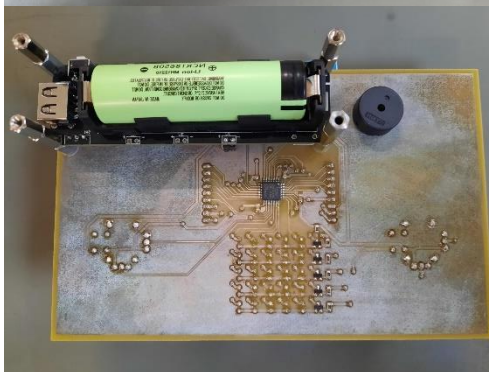
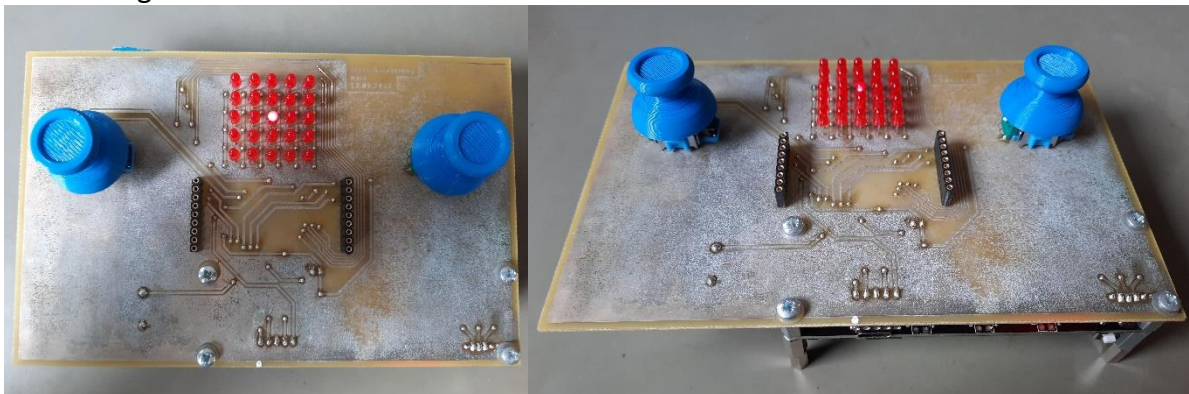


Zusammenfassung uC-Projekt Joysticksteuerung

Die Vorgaben für das Projekt sind 60h und ich es soll einen sinnvollen Eigennutzen haben. Eine vollumfängliche Dokumentation sowie eine Präsentation müssen ebenfalls gemacht werden.

Da ich die herkömmlichen Controller wie zum Beispiel von einer PS4 nicht so bequem finde wollte ich versuchen einen eigenen zu bauen und programmieren. Es ist ein Controller mit 2 Joysticks, wobei der linke Joystick seine Position über den Analog-Digital-Wandler (ADC) eines Mikrokontrollers an eine 5x5 LED-Matrix weitergibt. Als Mikrokontroller habe ich einen STM32F303CCT7 mit 48 Anschlüssen gewählt, der auch andere Schnittstellen als den ADC besitzt, wie man rechts sehen kann. Ausserdem habe ich auch einen Buzzer und den Rest der unbenutzten Ausgänge des Mikrokontrollers an Buchsenleisten angehängt. Damit das Ganze läuft habe ich einen Akku mit einer speziellen Leiterplatte, die den Akku ladet und vor Überladung schützt. Das Produkt sieht dann so aus.

| | | |
|---|---|--|
| System Power supply 1.2V regulator POR/PDR/PVD Xtal oscillators 32 kHz + 4 to 32 MHz Internal RC oscillators 40 kHz + 8 MHz PLL Clock control RTC/AWU 1x SysTick timer 2x watchdogs (independent and window) 51.8kV/115Vps Cyclic redundancy check (CRC) Touch-sensing controller 24 keys | 72 MHz ARM® Cortex®-M CPU Flexible Static Memory Controller (FSMC) Floating point unit (FPU) Nested vector interrupt controller (NVIC) Memory Protection Unit (MPU) JTAG/SW debug/ETM | Up to 512-Kbyte Flash memory Up to 64-Kbyte SRAM 32 PC Up to 16-Kbyte DCM-SRAM 64 bytes backup register |
| Control 3x 16-bit (144 MHz) motor control PWM Synchronized AC timer 1x 32-bit timers 5x 16-bit timers | Interconnect matrix AHB bus matrix 12-channel DMA | Connectivity 4x SPI (with 2x full duplex FS) 3x I2C 1x CAN 2.0B 1x USB 2.0 FS 5x USART/LART LIN, smartcard, I2C, modem control |
| | | Analog 2x 12-bit DAC with basic timers 4x 12-bit ADC channels / 5 MSPS 4x Programmable Gain Amplifiers (PGA) 7x comparators (25 ns) Temperature sensor |



Den ADC habe ich mit einem Hilfsprogramm in Betrieb genommen und die Ausgänge für die LED-Matrix ohne Hilfe programmiert. Dann habe ich das eigentliche Programm geschrieben, das je nach Werten des ADCs die entsprechenden LEDs einschaltet und alle anderen ausschaltet.

So wie das Produkt jetzt ist, funktioniert es sehr gut aber man kann noch viele Sachen verbessern, wie zum Beispiel einige kleine Fehler im Schema und Layout der

Leiterplatte, einen Reset-Schalter für den Mikrokontroller oder den Code kompakter schreiben. Ebenso hat dieses Projekt sehr viel Raum für Erweiterungen, da ich die unbenutzten Anschlüsse an Buchsenleisten angehängt habe. Damit kann ich in der Zukunft zum Beispiel ein Bluetooth-Modul einbauen, um kabellos mit dem Computer zu verbinden. Weitere Möglichkeiten sind auch ein Display, Taster oder Schalter für Spezialfunktionen.

Im Allgemeinen ist es ein Projekt bei dem ich sehr viel über ADCs gelernt habe und auch wie man einen Mikrokontroller benutzt und programmiert.