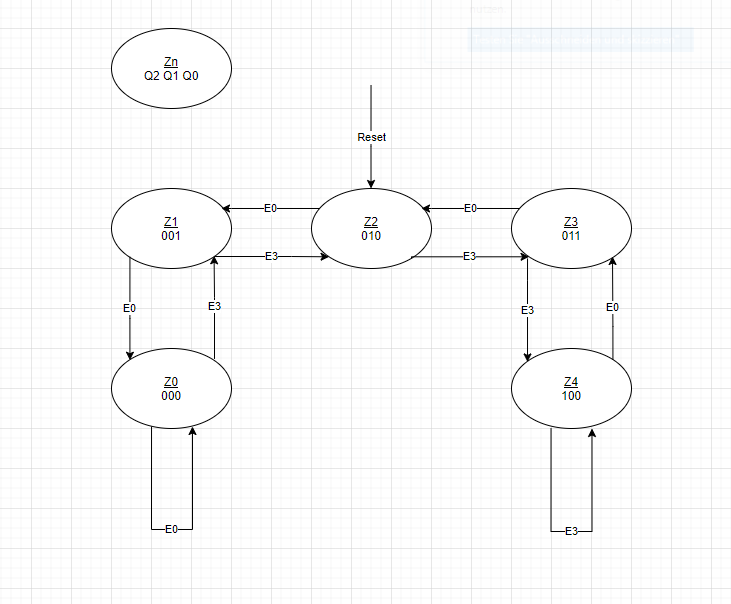
# Projekt 2:

## 2.1.1:

Y = E0 V E3

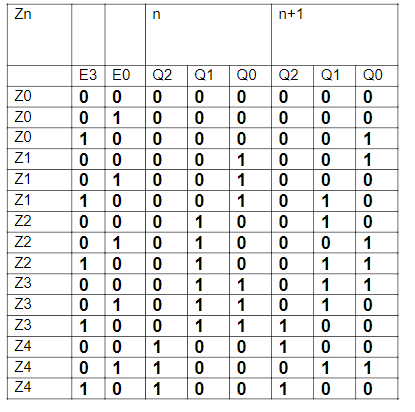
## 2.1.2:



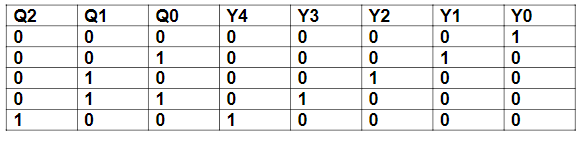
## 2.1.3:

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:

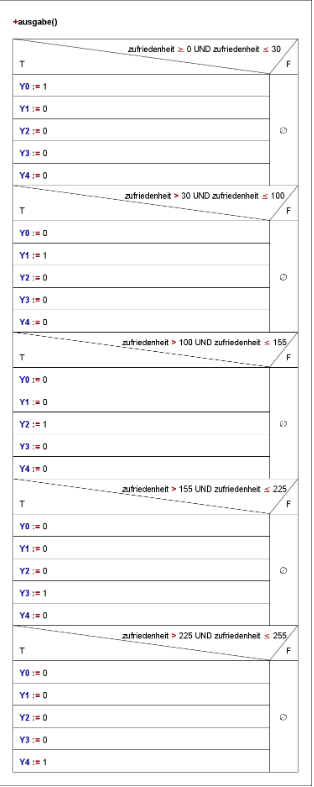


## 2.2.3:

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.

## 2.2.5:



## 2.2.6:

Um den Timer im Interrupt-Betrieb nutzen zu können, müssen als erstes jeweils der Timer und der Interrupt in der **init()** Methode initialisiert werden. Die Werte für den Timer sind folgende:

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

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

* TH1 = 0x3Ch
* TL1 = 0xB0h

Für den Interrupt werden in der init() Methode noch EA für die globale Freigabe und ET1 für die Freigabe des Timer1-Interrupts auf 1 gesetzt.

In der warten\_1() Methode wird nun zuerst TR1 = 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 wird und die Variable Anzahl um 1 erhöht wird, da ein Durchgang vorüber ist.

Der Timer wird wieder mit den Anfangswerten geladen und TR1 auf 1 gesetzt, um ihn nochmals zu starten.

Er läuft so lange bis die Variable Anzahl auf 20 ist. In der warten\_1() Methode wird die while-Schleife beendet, da Anzahl nun gleich 20 ist. Anschließend wird TR1 auf 0 gesetzt, um den Timer zu stoppen.

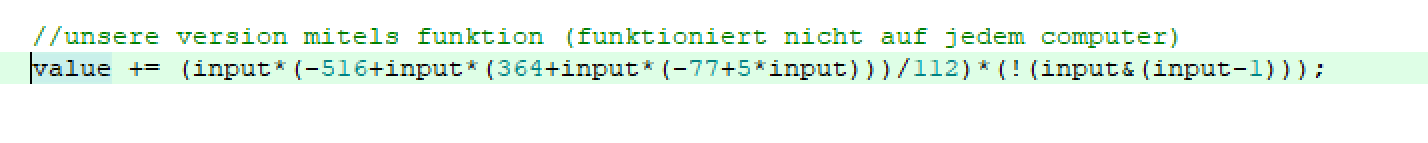
Die Variable Anzahl wird wieder auf 0 gesetzt für einen möglichen erneuten Aufruf der Verzögerung und das Unterprogramm wird beendet.

!!!Bild aus Keil zum Testen des Delays auf nächster Seite!!!

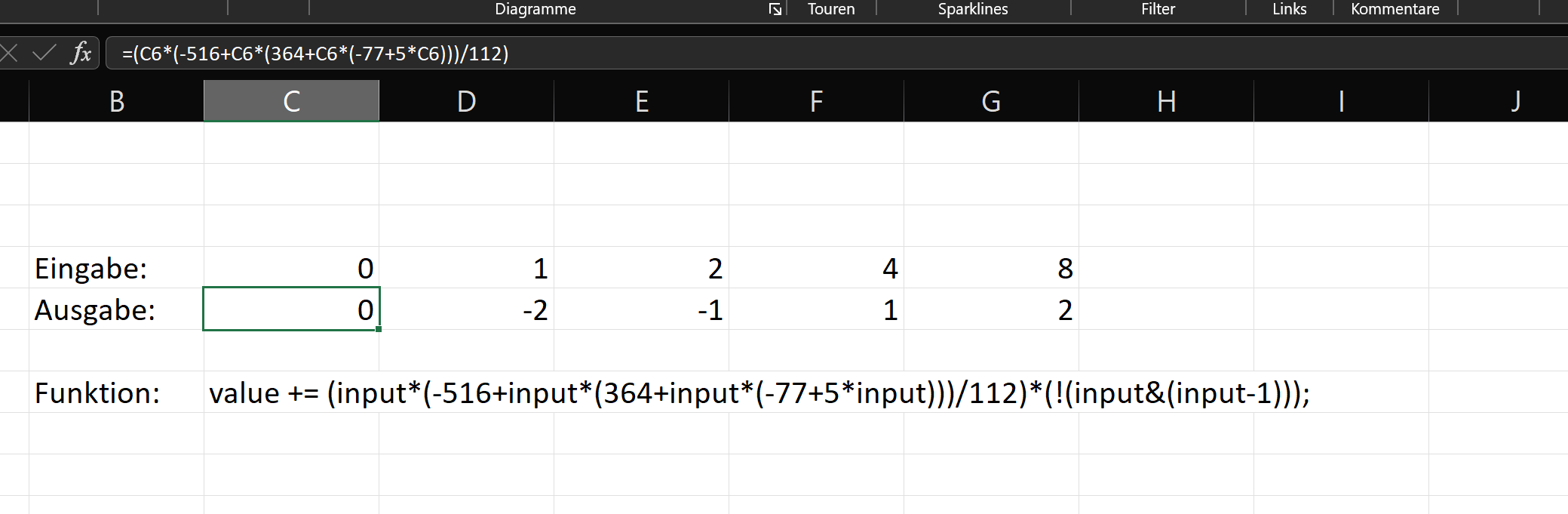
Ein Bild, das Text enthält.

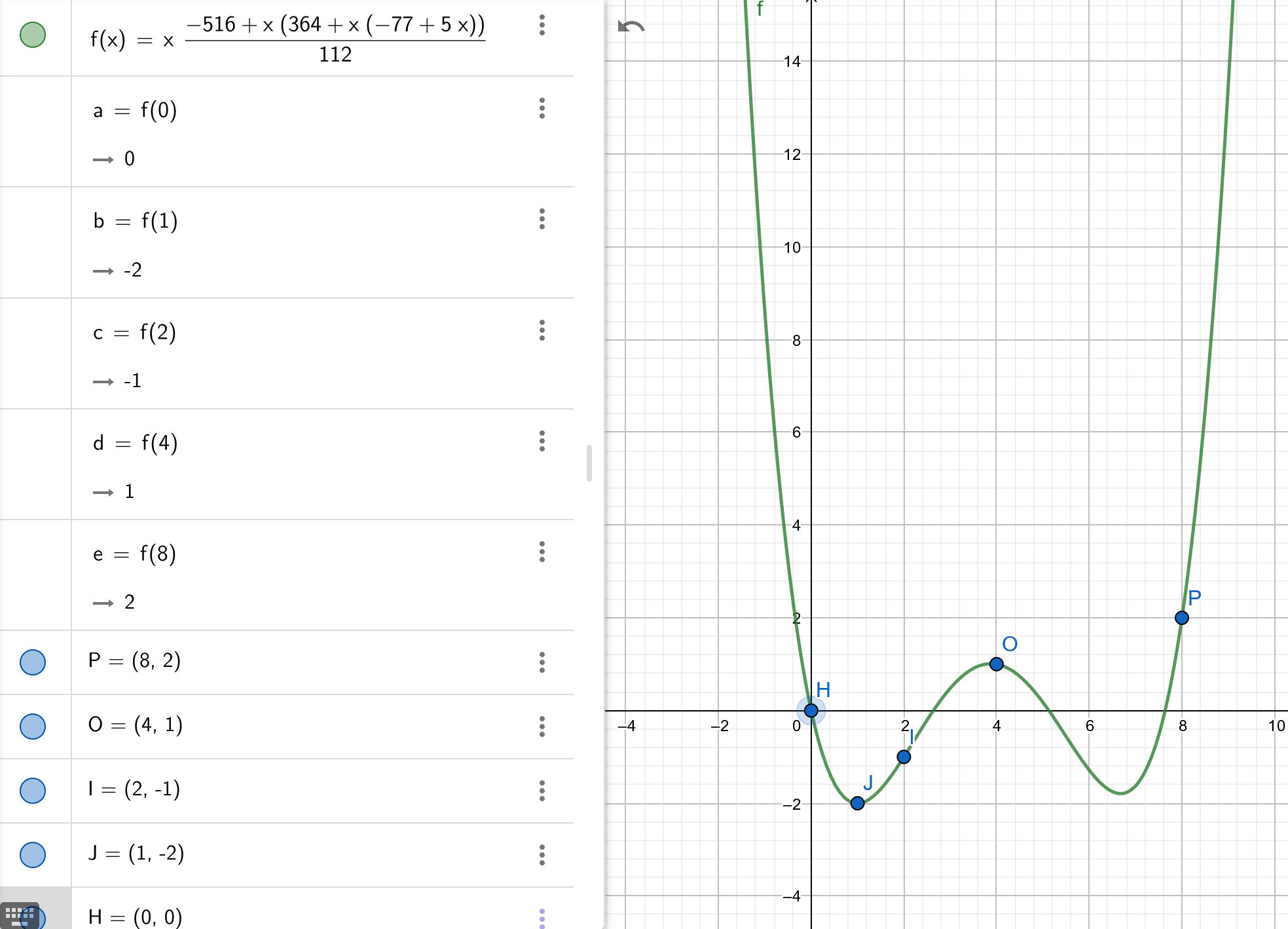
Automatisch generierte Beschreibung

Herleitung Zur Funktion welche in der Experimentellen Lösung benutzt wurde:



Funktioniert nicht auf allen Computern. Vermutlich rundungsfehler.





Wie im Bild erkennbar ist bei einer Eingabe von 1,2,4,8 (jeweils ein bit gesetzt) eine Änderung entsprechend der Aufgabenstellung von -2,-1,1,2 zu erkennen.

Diese Funktion ersetzt somit folgenden Switch-case:

