Smart Traffic Light

Bushra Yasin Ahmed Mohamed Fitim Faiku



Inhalt

[Inhalt 1](#_Toc30431742)

[Projektbeschreibung 2](#_Toc30431743)

[Blockschaltbild 2](#_Toc30431744)

[Module 2](#_Toc30431745)

[Ultraschallsensor 2](#_Toc30431746)

[1.Modus 2](#_Toc30431747)

[Berechnung der Entfernung 3](#_Toc30431748)

[Buzzer 3](#_Toc30431749)

[Clock Modul 4](#_Toc30431750)

[RTC real time clock (DS1302) 4](#_Toc30431751)

[LC-Display 5](#_Toc30431752)

[Quellen 5](#_Toc30431753)

[LED Streifen 6](#_Toc30431754)

[3D Model 6](#_Toc30431755)

[State Diagramm 6](#_Toc30431756)

# Projektbeschreibung

Das Ziel dieses Projekts ist eine smarte Ampelsteurung zu entwickeln. Dabei werden drei Mikrokontroller verwendet. An zwei dieser Mikrokontroller werden die Ampeln angehängt und mit dem Dritten werden diese gesteuert. Des Weiteren werden Ultraschallsensoren verwendet um zu testen ob eine Person zum oder ein Auto an der jeweiligen Ampel steht. Bei der Fußgängerampel soll auch ein Buzzer eingesetzt werden der in der Grünphase angehen soll. Ebenfalls soll ein Display und ein Clock Modul verbaut werden. Das Display soll die Aktuelle Uhrzeit anzeigen und wie lange es noch dauert bis die Ampel umschaltet. Die Verbindung zwischen den drei Mikrokontrollern erfolgt über SPI.

# Blockschaltbild

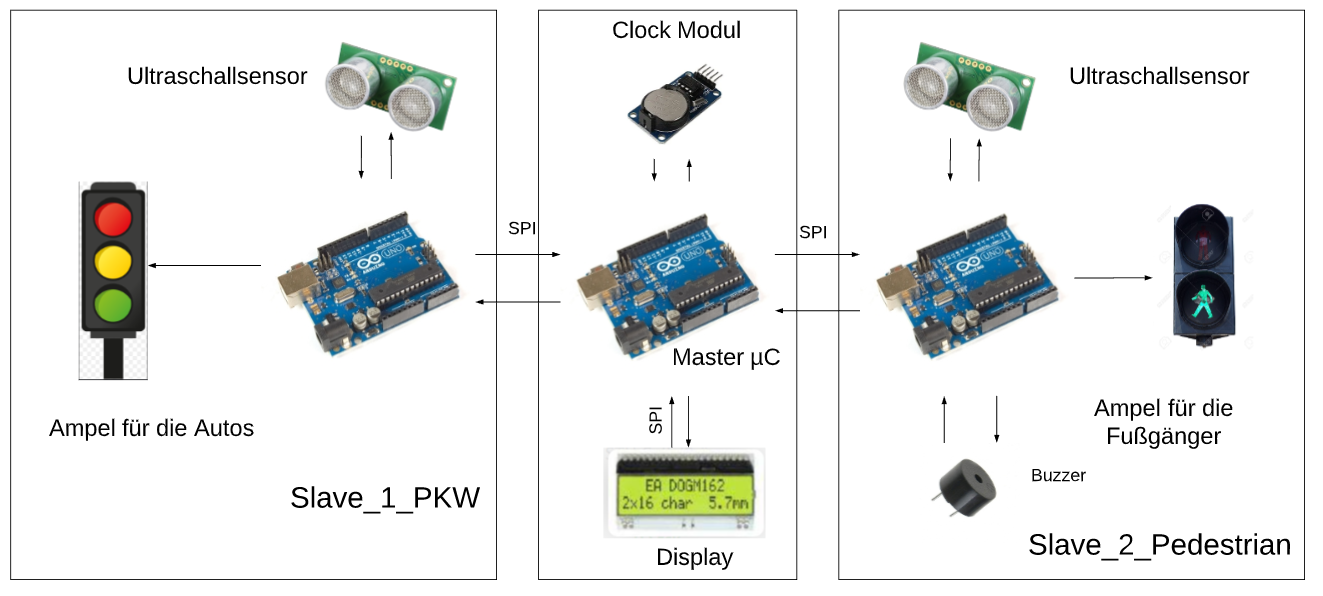


Abbildung 1 Blockschaltbild

# Module

## Ultraschallsensor

Als Ultraschallsensor wurde der SRF05 verwendet. Dieser hat eine Reichweite von 4 Meter. Der SRF05 hat zwei Modi mit denen er in Betrieb genommen werden kann. Bei der Verwendung des Sensors wurde der erste Modus benutzt.

### 1.Modus

Bei diesem Modus wird eine Messung über den Trigger Input Pin des Sensors ausgelöst, und anschließend wird auf dem Echo Output Pin das Echo der Ultraschallwelle gemessen. [1]

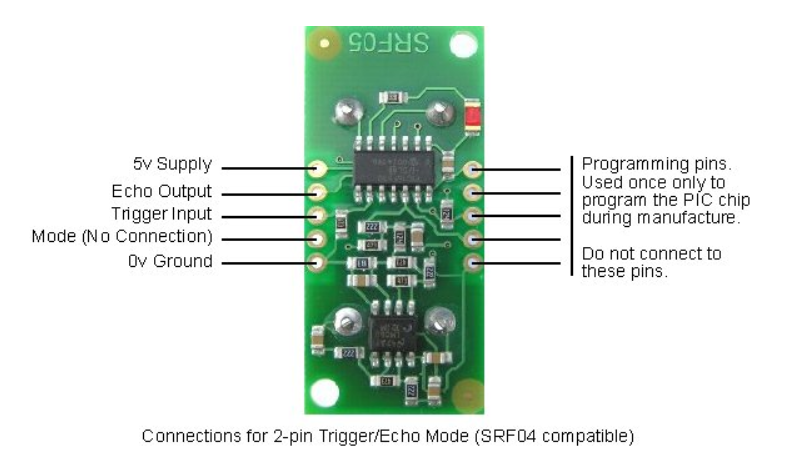


Abbildung 2 Anschlüsse des SRF05 für den 1.Modus [1]

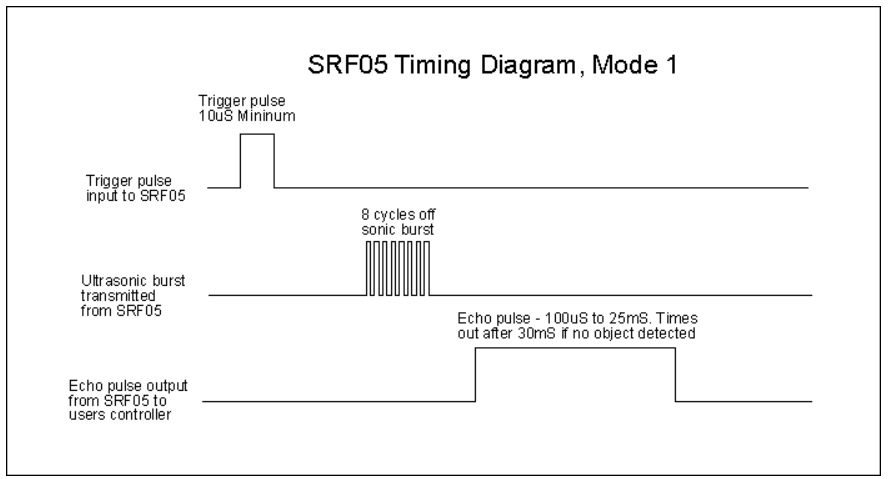


Abbildung 3 Timing Diagramm SRF05 [1]

## Berechnung der Entfernung

Für die Erfassung der Entfernung wird die Zeit, die der Echo Output Pin des SRF05 High ist, gemessen. Um die gemessene Zeit, die in Millisekunden angegeben wird, in Zentimeter umzuwandeln wurde folgende Formel verwendet:

Distanz in cm = Zeit in Millisekunden /29/2 [2]

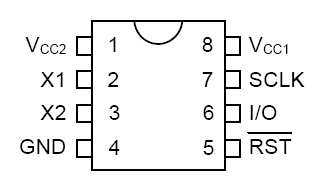
## Buzzer

Als Buzzer wurde der RS 7800712 verwendet. Diese wird mit einer Frequenz angesteuert um einen Ton zu erzeugen. Je höher die Frequenz ist, desto höher ist der Ton der erzeugt wird.

## Clock Modul

## RTC real time clock (DS1302)

Dieses Modul funktioniert wie eine physikalische Uhr. Es misst die Zeit. Wenn die real time clock nicht mit Vcc verbunden ist, läuft die RTC trotzdem weiter, sodass immer die aktuelle Uhrzeit angezeigt wird. Denn die RTC hat eine Batterie. d.h. wenn ein Stromausfall passiert, kann die Zeit weiter ohne Hindernisse erfasst werden.



Die DS1302 hat ein eigenes Protokoll (ähnlich, aber anders als I²C), das per Software auf beliebigen Pins realisiert wird. Sie hat ein sogenannte Simple Serial Interface. Man braucht dafür 3 PINS: CLK (Clock Leitung), DATA (Input/Output) und RST (CE). [3]

Jedes Mal wenn man den Register beschreiben: ein Byte (Hour, minutes, oder secondes...) auf den Register schreiben, muss das Modul resetet werden.

Bei Reset wird die Clock Leitung Low sein und CE wird Low dann High sein.

void DS13xx\_Reset(void) {

PORTD &= ~(1<<DS13xx\_SCLK);

PORTD &= ~(1<<DS13xx\_CE);

PORTD |= (1<<DS13xx\_CE);

}

Die zweite Möglichkeit den Register zu beschreiben ist Burst Mode. Hier werden alle Bytes gelesen oder geschrieben, dann wird die RTC resetet:

void DS13xx\_Write\_CLK\_Registers(void) {

DS13xx\_Reset();

DS13xx\_WriteByte(0xbe); /\* clock burst write (eight registers) \*/

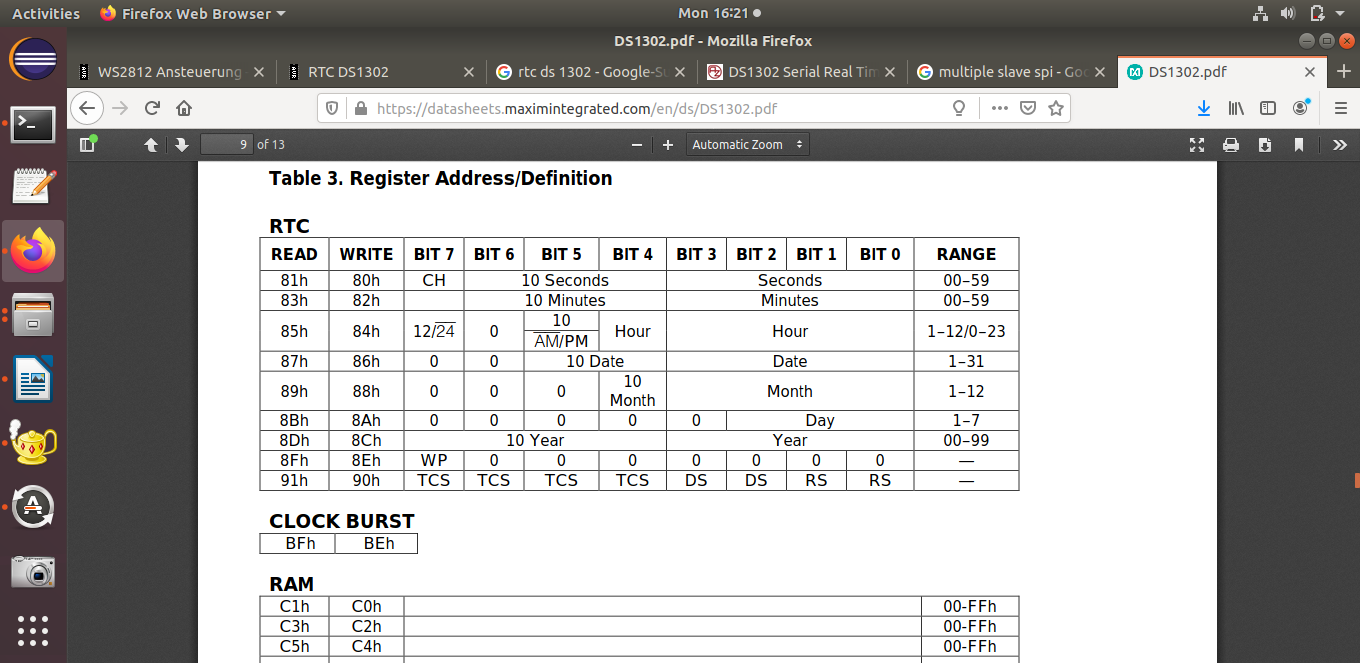
DS13xx\_WriteByte(ClockSecond);

DS13xx\_WriteByte(ClockMinute);

DS13xx\_WriteByte(ClockHour);

DS13xx\_Reset();

}



Wichtig auch bei diesem Modul ist, dass man die Data Leitung nicht anschließt, wenn man minicom startet, weil der Code von oben nach unten ausgeführt wird. D.h. DS13xx\_Init() in der main wird auch ausgeführt → die Zeit im Register wird überschrieben und das Modul beginnt die Zeit von Anfang an zu messen.

void DS13xx\_Init(void) {

// Initialise PORTA for the DS13xx clock

DDRD |= (1<<DS13xx\_CE) | (1<<DS13xx\_IO) | (1<<DS13xx\_SCLK);

DS13xx\_WriteByte(0x8E);

DS13xx\_WriteByte(0x00);

DS13xx\_Reset();

DS13xx\_WriteByte(0x80);

DS13xx\_WriteByte(0x00);

DS13xx\_Reset();

DS13xx\_WriteByte(0x81);

DS13xx\_Reset();

}

## LC-Display

## LED Streifen

# 3D Model

# State Diagramm

# Retrospektive

# Quellen

|  |  |
| --- | --- |
|  | <https://www.robot-electronics.co.uk/htm/srf05tech.htm> |
| [2] | <https://create.arduino.cc/projecthub/Nicholas_N/distance-measurement-with-an-ultrasonic-sensor-hy-srf05-64554e> |
| [3] | <https://datasheets.maximintegrated.com/en/ds/DS1302.pdf> |
|  |  |
|  |  |