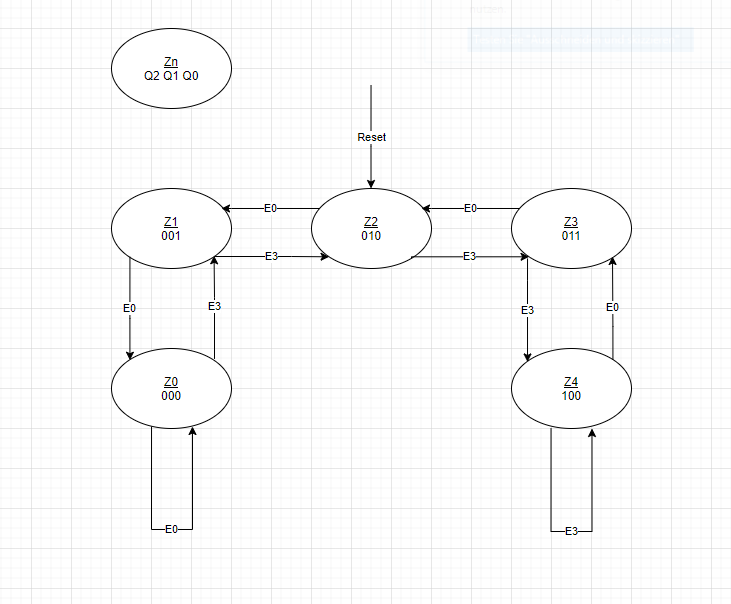
# Projekt 2:

## 2.1.1:

Y = E0 V E3

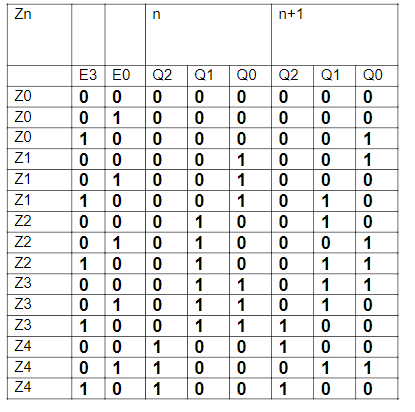
## 2.1.2:



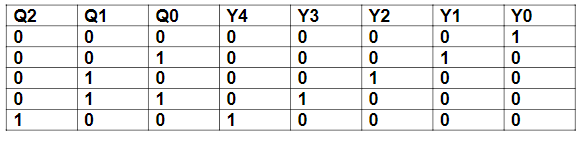
## 2.1.3:

Das Schaltwerk SW1 muss mindestens 3 Speicherelemente haben, da dadurch insgesamt 9 Zustände dargestellt werden können. Da wir lediglich 5 Zustände haben passt das. Bei 2 Speicherelementen könnten wir nur 4 Zustände darstellen, was ein Zustand zu wenig wäre.

## 2.1.4:



## 2.1.5:



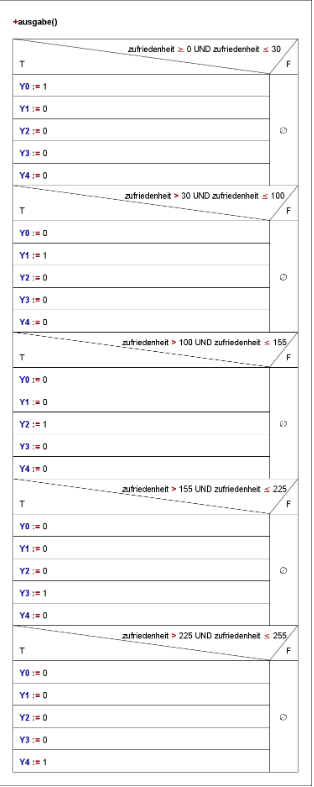
## 2.2.3:

Es wird nacheinander geschaut, ob einer der Taster gedrückt wurde. Wurde einer gedrückt (z.B. S3), so wird die Funktion für den gedrückten Schalter ausgeführt (hier sehr\_zufrieden()), danach die Funktion Ausgabe() für die Aktualisierung der Lampen und dann noch die warten\_1s() Funktion für eine 1 Sekunden Verzögerung. Am Ende wird gewartet solange der entsprechende Schalter noch gedrückt ist. Somit wird vermieden, dass durch dauerhaftes Drücken des Schalters die entsprechende Methode (hier sehr\_zufrieden())mehrmals ausgeführt wird. Auch ein schnelles mehrmaliges Drücken hintereinander wird vermieden, da eine

Sekunde gewartet wird und dadurch erst eine Sekunde später das Drücken des Tasters wieder eine Auswirkung hat.

//NOCH NICHT KONTROLLIERT AB HIER

## 2.2.5:



## 2.2.6:

Um denTimer im Interrupt-Betrieb benutzen zu können müssen erst der Timer und der Interrupt in der init() Methode initialisiert werden. Die Werte für den Timer sind:

TMOD = 0x10h (Timer 1 im 16-Bit Mode)

t0 = (2^16 − xd) ∗ 12f

xd = 2^16 − t0 ∗ 12f

xd = 2^16 – 50 ∗ 10^-3 ∗ 12 ∗ 10^6 \* 12

xd = 15536

𝑑=3𝐶𝐵𝑂ℎ (erzeugt 50ms Verzögerung)

➔TH1 =0x3C

➔TL1 = 0xB0

anzahl = 0 (am Anfang, am Ende sind es 20 (20 \* 50 ms = 1000ms)

Für den Interrupt werden in der init() Methodenoch EA für die globale Freigabe und ET1 für die Freigabe des Timer1-Interrupts auf 1 gesetzt.In derwarten\_1s() Methode wird nun zuerstTR1 = 1 gesetzt, um den Timer zu starten. Dann wird mit einer while-Schleife gewartet solange anzahl != 20 wahr ist.Währenddessen läuft der Timer.Läuft der Timer über, wird die ISR ausgeführt, in der das Timer-Flag gelöscht wirdund die Variable anzahl um 1 erhöht wird, da ein Durchgang vorüber ist.Der Timer wird wieder mit den Anfangswerten geladen und falls anzahl == 20 wahr ist wird TR1 auf 0 gesetzt, um den Timer zu stoppen,andernfalls wird TR1 auf 1 gesetzt, um ihn nochmals zu starten. Er läuftebensolange bis die Variable anzahl auf 20ist. Dann wird der Timer gestoppt, die ISR beendet und in der warten\_1s() Methode wird die while-Schleife beendet, da anzahl nun gleich 20ist. Die Variable anzahl wird wieder auf 0 gesetzt für einen möglichen erneuten Aufruf der Verzögerung und das Unterprogramm wird beendet.