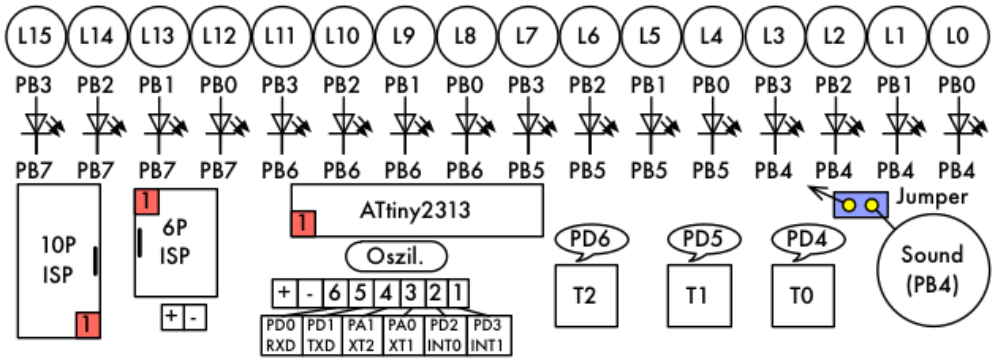
Einleitung

Ein Reaktionsspiel zum Beschäftigen von nervigen Kindern wurde entwickelt. Der Entwickler hat keine Doku erstellt, lediglich ein spärlich dokumentierter Quellcode (siehe Anlage) ist erhalten. Das Spiel wurde mit einem Zylonenauge **@1MHz** realisiert. Ihre Aufgabe ist, den Quellcode zu analysieren und schließlich eine Anleitung zu erstellen.



1. Töne ausgeben — 10 Punkte

Unterprogramm (UP) ***piepen(n:byte)***. Die Periodendauern der Töne sind im wortweise organisierten Programmspeicher abgelegt. Der Datentyp ist *unsigned int*, deshalb werden **2** Byte pro Feldelement verbraucht.

a) **ton** hat die Adresse 0x0100. Füllen Sie die Tabelle mit den Speicherinhalten für die ersten beiden Töne aus. 6P

Adresse	Dezimalwert	Hex-Wert	Frequenz
0x0100			
0x0101			

b) Welche Frequenz gibt piepen(12) aus (Berechnung, Begründung)? 4P

---

2. ISR(TIMER0_COMPA_vect) — 20 Punkte

a) Erstellen Sie ein Struktogramm für die ISR 10P

b) Warum sind l*euchtPunkt* und *piep* ausserhalb der ISR als *volatile* definiert? 2P

c) Wie lang ist der Abstand der ISR-Aufrufe bei Level0 und Level10 in ms? 4P

d) Wieso wird ***piepen(..)*** nicht in der ISR sondern im Hauptprogramm aufgerufen, welche Funktion hat dabei die Variable *piep*? 4P

---

3. Cheat-Modus — 4 Punkte

Zum Testen des Programms wurde ein Cheat-Modus eingebaut.

a) Wie aktiviert man den Cheat-Modus?. 2P

b) Wie kann das Spiel dann betrügerisch gespielt werden? 2P

---

4. Entprellen notwendig? — 6 Punkte

Die Tasten prellen mit ca. 10 ms. In **checkButton()** wurde keine Entprellung vorgesehen. Erörtern -Sie ob eine Entprellung notwendig wäre.

---

5. Spielanleitung erstellen — 5 Punkte

Erstellen Sie eine kurze nette Spielanleitung für das Spiel.

Total Punkte 45

```c
// *** Zylon-Reaktions-Spiel V1.0 (c) Oliver Mezger 9.2.2020 ***
#include <avr/io.h>         // Definitionen laden
#include <util/delay.h>     // Delay-Bibliothek laden
#include <avr/pgmspace.h>   // Flashzugriffe laden
// C-Dur Frequenzen als Periodendauer F_CPU ist Systemtakt
const unsigned int PROGMEM ton[]={F_CPU/264,F_CPU/297,F_CPU/330,F_CPU/352,
                                  F_CPU/396,F_CPU/440,F_CPU/495};
unsigned char ledausgabe[]={0xe1,0xe2,0xe4,0xe8,0xd1,0xd2,
0xd4,0xd8,0xb1,0xb2,0xb4,0xb8,0x71,0x72,0x74,0x78};
void piepen(unsigned char n){       // Ausgabe eines kurzen Tons
  OCR1A = pgm_read_word(&ton[n%7])>>(n/7);// pro Oktave halbe Periodendauer
  OCR1B = OCR1A/2;
  TCCR1A = 0b00100011; // Ausgang OC1B bei 0 setzen und bei OCR1B loeschen
  _delay_ms(70);        //Tondauer
  TCCR1A = 0b00000011; // Sound wieder aus
}
unsigned char buttonOld=0,buttonEnter=0;
void checkButton(){
  unsigned char bu=~PIND>>4; // Einlesen
  buttonEnter = ~buttonOld & bu;
  buttonOld=bu;
}
volatile unsigned char leuchtPunkt=0,piep=0;
ISR(TIMER0_COMPA_vect){       //bewegt den Leuchtpunkt
  static unsigned char up=1;
  if (up){
    leuchtPunkt = leuchtPunkt + 1;
    if (leuchtPunkt >= 8){ //wenn oben angekommen
      up=0;
      piep=1;               // einen tiefen Piep ausgeben
    }
  }
  else{
    leuchtPunkt = leuchtPunkt - 1;
    if (leuchtPunkt == 0){  //wenn unten angekommen
      up =1;
      piep=2;               // einen hohen Piep ausgeben
    }
  }
  PORTB = ledausgabe[leuchtPunkt];     //ausgeben Leuchtpunkt
}
int main(){
  unsigned char level=0,cheat=0;
  DDRB  = 0xff;  // PB0..7 als Ausgang
  PORTD = 0x7f;  // Pullups an
  TCCR1A= 0b00100011;  // Ausgang OC1B bei 0 setzen und bei OCR1B loeschen
  TCCR1B= 0b00011001;  // Waveform Generation Mode: Fast PWM it OCR1A als Top, Timer mit CPU-CLK
  TCCR0B= 5;           // Phi / 1024
  TCCR0A= 1<<WGM01;    // CTC Mode mit OCR0A als TOP
  TIMSK = 1<<OCIE0A;   // Interrupt frei geben
  OCR0A = 199-level;   // bei (199-level)*1024µs Interrupt
  if ((PIND & 1<<PD4) == 0) cheat=1;
  sei();
  while (1){               // Endlosschleife
    checkButton();         // Tastaturabfrage
    if (buttonEnter == 2){ // wurde Taste T1 gedrueckt?
      if (leuchtPunkt == 4){ //LED4
        piepen(12);piepen(24); // Ton fuer getroffen
        level++;           // naechster Level
      }
      else{
        piepen(14);piepen(8);piepen(0);// Ton fuer daneben
        level=0;         // zurueck auf unteren Level
      }
      OCR0A=199-(level*10);  // Geschwindigkeit anpassen
    }
    if (cheat && buttonEnter == 4){ // Taste T2
      piepen(12);piepen(24);
      level++;
      OCR0A=199-(level*10);
    }
    if (piep){  // wenn Eckpunkt erreicht Ton ausgeben
      piepen(level+(piep-1)*8);
      piep=0;
    }
  }
}
```