Praktikum Digitaltechnik

IR - Remote

Benedikt Schnörr, Sebastian Pasinski

# Gliederung

[**Gliederung 2**](#_9b6hsne96upf)

[**Interrupts für Schalter 3**](#_4n0yxcfkanp8)

[**Schnittstelle zwischen Interrupts und RC5-Code 5**](#_3j3njq9nyn57)

[**Sleep-Befehl 5**](#_9tfw93bdxetd)

[**Hardware 5**](#_i87iuhoiodqg)

[**Quellen 6**](#_bnortr7ryggc)

# Interrupts für Schalter

Es befinden sich an den Pins RB0, RB1, RB2, RB4 und RB5 Schalter. Um das Drücken der Schalter an der Fernbedienung zu erkennen, musste das Programm mit den Interrupts um weitere Unterbrechungen erweitert werden. Es galt, zusätzlich zum Interrupt 0 die Interrupts 1,2 und die Pins RB4 und RB5 als Interrupts festzulegen.

clrf PORTB

bcf ADCON0,ADON ; AD Converter disabled

movlw 0xFF

movwf TRISB

movlw 0xFF

movwf ADCON1

; Clear Interrupt flags

bcf INTCON, INT0IF

bcf INTCON3, INT1IF

bcf INTCON3, INT2IF

bcf INTCON, RBIF

; enable pullups

bcf INTCON2, RBPU

; Enable Interrupts

bsf INTCON, INT0IE

bsf INTCON3, INT1IE

bsf INTCON3, INT2IE

bsf INTCON, RBIE

; triggers interrupts when falling edge occurs on RB0-2

bcf INTCON2, INTEDG0

bcf INTCON2, INTEDG1

bcf INTCON2, INTEDG2

; Interrupt priority of int1 / 2 needs to be configured

bsf INTCON, GIEH ; Enables high and low Priorities

bsf INTCON, GIEL

bsf RCON, IPEN ; Enables Priorities

; Setting Priority high for INT1, INT2 and RB On Change

bsf INTCON3, INT2IP

bsf INTCON3, INT1IP

bsf INTCON2, RBIP

Um in der Anwendung des Programms nachvollziehen zu können, welcher Schalter gedrückt wurde, wurde ein Algorithmus innerhalb der High Interrupt Routine implementiert, welcher die jeweiligen Interrupt-Flags abfragt oder entsprechend die Zustände an den Pins. Wenn der entsprechende Schalter erkannt wurde, wird eine Sendefunktion mit einem bestimmten RC-5 Befehl ausgeführt.

HighInt:

btfsc INTCON, RBIF ; was pin 4/5 pressed?

bra RBonChange

;Routine to find out on what Pin the high signal was

In der Routine wurde dabei zuerst ermittelt, ob sich eine Änderung an den Pins RB4 und RB5 ereignet hat, um diese anhand der Pin Zustände in der Routine RBonChange abzufragen.

btfsc INTCON, INT0IF ; was Pin 0 pressed?

bra SendRB0 ; Yes Pin 0 was pressed, send command

btfsc INTCON3, INT1IF ; was Pin 1 pressed?

bra SendRB1

btfsc INTCON3, INT2IF ; was Pin 2 pressed

bra SendRB2

retfie

RBonChange:

btfsc PORTB, 4 ;High Signal on RB4?

bra SendRB4; in routine sende befehl und return

btfsc PORTB, 5 ;Test Port RB5

bra SendRB5;

bcf INTCON, RBIF ; else none of the relevant pins

retfie;

In den jeweils aufgerufenen Senderoutinen wird das Interruptflag zurückgesetzt, um mehrfache Unterbrechungen zu vermeiden und wieder in das Hauptprogramm zu springen. Im Folgenden ist eine beispielhafte Routine dargestellt.

SendRB0:

; Sending Routine for Command 1

nop

bcf INTCON, INT0IF ; clearing Flag to avoid Double interrupt

retfie

Die Schnittstelle wurde am Modell getestet, um das Programmverhalten beim Drücken der Taster zu beobachten.

# Schnittstelle zwischen Interrupts und RC5-Code

Die Schnittstelle zwischen den beiden weitestgehend unabhängig entwickelten Programmteilen zu den Interrupts und der RC5-Übertragung besteht aus zwei Bit-Buffern. Diese werden in den Sende-Routinen bei einem High Interrupt auf den passenden Befehl gesetzt. [2] Die 14 Bits im Bit-Buffer, die den Befehl enthalten, werden daraufhin als RC5-Code mit dem bereits im letzten Bericht erläuterten Programm gesendet. Beispielhaft ist hier die Sende-Funktion für die Stop-Taste dargestellt.

send\_rb\_0: ; send stop command

movlw b'00110101'

movwf bit\_buffer\_h

btfsc bit\_buffer\_toggle ; set toggle bit in bit buffer

bsf bit\_buffer\_h,3

movlw b'00110110'

movwf bit\_buffer\_l

bcf INTCON,INT0IF

return

In diesem Code-Abschnitt wird auch das Toggle-Bit behandelt. Dabei handelt es sich um ein Bit im Code, das bei jedem neuen Tastendruck getoggelt wird. Dadurch kann der Empfänger einen langen Tastendruck, bei dem der Befehl durchgängig gesendet wird und das Toggle-Bit den Wert beibehält, von einem erneuten Tastendruck unterscheiden.

# Sleep-Befehl

Damit im Batteriebetrieb Strom gespart wird, soll der PIC bei Inaktivität in den Sleep-Modus gesetzt werden. Um einen Sleep- Befehl ausführen zu können, wird im OSCCON-Register das IDLE-Bit zurückgesetzt. Dann kann der Mikrocontroller in den Sleep-Modus versetzt werden. [1]

bcf OSCCON, IDLEN

sleep

# Hardware

Anhand der finalen Layouts aus dem vorherigen Bericht wurde seitens des Labors eine Platine erstellt und mit Bauteilen bestückt. Diese ist in Abb. 1 zu sehen.

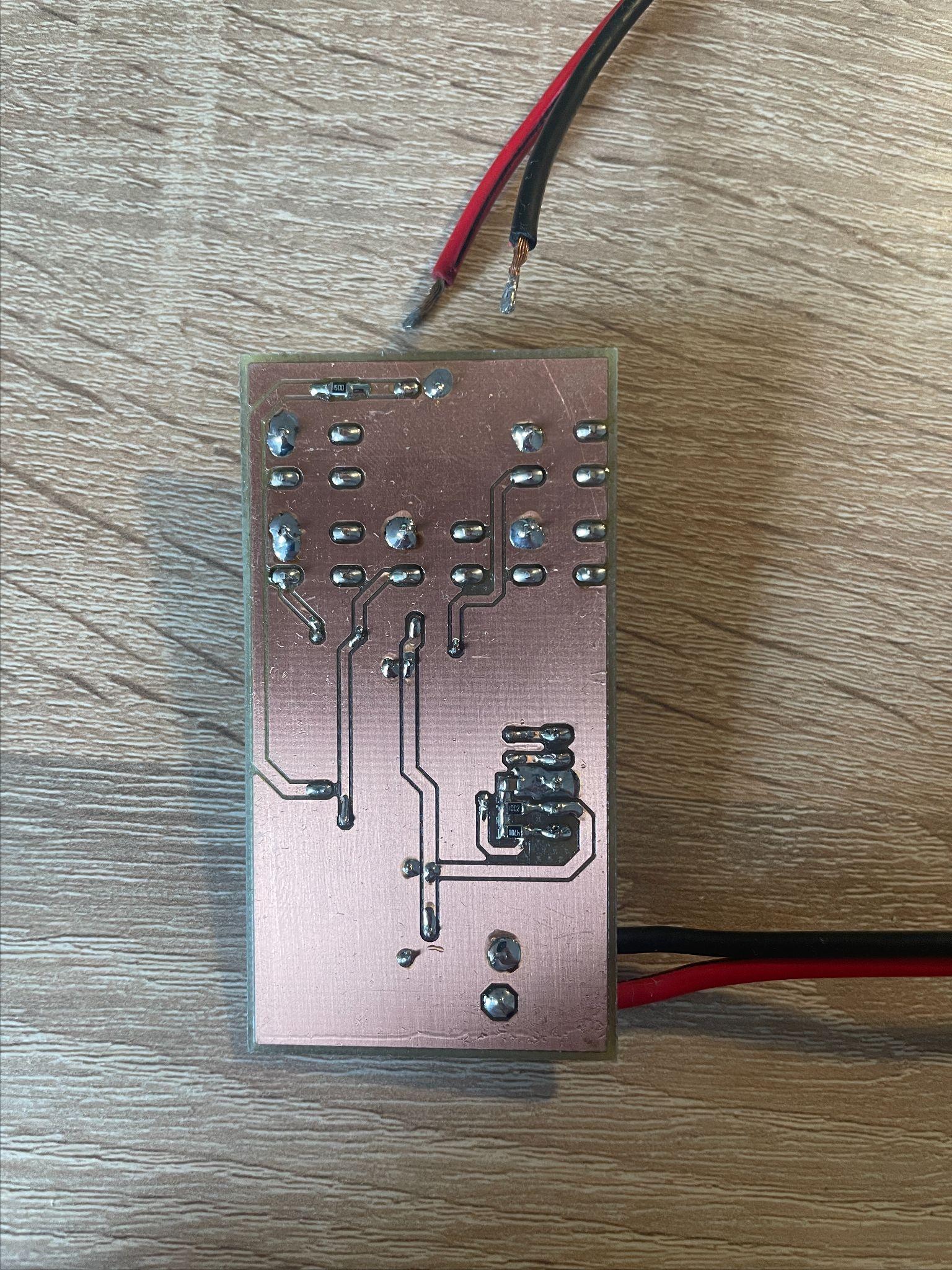
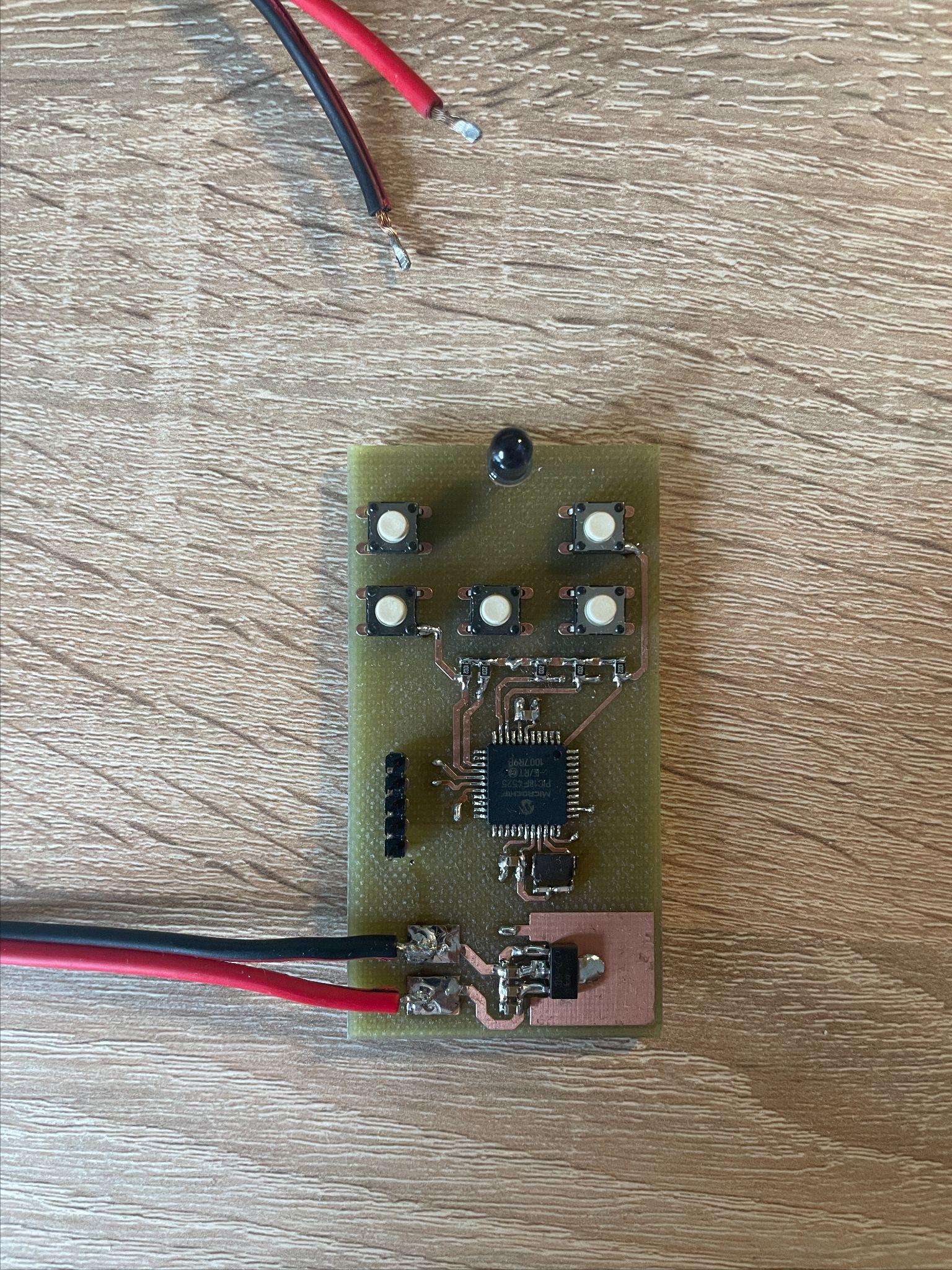


Abb. 1: Vom Labor bestückte Platine

# Quellen

[1] <https://ww1.microchip.com/downloads/en/devicedoc/39626e.pdf> S.37

[2] <https://www.opendcc.de/info/rc5/rc5.html>