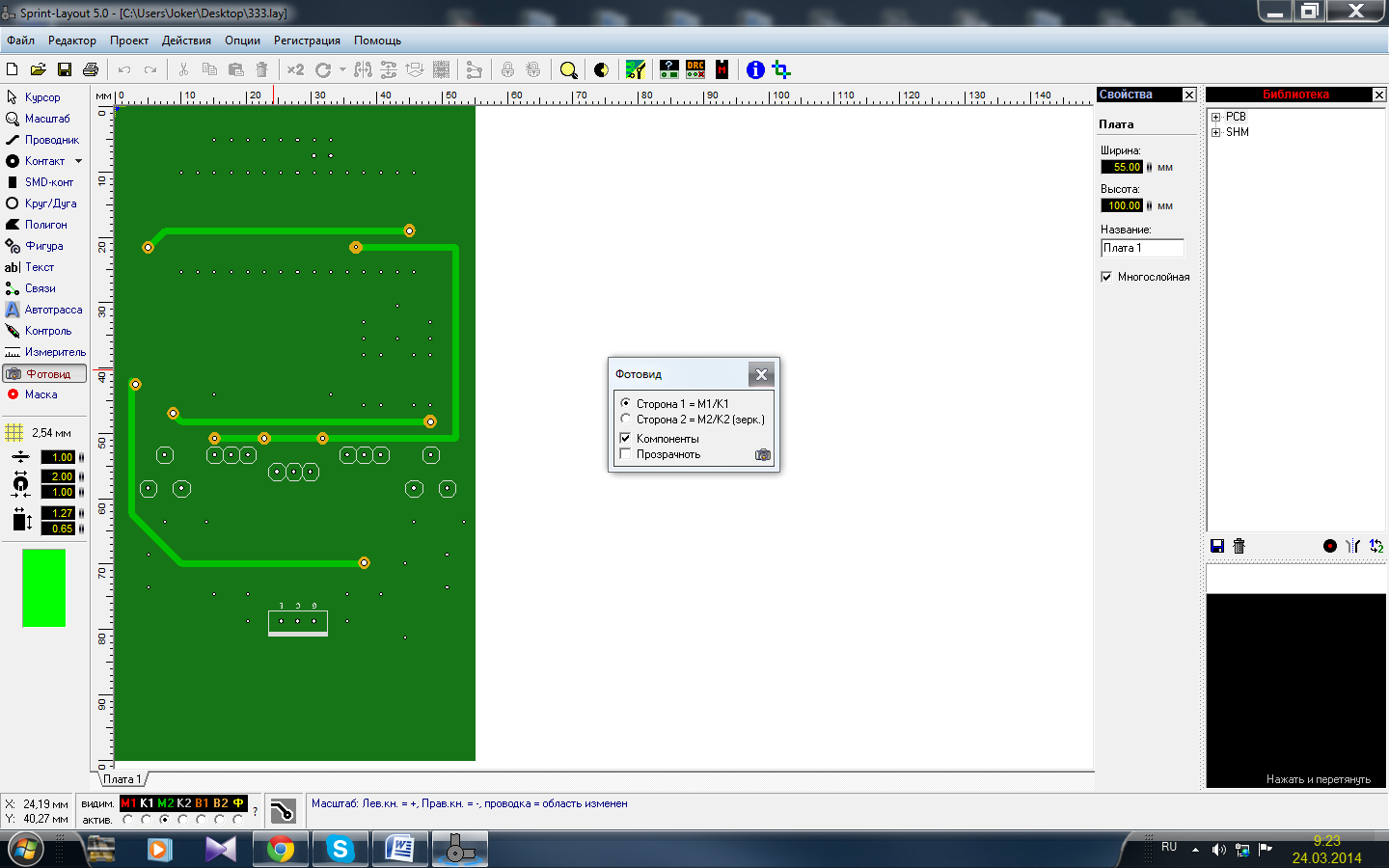
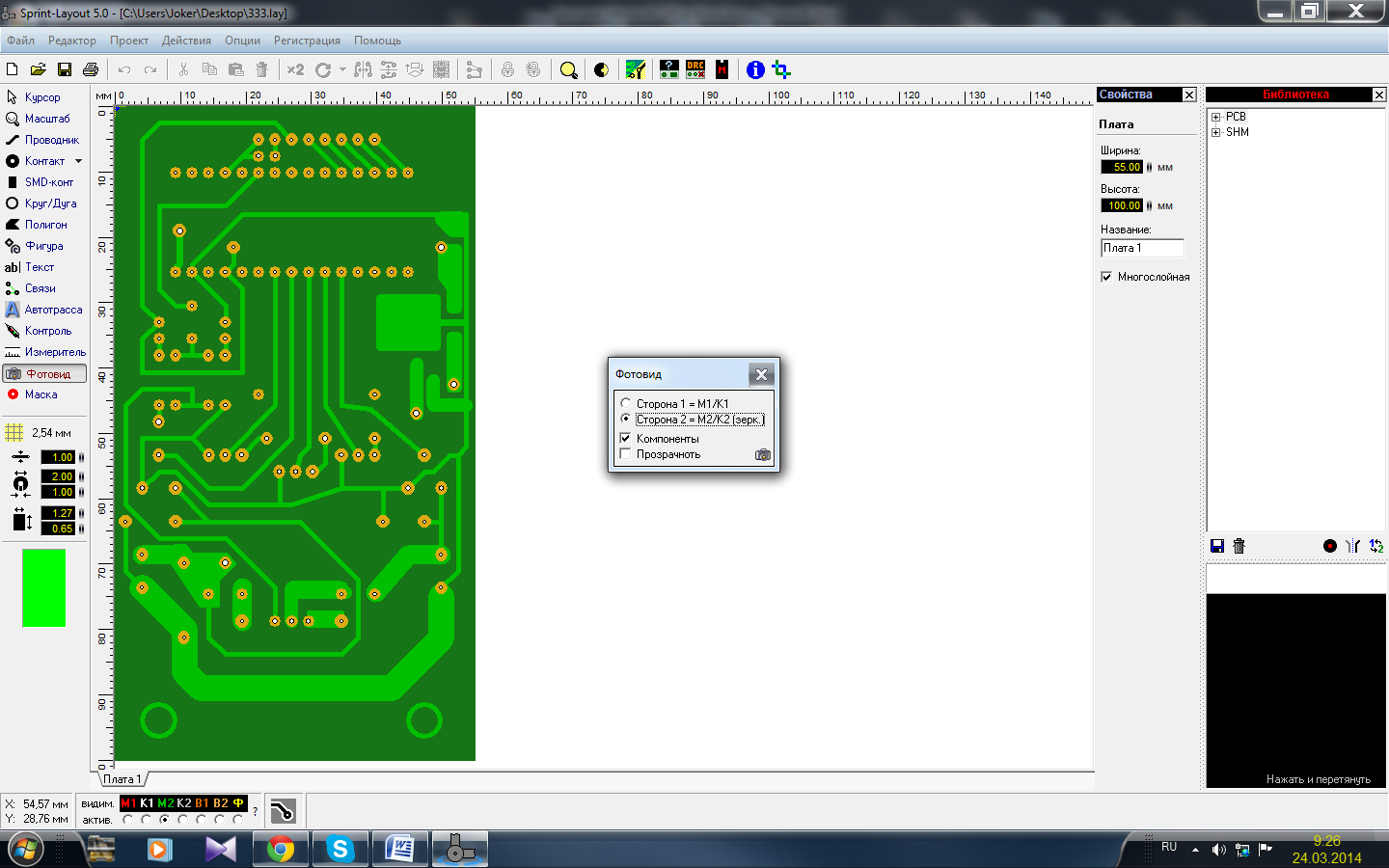
Итак, начнем с со схемы:



Впринципе тут все понятно и довольно доступно расписано. А какие изменения были внесены мною в схему, я опишу ниже!

Далее переходим к печатной плате и тому, что на него установлено:

17

16

15

14

13

12

11

10

9

8

7

6

5

4

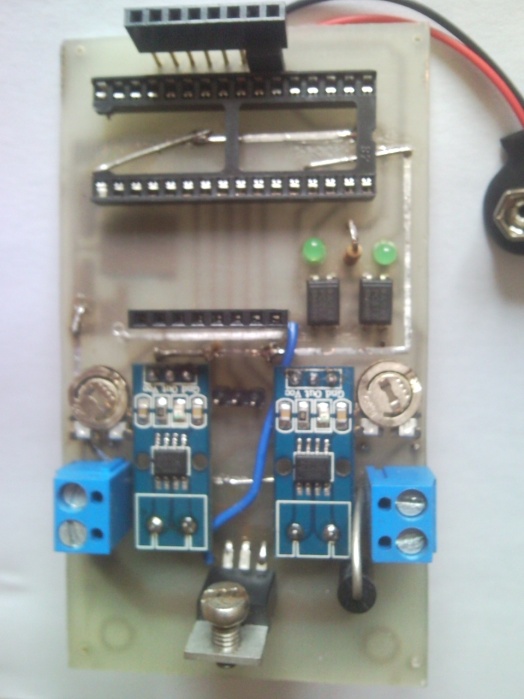
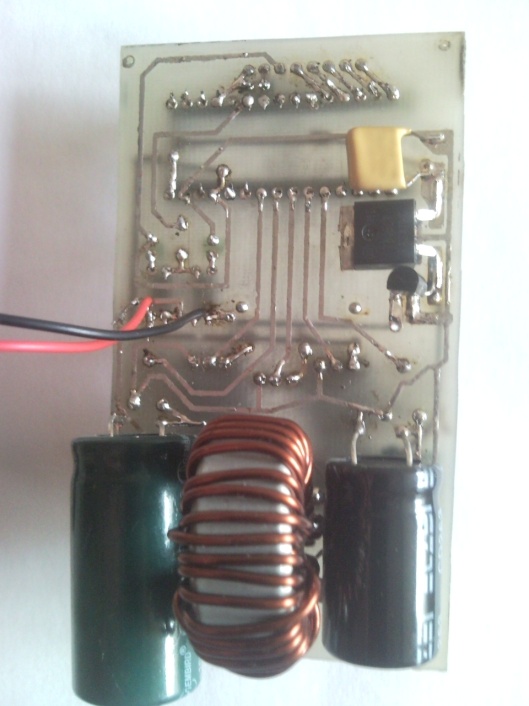
3

2

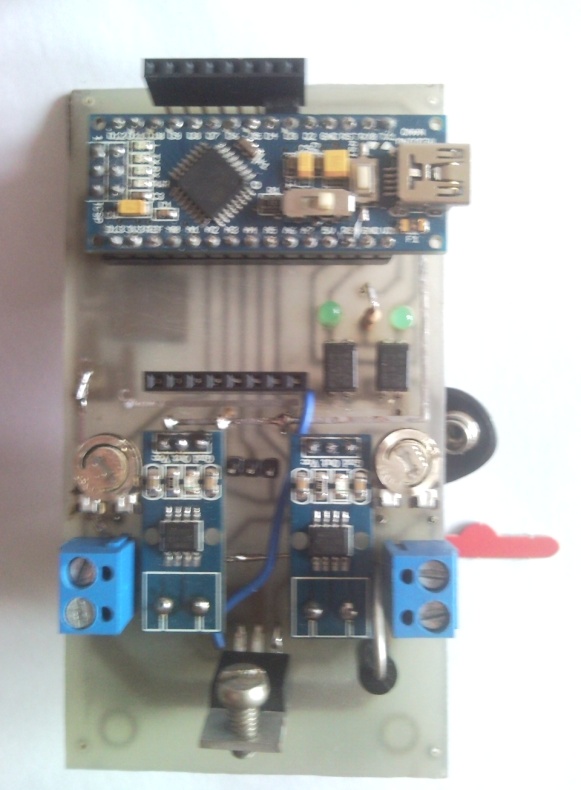
1

1. Ардуино Нано(на плату установлен разъем для микросхемы на 32 ножки, две ножки подогнуты)
2. Шилд с экраном Nokia 5110(на плату установлен разъем с длиннымы ножками)
3. Опторазвязка (оптопара РС817 – 2 шт, светодиод 3мм - 2шт, резистор 330Ом )
4. Разъем для переменного резистора 10-47кОм
5. Подстроечный резистор 47-68кОм для измерения выходного напряжения(вместо модуля измерения напряжения)
6. Подстроечный резистор 47-68кОм для измерения входного напряжения(вместо модуля измерения напряжения)
7. Модуль измерения Тока на входе в контроллер ACS712-20
8. Модуль измерения Тока на вsходе в контроллер ACS712-20
9. Полевой транзистор IRF3205
10. Выходная колодка
11. Входная колодка
12. Конденсатор 1000мкф х 50в
13. Конденсатор 3300мкф х 50в
14. Стабилизатор напряжения 7805 для питания ардуины и экрана
15. Ферритовый дроссель(у меня намотано 33витка провод диаметром 1.25мм)
16. Стабилизатор напряжения 77L09 для открывания транзистора(вместо «кроны»!)
17. Перемычка для отключения подсветки экрана.

В реале выглядит так:

Вид спереди Вид сзади

С ардуиной С экраном

Теперь переходим к скетчу для ардуины. Выглядит он так:

// Wiring:

//A3 - Pot

//A1 - Volts (solar)

//A4 - Volts (battery)

//A2 - Amps (solar)

//A5 - Amps (battery)

//D3 - PWM out to DCOI

//D12 -LCD\_RST (1)

//D11 - LCD\_CE (2)

//D10 - LCD\_DC (3)

//D9 - LCD\_DIN (4)

//D8 - LCD\_CLK (5)

//-----LCD\_VCC (6)

//---LCD\_LIGHT (7)

//-----LCD\_GND (8)

#define LCD\_C LOW

#define LCD\_D HIGH

#define PIN\_RESET 12

#define PIN\_SCE 11

#define PIN\_DC 10

#define PIN\_SDIN 9

#define PIN\_SCLK 8

int a1 =11;

//#include <stdlib.h>;

#include "header.h";

#include <PWM.h>

int32\_t frequency = 15000; //frequency (in Hz)

int sensorValue = 0;

float panelVolts = 0;

float batteryVolts = 0;

float lastPanelVolts = 0;

float lastBatteryVolts = 0;

float panelAmps = 0;

float lastPanelAmps = 0;

float batteryAmps = 0;

float lastBatteryAmps = 0;

float efficiency = 0;

float siemens = 0;

float panelWatts = 0;

float lastPanelWatts = 0;

float batteryWatts = 0;

float maxwatts = 0;

int barwatts = 0;

float Voc = 0;

float Isc = 0;

char tmp[25];

//boolean gotVoc = false;

//boolean gotIsc = false;

boolean highWatts = false;

int chartX = 0;

int chartY = 0;

//int bin = 0;

boolean mpptOn = false;

const boolean initialise = true;

void LcdInitialise(void)

{

pinMode(PIN\_SCE, OUTPUT);

pinMode(PIN\_RESET, OUTPUT);

pinMode(PIN\_DC, OUTPUT);

pinMode(PIN\_SDIN, OUTPUT);

pinMode(PIN\_SCLK, OUTPUT);

digitalWrite(PIN\_RESET, LOW);

digitalWrite(PIN\_RESET, HIGH);

LcdWrite(LCD\_C, 0x21 ); // LCD Extended Commands.

LcdWrite(LCD\_C, 0xBC ); // Set LCD Vop (Contrast).

LcdWrite(LCD\_C, 0x04 ); // Set Temp coefficent. //0x04

LcdWrite(LCD\_C, 0x14 ); // LCD bias mode 1:48. //0x13

LcdWrite(LCD\_C, 0x0C ); // LCD in normal mode.

LcdWrite(LCD\_C, 0x20 );

LcdWrite(LCD\_C, 0x0C );

}

void LcdClear()

{

for (int i=0; i<504; i++) LcdWrite(LCD\_D, 0x00);

}

void LcdWrite(byte dc, byte data)

{

digitalWrite(PIN\_DC, dc);

digitalWrite(PIN\_SCE, LOW);

shiftOut(PIN\_SDIN, PIN\_SCLK, MSBFIRST, data);

digitalWrite(PIN\_SCE, HIGH);

}

void LcdString(char \*characters, int x, int y)

{

LcdWrite(LCD\_C, 0x80 | x); // Column.

LcdWrite(LCD\_C, 0x40 | y); // Row.

while (\*characters) LcdCharacter(\*characters++);

}

void LcdCharacter(char character)

{

for (int index = 0; index < 5; index++) LcdWrite(LCD\_D, ASCII[character - 0x20][index]);

LcdWrite(LCD\_D, 0x00);

}

void LcdXY(int x, int y)

{

LcdWrite(LCD\_C, 0x80 | x); // Column.

LcdWrite(LCD\_C, 0x40 | y); // Row.

}

void LcdPlot (int x, int y)

{

// static int lastX;

// static int lastY;

LcdXY(x,5-(y/8)); //set display address for plot

int bin=128;

for (int q=0; q<y%8; q++) bin=bin/2; //calculate pixel position in byte

LcdWrite(LCD\_D, bin); //plot pixel

float slope=float(47-y)/float(83-x);

for (int j=x+2; j<84; j++) {

float dy=slope\*float(j-x);

int k=y+round(dy);

LcdXY(j,5-(k/8)); //set display address for plot

int bin=128;

for (int q=0; q<k%8; q++) bin=bin/2; //calculate pixel position in byte

LcdWrite(LCD\_D, bin); //plot pixel

}

// lastX = x;

// lastY = y;

}

void perturb(boolean init=false)

{

static byte pulseWidth = 0;

static boolean trackDirection = false; //counts down / pwm increases

// static int loopCounter = 0;

if (init) {

pulseWidth = 255;

trackDirection = false;

}

else {

if (!trackDirection) {

if (pulseWidth != 0) {pulseWidth--;} else {trackDirection = true;}

}

else {

if (pulseWidth != 255) {pulseWidth++;} else {trackDirection = false;}

}

}

pwmWrite(3, pulseWidth); //write perturbed PWM value to PWM hardware

if ((panelWatts - lastPanelWatts) < -0.1) trackDirection = !trackDirection;

}

void setup()

{

pinMode(a1, OUTPUT);

digitalWrite(a1, LOW);

LcdInitialise();

LcdClear();

LcdString("Solar",0,0);

LcdString("MPPT",3,1);

LcdString("B",31,2);

LcdString("V",31,3);

LcdString("Wp",65,3);

LcdString("A",31,4);

LcdString("W",65,4);

InitTimersSafe();

bool success = SetPinFrequencySafe(3, frequency);

if(success) {

pinMode(13, OUTPUT);

digitalWrite(13, HIGH);

}

}

void loop()

{

panelVolts = analogRead(A1) \* 0.0163; //get the volts

panelVolts = (panelVolts + lastPanelVolts) / 2; //average the volts

lastPanelVolts = panelVolts;

LcdString(dtostrf(panelVolts,5,2,&tmp[0]),0,3); //display the volts

panelAmps = (analogRead(A2)-516) \* 0.0337\*1,5; //get the panelAmps

panelAmps = (panelAmps + lastPanelAmps) / 2; //average the panelAmps

lastPanelAmps = panelAmps;

if (panelAmps < 0.02) panelAmps = 0; //don't let the panelAmps go below zero

LcdString(dtostrf(panelAmps,5,2,&tmp[1]),0,4); //display the panelAmps

panelWatts = panelVolts \* panelAmps; //calculate the watts

LcdString(dtostrf(panelWatts,4,1,&tmp[2]),40,4); //display the panel watts

sensorValue = analogRead(A3);

if (sensorValue > 1020) {

// pwmWrite(3, sensorValue / 4);

if (mpptOn) {

perturb();

}

else {

mpptOn = true;

perturb(initialise); //initialise the perturb algorithm

LcdString("A",35,0); //display an A for auto

}

}

else {

if (!mpptOn) {

pwmWrite(3, sensorValue / 4);

}

else {

mpptOn = false;

LcdString("M",35,0); //display an M for manual

}

}

lastPanelWatts = panelWatts;

batteryVolts = analogRead(A4) \* 0.0163; //get the battery volts

batteryVolts = (batteryVolts + lastBatteryVolts) / 2; //average the volts

lastBatteryVolts = batteryVolts;

LcdString(dtostrf(batteryVolts,5,2,&tmp[0]),0,2); //display the battery volts

batteryAmps = (analogRead(A5)-759)\*0.0337; //get the panelAmps

batteryAmps = (batteryAmps + lastBatteryAmps) / 2; //average the panelAmps

lastBatteryAmps = batteryAmps;

if (batteryAmps < 0.2) batteryAmps = 0; //don't let the batteryAmps go below zero

// LcdString(dtostrf(batteryAmps,5,2,&tmp[1]),0,4); //display the panelAmps

// if (volts > 1) {

// siemens = panelAmps / volts; //calculate the conductance

// LcdString(dtostrf(siemens,5,2,&tmp[2]),0,2); //display the siemens

// }

batteryWatts = batteryVolts \* batteryAmps; //calculate the battery watts

efficiency = batteryWatts / panelWatts \* 100;

LcdString(dtostrf(efficiency,3,0,&tmp[2]),0,1); //display the efficiency

maxwatts = max(maxwatts, panelWatts); //calculate the max watts

LcdString(dtostrf(maxwatts,4,1,&tmp[3]),40,3); //display the max watts

if (panelWatts > 27) highWatts = true; //go to 83Watt bargraph

LcdXY(0,5);

if (!highWatts) {

barwatts = round(panelWatts \* 3);

for(int k=0; k<84; k++)

{

if (k >= barwatts) LcdWrite (LCD\_D, 0x00);

else if (k % 30 == 0) LcdWrite (LCD\_D, 0xf0);

else if (k % 15 == 0) LcdWrite (LCD\_D, 0xe0);

else LcdWrite (LCD\_D, 0xc0);

}

}

else {

barwatts = round(panelWatts);

for(int k=0; k<84; k++)

{

if (k >= barwatts) LcdWrite (LCD\_D, 0x00);

else if (k % 10 == 0) LcdWrite (LCD\_D, 0xf0);

else if (k % 5 == 0) LcdWrite (LCD\_D, 0xe0);

else LcdWrite (LCD\_D, 0xc0);

}

}

Voc = max(Voc, panelVolts);

Isc = max(Isc, panelAmps);

chartX = min(panelVolts/Voc\*84, 83);

chartY = min(panelAmps/Isc\*48, 47);

if (chartX > 41 & chartY > 23) { //if in upper right quadrant

LcdPlot (chartX, chartY);

}

delay(10);

}

Что я изменил в скетче?!

1. У автора:

// Wiring:

//A0 - Pot

//A1 - Volts (solar)

//A2 - Volts (battery)

//A4 - Amps (solar)

//A5 - Amps (battery)

//D3 - PWM out to DCOI

//D8 -LCD\_RST (1)

//D9 - LCD\_CE (2)

//D10 - LCD\_DC (3)

//D11 - LCD\_DIN (4)

//D12 - LCD\_CLK (5)

//-----LCD\_VCC (6)

//---LCD\_LIGHT (7)

//-----LCD\_GND (8)

#define LCD\_C     LOW

#define LCD\_D     HIGH

#define PIN\_RESET 8

#define PIN\_SCE   9

#define PIN\_DC    10

#define PIN\_SDIN  11

#define PIN\_SCLK  12

У меня:

// Wiring:

//A3 - Pot

//A1 - Volts (solar)

//A4 - Volts (battery)

//A2 - Amps (solar)

//A5 - Amps (battery)

//D3 - PWM out to DCOI

//D12 -LCD\_RST (1)

//D11 - LCD\_CE (2)

//D10 - LCD\_DC (3)

//D9 - LCD\_DIN (4)

//D8 - LCD\_CLK (5)

//-----LCD\_VCC (6)

//---LCD\_LIGHT (7)

//-----LCD\_GND (8)

#define LCD\_C LOW

#define LCD\_D HIGH

#define PIN\_RESET 12

#define PIN\_SCE 11

#define PIN\_DC 10

#define PIN\_SDIN 9

#define PIN\_SCLK 8

2. у автра:

panelVolts = analogRead(A1) \* 47.0 / 1023; //get the volts

………………………

panelAmps = (514 - analogRead(A4)) \* 27.03 / 1023; //get the panelAmps

………………………

sensorValue = analogRead(A0);

………………………

batteryVolts = analogRead(A2) \* 25.0 / 1023; //get the battery volts

……………………

batteryAmps = (514 - analogRead(A5)) \* 27.03 / 1023; //get the panelAmps

У меня:

panelVolts = analogRead(A1) \* kv; //get the volts

……………..

panelAmps = (analogRead(A2)-516) \* 0.0394; //get the panelAmps

…………….

sensorValue = analogRead(A3);

…………….

batteryVolts = analogRead(A4) \* kv; //get the battery volts

…………….

batteryAmps = (analogRead(A5)-516)\*0.0394; //get the panelAmps

А теперь по-подробнее:

1. Здесь мне пришлось подправить скетч для более легкой разводки платы, чтоб дорожки как можно реже пересекались, и не приходилось делать перемычки и было минимум дорожек на обратной стороне. Это так-же касается строчки: sensorValue = analogRead(A3);

2. Далее распишу каждую измененную строку:

panelVolts = analogRead(A1) \* kv;

batteryVolts = analogRead(A4) \* kv;

Как известно, у ардуины АЦП в качестве аналогового входа. Напряжение 0..5 вольт ардуинка воспринимает значениями 0..1023. Теперь необходимо чтоб при подаче напряжения на контроллер, на вход ардуины подавалось не более 5 вольт. Иначе есть вариант спалить ардуину.

Для этого делаем делитель напряжения, у меня его роль выполняет подстроечный резистор на 68кОм. Теперь надо вычислить коэффициент kv, для этого вставляем в строку максимальное напряжение, которое хотим подключить к контроллеру.

К примеру:

Максимальное напряжение – 30 вольт, при этом на дуине должно поступить 5 вольт. Итак получаем: 30=1023\*kv kv = 0.0293

Строки принимают вид:

panelVolts = analogRead(A1) \* 0,0293;

batteryVolts = analogRead(A4) \* 0,0293;

Далее переходим к строкам :

panelAmps = (analogRead(A2)-516) \* ka;

batteryAmps = (analogRead(A5)-516)\*ka;

Тут немного сложнее с вычислениями. Сначала необходимо вычислить нулевую точку датчика. Дело в том, что этот датчик меряет ток от -20 .. 20 ампер. Т.е. когда ток=0 датчик должен выдавать 2.5 вольт и на дуине должно быть значение 512. Вот только оно не всегда так бывает. У меня нулевая точка была на значении 516. Нулевую точку необходимо вычислять для обоих датчиков отдельно.

После определения нулевой точки, необходимо вычислить коэффициент ka. Для этого подставляем в строчку значения: Максимальный ток датчика 20А при максимальном значении на входе 1023 и получаем:

20=(1023-516)\*ka ka=20/507=0,0394

Строки принимают вид:

panelAmps = (analogRead(A2)-516) \* 0,0394;

batteryAmps = (analogRead(A5)-516)\*0,0934;

Но это только мои данные, для каждого контроллера это все вычисляется индивидуально!!!

Далее калибровка:

Собираем плату, подстроечные резисторы выкручиваем в крайне правое положение, чтоб на входе дуины было напряжение = 0(дабы не спалить дуину). Параллельно подключаем вольтметр и плавно крутя ручку подстроечника выставляем на экране дуины такое-же напряжение как на вольтметре. То же самое проделываем на другой стороне. Калибровка датчиков напряжения закончена.

Токовые датчики Мы откалибровали, но вполне возможно потребуется переколибровать под нагрузкой. Т.е. подать нагрузку, подключить амперметр и сравнить значения и подправить коэффициент ka.

P.S. Мои 5 копеек в конструкции..

Во время сборки контроллера я избавился от стороннего питания ардуины, установив стабилизатор напряжения 7805 на 5 воль. Теперь контроллер включается при подключении к аккумулятору. Думаю и от 9-вольтовой кроны так-же можно избавиться!

Так же исходя из скетча, думаю что можно «выкинуть» датчик тока на аккуме, потому как его значения считывается, но не нашел, где оно используется. Полагаю что нигде, даже на экран не выводит!

В скетче, представленном выше, я выделил красным строчки, которые необходимо настраивать!

Ну, как-то так! УДАЧИ!