LED Echtzeituhr mit dem 4 Bit Digital Tube LED Display (Teil 1)

Hallo und willkommen zu einem neuen spannenden Blog in der Welt der Microelektronik. Im heutigen ersten Teil einer mehrreihigen Blogreihe befassen wir uns mit einem Re-Work eines bereits erschienen Blogartikels mit dem Aufbau einer Echtzeituhr. Die Uhrzeit ist jederzeit bequem mit zwei Tastern einstellbar, bringt einen animierten Sekundenanzeiger (blinkenden Doppelpunkt) mit und ist darüber hinaus auch sehr preiswert aufzubauen, da auf eine externe RTC Uhr verzichtet werden kann.

Der Nachteil besteht in diesem Teil des Blogs noch jedoch daraus, dass bei einem Stromausfall die Uhr wieder neu gestellt werden muss. Dies werden wir noch in den kommenden Teilen mit Erweiterungen unserer Uhr ändern. Für viele Anwendungen steht jedoch mit dem Code bereits ein solides Grundgerüst bereit. Darüber hinaus ist die Teileliste sehr übersichtlich:

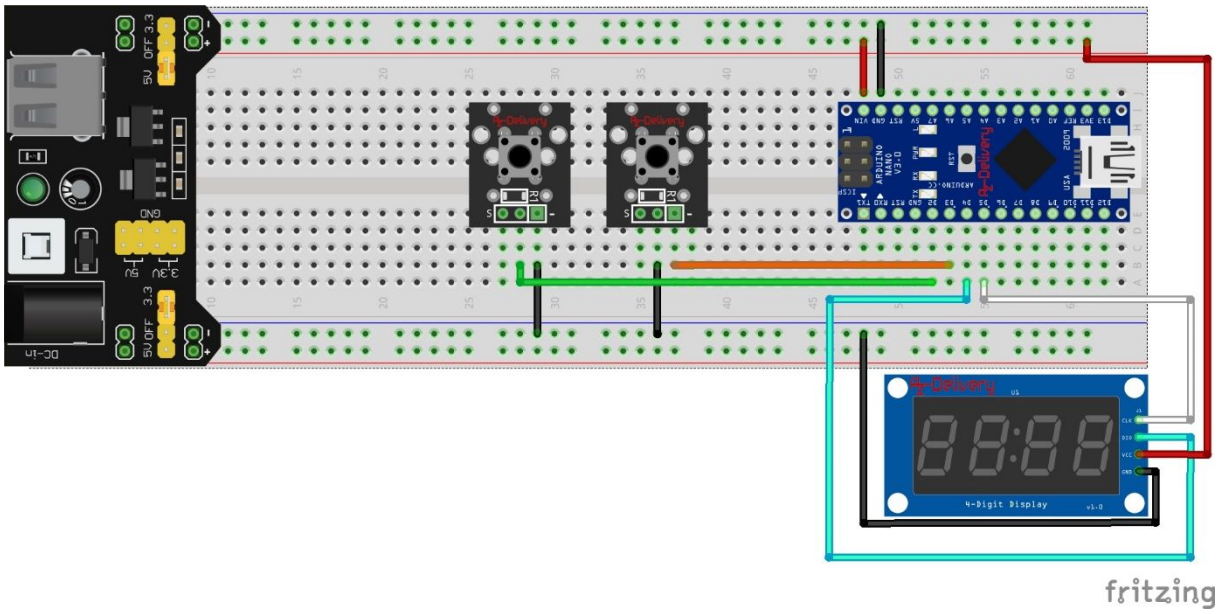
1x 4 Digit 7 Segment Display (TM1637)

2x KY-004 Button Module

1x Nano V3

1x YwRobot Breadboard Power Supply

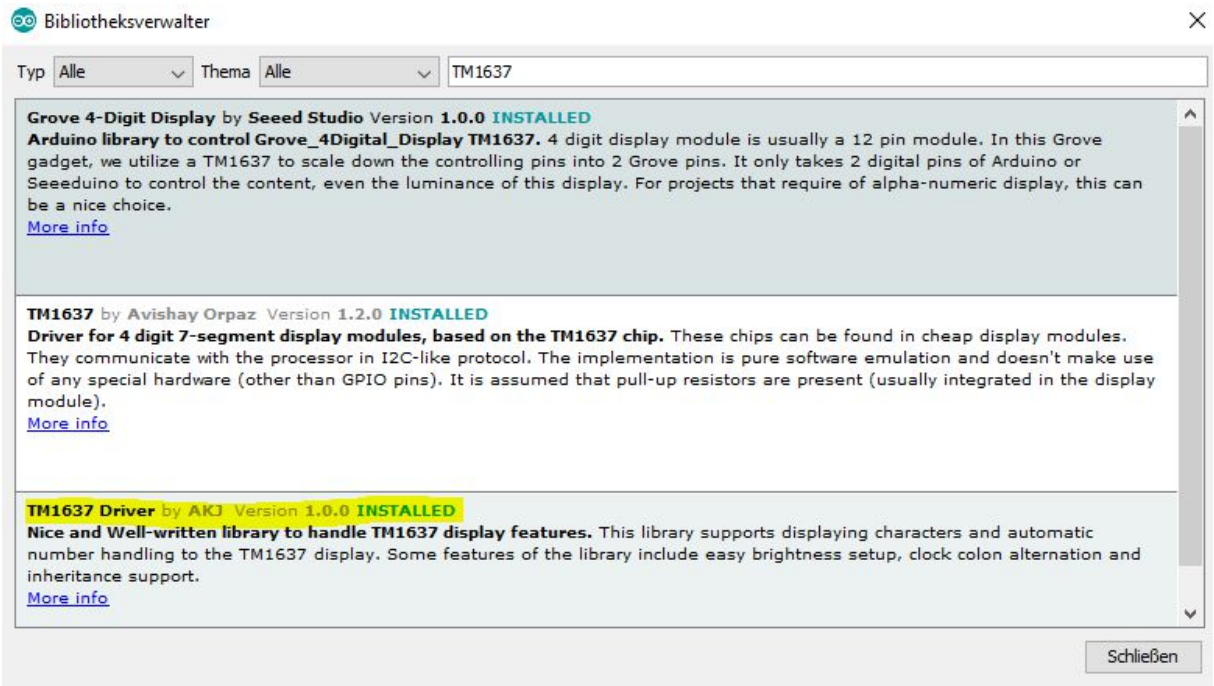
Wir verdrahten die Teile wie auf folgendem Bild ersichtlich miteinander:



Die beiden Taster dienen zum Einstellen der Uhr.

AZ-Delivery Nano V3	4 Digit 7 Segment Display
D4	DIO
D4	CLK
+5V	VCC
GND	GND

Bevor wir nun zur eigentlichen Programmierung schreiten können, müssen wir uns noch die TM1637 Driver by AKJ Bibliothek zur Ansteuerung des TM1637 installieren. Nur diese unterstützt bei gleichzeitigem kleinem Speicherbedarf die einfache alternierende Ansteuerung der Doppelpunkte durch einen Befehl.



Sobald die Bibliothek installiert ist, kann folgender Code hochgeladen werden:

```
#include <TM1637.h>
// Code by Tobias Kuch 2019, License unter GPL 3.0

// Initialisation and pin configuration
// Pin 4 - > DIO
// Pin 5 - > CLK
TM1637 tm1637(4, 5);


#define BUTTON_MINUTEUP_PIN  2   // Digital IO pin connected to the button.  This will be
                  // driven with a pull-up resistor so the switch should
                  // pull the pin to ground momentarily.  On a high -> low
                  // transition the button press logic will execute.
```

```
                // Used for Setting the Clock Time

#define BUTTON_HOURUP_PIN   3    // Digital IO pin connected to the button.  This will be
                // driven with a pull-up resistor so the switch should
                // pull the pin to ground momentarily.  On a high -> low
                // transition the button press logic will execute.
                // Used for Setting the Clock Time


// interrupt Control
bool SecInterruptOccured = true;
bool A60telSecInterruptOccured = true;
byte A60telSeconds24 = 0;


// Clock Variables
byte Seconds24;
byte Minutes24 ;
byte Hours24;
bool DisableSecondDisplay = false;
bool MinSetQuickTime = false;
bool HourSetQuickTime = false;
bool ButtonDPress = false;
bool ButtonEPress = false;

//Interrupt Routines

ISR(TIMER1_COMPA_vect)
{
 A60telSeconds24++;
  tm1637.switchColon();
 if ((A60telSeconds24 > 59) and !(MinSetQuickTime))
   {
     A60telSeconds24 = 0;
    //Calculate Time 24 Stunden Format
    SecInterruptOccured = true;
    Seconds24++;
    if (Seconds24 > 59)
      {
        Seconds24 = 0;
        Minutes24++;
      }
    if (Minutes24 > 59)
      {
        Minutes24 = 0;
        Hours24++;
      }
    if (Hours24 > 23)
      {
        Hours24 = 0;
      }
   }
```

```
   if  (MinSetQuickTime)
   {
      A60telSeconds24 = 0;
    //Calculate Time 24 Stunden Format
    SecInterruptOccured = true;
    Seconds24++;
    if (Seconds24 > 59)
     {
      Seconds24 = 0;
      Minutes24++;
     }
    if (Minutes24 > 59)
     {
      Minutes24 = 0;
      Hours24++;
     }
    if (Hours24 > 23)
     {
      Hours24 = 0;
     }
   }

  TCNT1 = 0;      // Register mit 0 initialisieren


if  (HourSetQuickTime)
   {
    OCR1A =  200;
   } else
   {
    OCR1A =  33353;     // Set Output Compare Register
   }
  A60telSecInterruptOccured = true;
}




//Interrupts ende

void CheckConfigButtons ()    // InterruptRoutine

{

 bool PressedZ;

PressedZ= digitalRead(BUTTON_MINUTEUP_PIN);
if ((PressedZ == LOW) and (ButtonDPress == false))
  {
   ButtonDPress = true;
   delay(100);
   Minutes24++;
   Seconds24 = 0;  // Reset Seconds to zero to avoid Randomly time
```

```
   DisableSecondDisplay = true ;   // Disable Seconds While Clock Set
   MinSetQuickTime = true; //Enable Quick Tmime Passby

  }
if ((PressedZ == HIGH) and (ButtonDPress == true))
  {
  ButtonDPress = false;
  delay(100);
  DisableSecondDisplay = false ;   // Enable Seconds While Clock Set
  MinSetQuickTime = false;
  Seconds24 = 0;  // Reset Seconds to zero to avoid Randomly time
  A60telSeconds24 = 0;

  }

PressedZ= digitalRead(BUTTON_HOURUP_PIN);
if ((PressedZ == LOW) and (ButtonEPress == false))
  {
    ButtonEPress = true;
    delay(100);
    DisableSecondDisplay = true ;   // Disable Seconds While Clock Set
    MinSetQuickTime = true; //Enable Quick Tmime Passby
    HourSetQuickTime = true;
  }
if ((PressedZ == HIGH) and (ButtonEPress == true))
  {
    noInterrupts();
    ButtonEPress = false;
    delay(100);
    Minutes24++;
    DisableSecondDisplay = false ;   // Enable Seconds While Clock Set
    MinSetQuickTime = false; //Enable Quick Tmime Passby
    HourSetQuickTime = false;
    Seconds24 = 0;  // Reset Seconds to zero to avoid Randomly time
    A60telSeconds24 = 0;
    interrupts();
  }


}




void setup()
{
   tm1637.init();
   tm1637.setBrightness(8); // Highest Brightness
   pinMode(BUTTON_MINUTEUP_PIN, INPUT_PULLUP);
   pinMode(BUTTON_HOURUP_PIN, INPUT_PULLUP);
   digitalWrite(LED_BUILTIN, LOW);
   noInterrupts();
   TCCR1A = 0x00;
```

```
   TCCR1B =  0x02;
   TCNT1 = 0;      // Init Register
   OCR1A =  33353;     // Output Compare Register vorbelegen
   TIMSK1 |= (1 << OCIE1A);  // Timer Compare Interrupt aktivieren
   interrupts();
   Seconds24 = 1;
   Minutes24 = 1;
   Hours24 = 0;
   tm1637.dispNumber(Minutes24 + Hours24 * 100);
}


void DisplayClockOnLedTM1637()

{
 tm1637.switchColon();
 tm1637.dispNumber(Minutes24 + Hours24 * 100);
}

void loop()
{
 bool PressedC;
 if (A60telSecInterruptOccured)
   {
    A60telSecInterruptOccured = false;
   }
 if (SecInterruptOccured)
   {
      SecInterruptOccured = false;
      DisplayClockOnLedTM1637();
    }
  CheckConfigButtons();
}
```

In den nachfolgenden Teilen werden wir unserer Uhr noch ein paar schöne Features spendieren.

Ich wünsche viel Spaß und bis zum nächsten Mal.