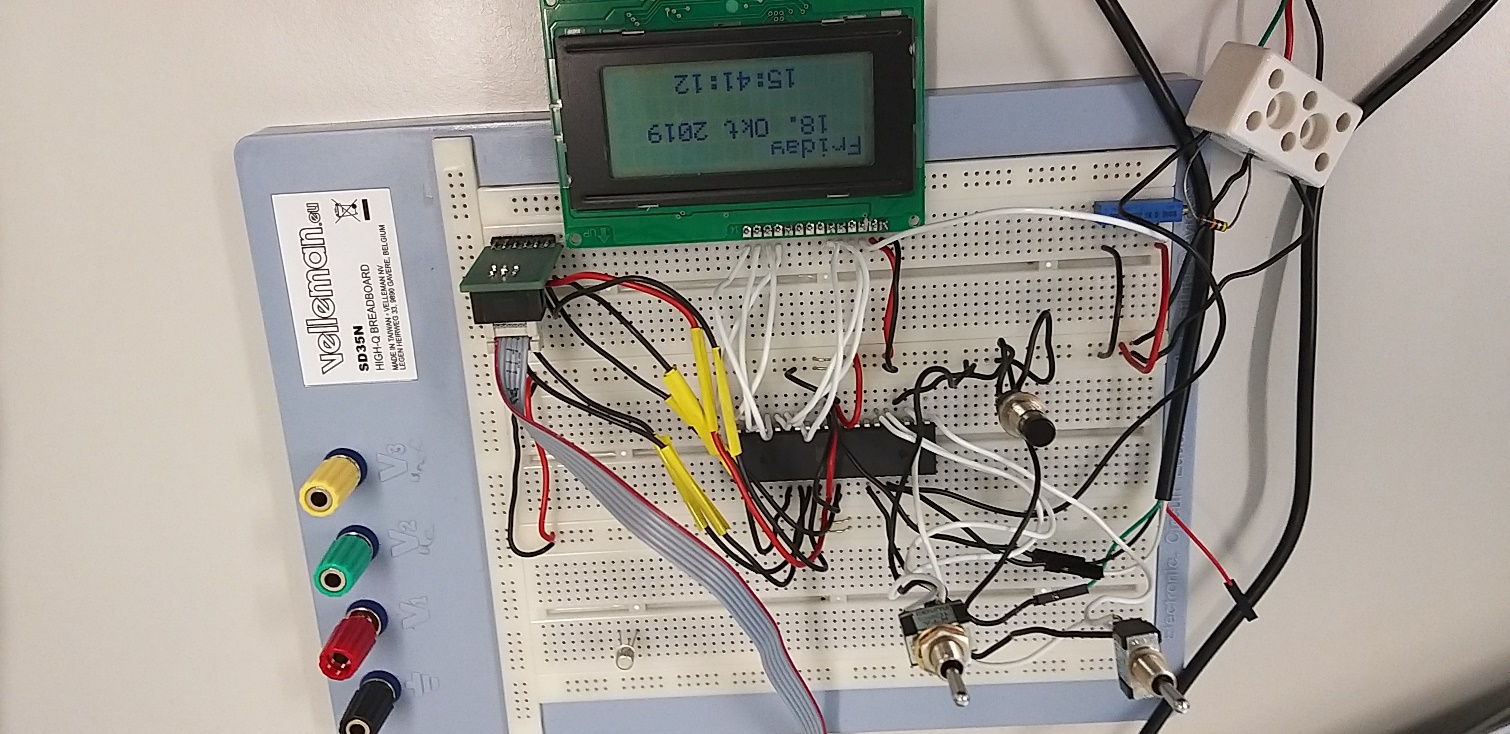
**Projekt: Wecker**



**Autor: Alexandra Spring**

**Datum: 18.10.2019**

**Ausgangslage:**

Vom Lehrbetrieb wurde mir vorgegeben, dass ich spätestens am 07.45 Uhr im Betrieb erscheinen muss. Da ich aber keinen geeigneten Wecker habe, ist das ein bisschen schwierig. Da ich aber eine auszubildende Elektronikerin bin, dachte ich mir sollte ich in der Lage sein, einen Wecker selber zu programmieren. Ich bin dann in die WDU und habe gefragt ob ich den als nächstes Projekt programmieren darf. Das Projekt wurde bestätigt und ich darf mich nun dem Wecker widmen.

**Anforderungen des Weckers:**

**Meine persönlichen Anforderungen:**

1. Zeit soll mit einem DCF77 Signal betrieben werden.
2. Falls es zu Unterbrüchen im Signal kommt soll di Zeitanzeige selber weiter zählen können.
3. Die Weckzeit soll beliebig eingegeben werden können.
4. Der Wecker soll temporär ausgeschaltet werden können.
5. Der Wecker soll alle 2 Minuten wieder ertönen.
6. Bei jedem erneuten Wecken soll der Ton aufdringlicher werden.
7. Es soll 4 Tonfolgen haben.
8. Während dem Alarm soll eine Uhr auf dem Display in einer oberen Ecke angezeigt werden.
9. Falls der Wecker aktiviert ist, muss es eine kleine Uhr auf dem Display haben.
10. Es soll ein Countdown auf dem Display angezeigt werden bis zum nächsten Alarm
11. Es soll ein Menü oder eine Rechnung geben die der Schlafende eingeben oder Lösen muss, dass der Wecker definitiv abstellt.
12. Das ganze soll mit einem Atmega 324P angesteuert werden.
13. Die Hintergrundbeleuchtung des Displays soll während dem Alarm aufleuchten und mittels Taste abstellbar sein.

**Anforderungen von Roman:**

1. Watchdog Timer
2. Alarm in EEPROM einschreiben und beim Falle einer Änderung den Alarm abspeichern und bei Neustart entsprechend setzen.

**Arbeitsschritte:**

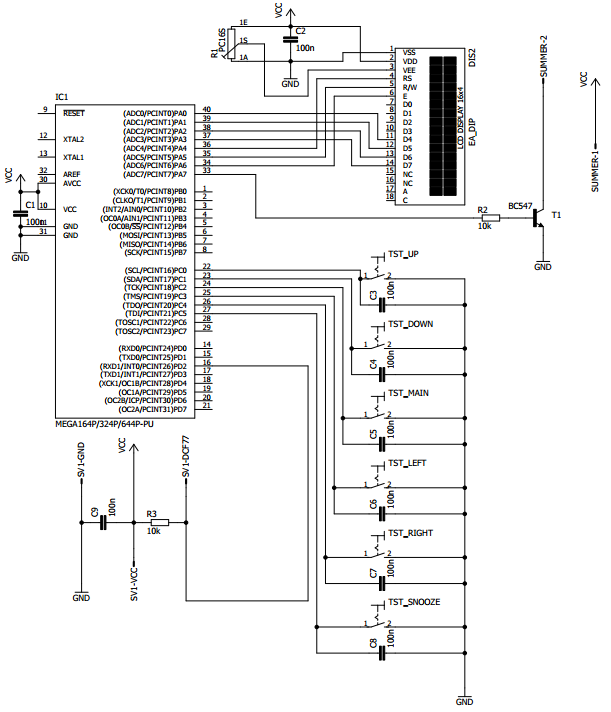
* Schema zeichnen
* Atmega einstellungen
* Materialien suchen
* Prototyp auf Steckbrett aufbauen
* Programm modifizieren
* Programm erweitern
* Gehäuse layout
* Gehäuse fertigne

**Selbstreflexion:**

In dieser Woche habe ich gelernt wie man ein Menu im C programmiert. In meinem Fall habe ich das mit einem Switch case gemacht, aber habe mir noch eine andere Methodik erklären lassen. Ich hätte früher nach einem Anhaltspunkt fragen können. Ich weiss jetzt wie man mit einem Compiler die Fusebits eines Mikrocontrollers lesen und neu setzen kann. Vorher habe ich das mit dem AVR Burn O Mat gemacht aber das Programm konnte den Atmega 324 nicht lesen weil der im File nicht als solcher angegeben war. Deshalb hat Roman mir gezeigt wie das anders gemacht werden kann und wo man das im Datenblatt auslesen kann und nicht auf engbedded.com. Ich habe gelernt wie man mit Pointern arbeiten kann und wie die funktionieren. Ich hätte von anfang an beim Subprogramm fragen sollen und bei diesem Punkt genau mein programm durchlesen und meine Flüchtigkeitsfehler mit der Umstrukturierung selber bemerken können.

Ich habe Probleme mit den Register Descriptions im Datenblatt. Ich brauche sehr lange um diese zu lesen und zu setzen und am Schluss sind sie nicht komplett richtig und ich bin mir selber unsicher. Ich muss unbedingt von anfang an fragen was ich nicht verstanden habe an der Aufgabenstellung und mir ein bisschen merh zutrauen. Ich sollte auch ein bisschen mehr ausprobieren und nicht nur überlegen und Fragen und eine Lösung haben. Ich sollte auch Raum für Fehlschläge lassen und nicht nur für Erfolge, welche ich schlussendlich vielleicht nicht einmal erreiche.

**Schema:**



**Nächste Schritte:**