Recherche Hardware und Software

AUER-JAMMERBUND, BINDER, CSURMANN, ENGELHART, FELDHOFER, FRITZEL, GRAF

Dies ist das Projekttagebuch der Hardware- und Software-Gruppe

Inhaltsverzeichnis

Arbeitsauftrag

Hardware-Gruppe:

Mithilfe der Hardware-Liste sollen alle benötigten Bauteile bestellt und anschließend zusammengebaut werden. Hierzu wird die Liste von der Vorgängergruppe übernommen. Es sollen, wenn möglich, alle Bauteile auf at.rs gefunden und bestellt werden.

Software-Gruppe:

Existierende Software einlesen und fertigstellen. Es wurde uns eine Software von, der Vorgänger Gruppe weitergegeben. Diese muss zuerst verstanden und dann erweitert werden.

Abzugebendes

Am Ende des Zyklus soll eine übersichtliche Dokumentation der erbrachten Leistungen vorliegen.

Projekttagebuch

07/11/2019

Ausgabe des Arbeitsauftrags und Einarbeiten in die Unterlagen, welche wir von der Vorgänger-Gruppe übernommen haben.
Es wurden zwei Gruppen gebildet. Die Hardware- und Software-Gruppe. Die Hardware Gruppe besteht ausfolgenden Mitschülern:

- Binder Juliana
- Engelhart Sophie
- Graf Linus

Die Software Gruppe besteht aus:

- Auer-Jammerbund Markus
- Csurmann Mathias
- Manuel Fritzel
- Feldhofer Hans-Peter

Hardware-Gruppe: Suche nach passenden Bauteilen auf RS.at
Software-Gruppe: Einlesen der erhaltenen Software.

14/11/2019

Hardware-Gruppe: Die Suche nach den passenden Bauteilen wurde fortgeführt
Software-Gruppe: Verwerfen der erhaltenen Software nach Empfehlung der vorherigen Laborgruppe. Einlesen in die Fachtheorie und Neubeginn des Programmes.

21/11/2019

Hardware-Gruppe: Ein paar bereits gefundene Bauteile wurden ausgetauscht, da wir doch etwas anderes zur Realisierung gewählt haben.
Software-Gruppe: Recherche Stromsensor sowie die Einbindung der Kalibrierung im Programm. (Siehe: Beilage)

28/11/2019

Hardware-Gruppe: Die Bauteile wurden in einer übersichtlichen Liste zusammengeführt, um jene dem Abteilungsvorstand vorzulegen.
Software-Gruppe: Aufstellung des Grundkonzeptes. Beginn der Programmierung ohne Klassen.

05/11/2019

Hardware-Gruppe: Die Liste wurde genehmigt und die Bauteile bestellt. Lediglich ein Bauteil, welches auf Amazon bestellt wird, wurde nicht bestellt, da dieses von den Schülern bestellt werden wird.
Software-Gruppe: Weiterführung des Programmes leider konnten wir dieses Vorhaben nicht zur Gänze erfüllen. Der Programmcode wurde auf unseren eigens für dieses Projekt angelegten GIT-Hub Repository hochgeladen. Der Link für diesen Git-Hub Repository ist wie folgt:
https://github.com/CsurMathias/LA1V_HK

12/11/2019

Fertigstellen der Dokumentation

Fertige Stückliste von Hardware-Gruppe:

Website	Bestellnummer	Hersteller. T-Nr.	Produktbeschreibung	Anzahl	Preis
at.rs-online.com	**188-8310**	Pi4 4GB Bulk	Raspberry Pi	1	49,159€
at.rs-online.com	**379-2487**	MCP3208-BI/P	Analog Digital Umwandler	1	4,02€
conrad.at	**1616236 - 62**	Iduino ME067	Stromsensor	3	je 6,24€
at.rs-online.com	**765-2939**	CH143DU	Sicherungsschalter	1	44,0€
at.rs-online.com	**187-3413**	RPI4 PSU EU WHITE	Raspberry Netzteil	1	7,69€
at.rs-online.com	**916-0340**	EUC3-40-4P	Schütz	1	74,76€
conrad.at	**1695384 - 62**	COM-KY019RM	Relais-Modul	3	je 3,33€

Preis gesamt: **208,269€**

Programmcode von Software-Gruppe:

Hier folgt der Programmcode sowie die Kommentare die den Code erläutern:

```
1   #include <wiringPi.h>
2   #include <string>
3   #include <bitset>
4   #include <iostream>
5
6   using namespace std
7
8   #define ADU_PIN0  6
9   #define ADU_PIN1 10
10  #define ADU_PIN2 11
11  #define ADU_PIN3 31
12
13  #define RELAIS_SOURCE 2
14  #define RELAIS_STAR 3
15  #define RELAIS_TRIANGLE 4
16
17  const int analogInCurrent = 8;
18  const int analogInVoltage = 9;
19
20
21  int mV_A = 100;             //mV/A (datasheet)
22  int rawValue = 0;
23  int acOffset = 2500;        //datasheet
24
25
26  const int maxAmount = 20;
27  int rawVoltage[maxAmount];
28
29
```

Pin-Belegung, definieren Variablen, Magic Numbers wurden aus Datenblatt entnommen

```
30  int main(void)
31  {
32          wiringPiSetupSys();
33
34          pinMode(ADU_PIN0, INPUT);
35          pinMode(ADU_PIN1, INPUT);
36          pinMode(ADU_PIN2, INPUT);
37          pinMode(ADU_PIN3, INPUT);
38
39          pinMode(analogInCurrent, INPUT);
40          pinMode(analogInVoltage, INPUT);
41
42
43
44          return 0;
45  }
46
47  double findPeakV()
48  {
49          int arrayMax = 0;
50
51          //read ADC
52          for (int counter = 0; counter < maxAmount; counter++)
53          {
54                  rawVoltage[counter] = readADC();
55          }
56
```

findPeakV Funktion um den Scheitelwert der Spannung zu bestimmen.

```
57          //find max U val
58          for (int counter = 0; counter < maxAmount; counter++)
59          {
60                  if (rawVoltage[counter] > arrayMax)
61                  {
62                          arrayMax = rawVoltage[counter];
63                  }
64          }
65  }
66
67  //calculate Current value from raw input
68  double calcCurrent(int rawInput)
69  {
70          rawValue = rawInput;
71          double voltage = (rawValue / 1024.0) * 5000;
72          return ((voltage - acOffset) / mV_A);
73  }
74
75  //reads ADC-Value converts it to ulong
76  int readADC()
77  {
78          int bit0_int = digitalRead(ADU_PIN0);
79          int bit1_int = digitalRead(ADU_PIN1);
80          int bit2_int = digitalRead(ADU_PIN2);
81          int bit3_int = digitalRead(ADU_PIN3);
82
83          std::string bit0 = std::to_string(bit0_int);
84          std::string bit1 = std::to_string(bit1_int);
85          std::string bit2 = std::to_string(bit2_int);
86          std::string bit3 = std::to_string(bit3_int);
```

calcCurrent Funktion um Strom auszulesen und in ein brauchbares Format umrechnen
readADC brauchen wir da Raspberry Pi keinen Analogen Eingangspin besitzt.

```
87
88          std::string nibble = bit3 + bit2 + bit1 + bit0;
89
90          int decVal = std::bitset<8>(nibble).to_ulong();
91
92          return decVal;
93  }
94
95
96  void meassure(int mode)
97  {
98          digitalWrite(RELAIS_SOURCE, HIGH);
99
100         //TDODO: Software lock
101         switch (mode)
102         {
103                 case 1:
104                         //star-triangle
105                         //TODO: meassure current to go to triangle
106                         digitalWrite(RELAIS_STAR, HIGH);
107                         break;
108
109                 case 2:
110                         //star
111                         digitalWrite(RELAIS_TRIANGLE, LOW);
112                         digitalWrite(RELAIS_STAR, HIGH);
113                         break;
114                 case 3:
115                         //triangle
116                         digitalWrite(RELAIS_STAR, LOW);
117                         digitalWrite(RELAIS_TRIANGLE, HIGH);
```

Stern-Dreieck Anlauf.

```
118                 case 100:
119                         //error
120                         break;
121
122                 default:
123                         break;
124         }
125 }
126
127 //get userinput
128 int userChoice()
129 {
130         int input;
131         int ack;
132
133         std::cout << "Select your desired starting manner. 1 = Stern-Dreieck / 2 = Sternbetrieb / 3 = Dreieckbetrieb";
134         std::cin >> input;
135
136         std::cout << "You have chosen " << input << " correct = 1 / exit = 0";
137         std::cin >> ack;
138
139         if (ack == 1)
140         {
141                 return input;
142         }
143         else
144         {
145                 return 100;
146         }
147 }
```

User Input für Navigation im Programm.