

Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

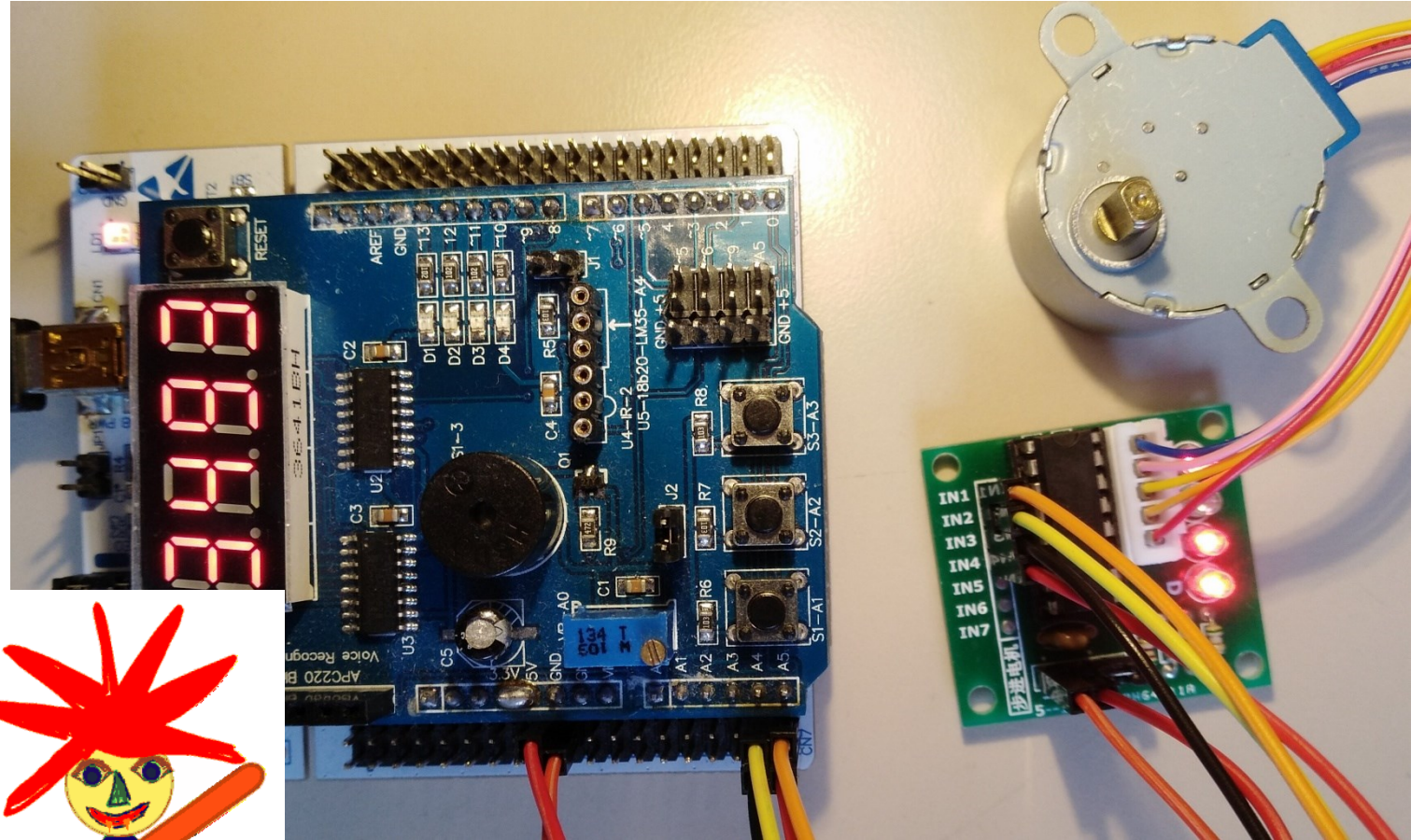
Multifunctionshield + Schrittmotorshield



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

Projekthighlights:

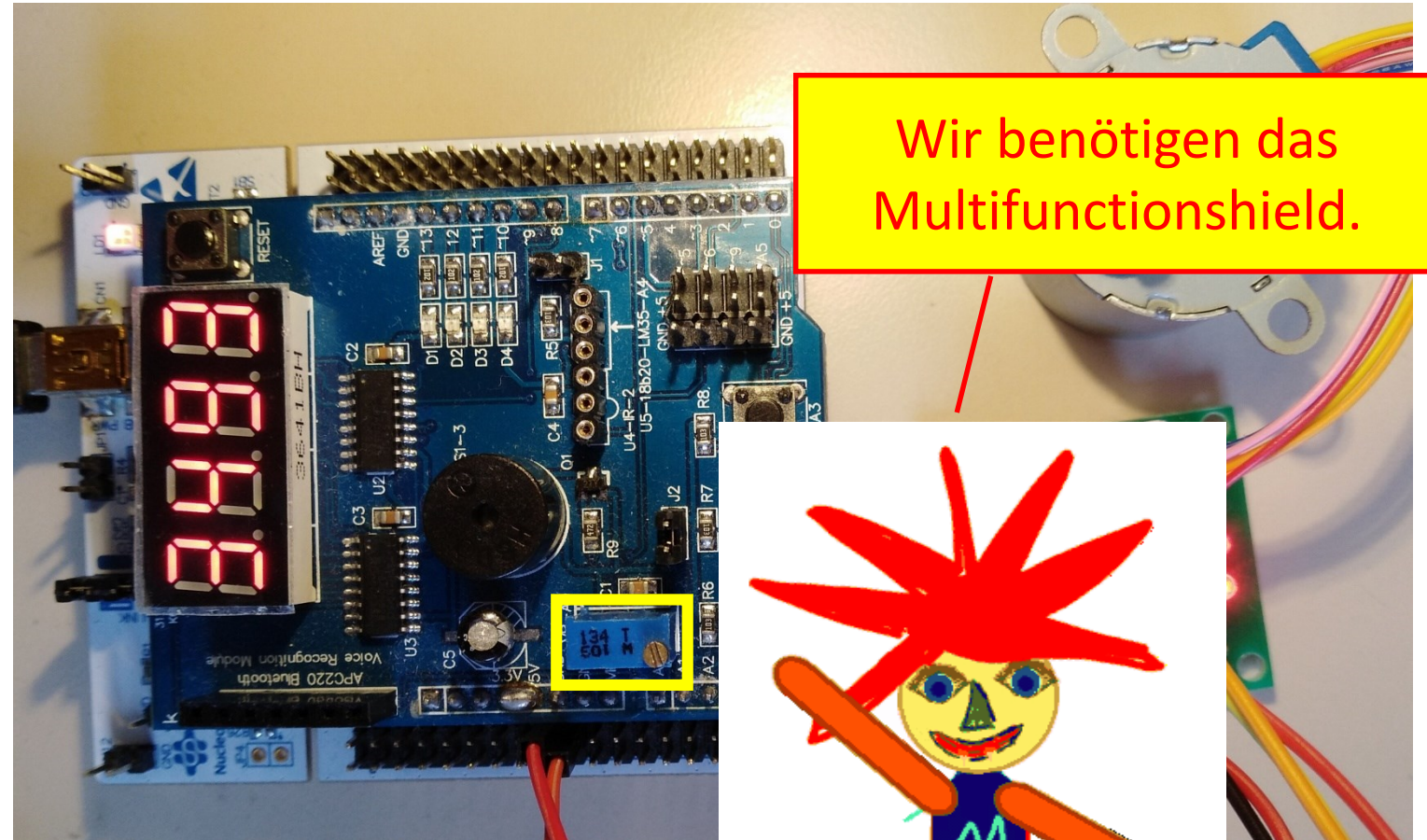
- Schrittmotor an PC3..PC0
- Multifunctionshield
 - Analoge Geschwindigkeitsvorgabe mit Poti
 - 4-stellige Geschwindigkeitsanzeige auf Siebensegmentanzeige
 - Interruptgesteuerte Richtungsumkehr mit Taste A1



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

Projekthighlights:

- Schrittmotor an PC3..PC0
- Multifunctionshield
 - Analoge Geschwindigkeitsvorgabe mit Poti
- 4-stellige Geschwindigkeitsanzeige auf Siebensegmentanzeige
- Interruptgesteuerte Richtungsumkehr mit Taste A1



Stufe 2: Analoge Geschwindigkeitsvorgabe



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

Projekthighlights:

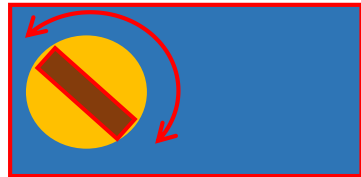
- Schrittmotor an PC3..PC0
- Multifunctionshield
 - Analoge Geschwindigkeitsvorgabe mit Poti
- 4-stellige Geschwindigkeitsanzeige auf Siebensegmentanzeige
- Interruptgesteuerte Richtungsumkehr mit Taste A1



Stufe 2: Analoge Geschwindigkeitsvorgabe



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe



PA0

$R0=0 \dots 4095$

Mikrocontroller



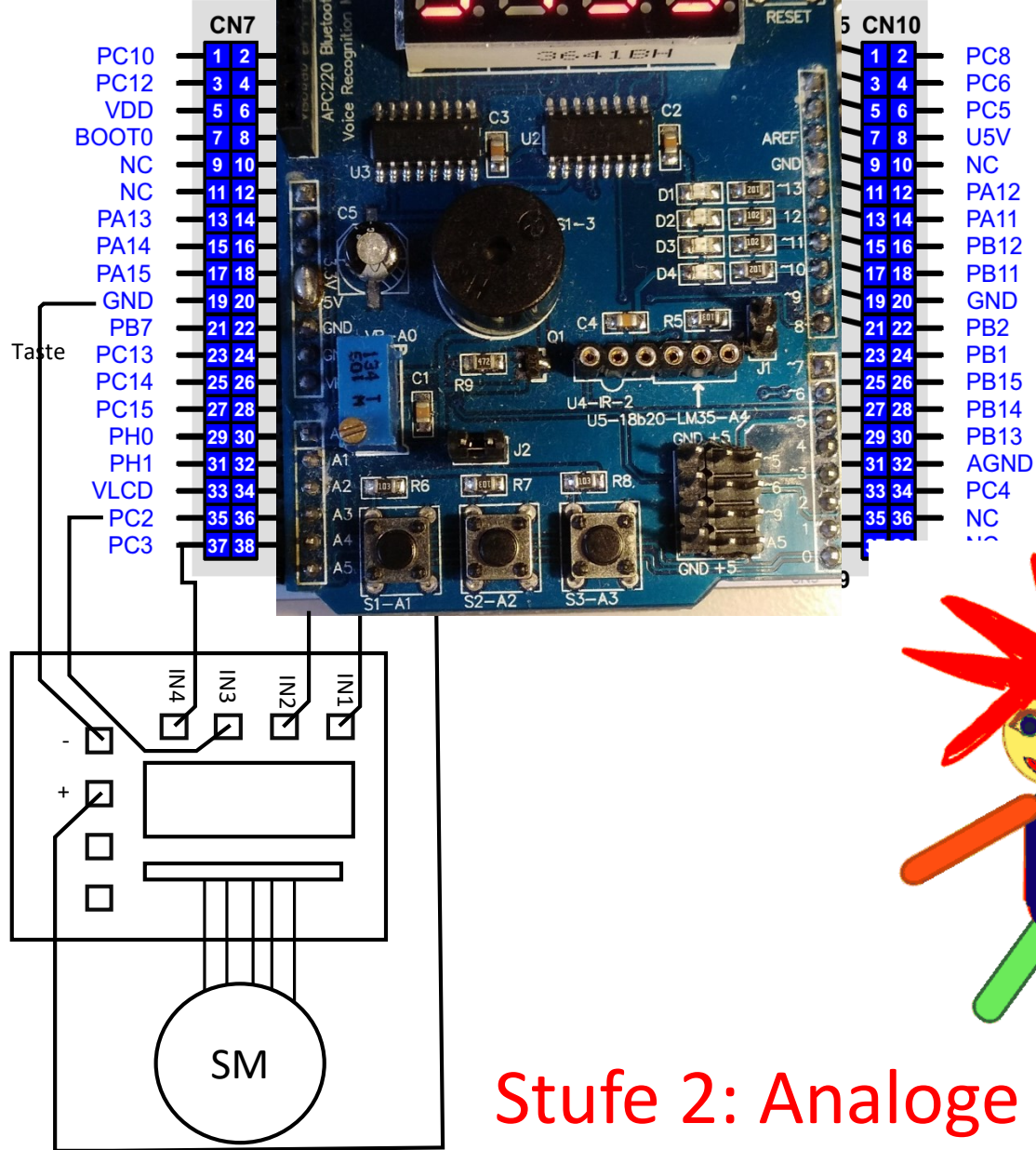
Die Einstellung des Potis verändert einen Zahlenwert von 0 bis 4095



Stufe 2: Analoge Geschwindigkeitsvorgabe



Projekt Schrittmotor mit Sicherungsstromanzeige und analoger Ausgabe



Anschlussbelegung und Konfiguration:

Nucleo	Schrittmotor
CN7 Pin 38 PC0	IN1 GPIO-Output
CN7 Pin 36 PC1	IN2 GPIO-Output
CN7 Pin 35 PC2	IN3 GPIO-Output
CN7 Pin 37 PC3	IN4 GPIO-Output
CN7 Pin 18 +5V	+
CN7 Pin 19 GND	-
	Multifunctionshield
CN8 Pin1 PA0	Poti PA0



Der Anschluss erfolgt einfach durch Aufstecken des Multifunctionshield

Stufe 2: Analoge Geschwindigkeitsvorgabe



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

PA0 als ADC_IN0 konfigurieren

B1 [Blue PushButton]

RCC_OSC32_IN

RCC_OSC32_OUT

RCC_OSC_IN

RCC_OSC_OUT

GPIO_Output

GPIO_Output

GPIO_Output

GPIO_Output



PA0-WKUP1

Reset State

ADC_IN0

COMP1_INP

RTC_TAMP2

SYS_WKUP1

TIM2_CH1

TIM2_ETR

TIM5_CH1

TIMX_IC1

TS_G1_IO1

USART2_CTS

GPIO_Input

GPIO_Output

GPIO_Analog

EVENTOUT

GPIO_EXTIO

ADC_IN0

PA0-...



Stufe 2: Analoge Geschwindigkeitsvorgabe

Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe



Pinout & Configuration

ADC Mode and Configuration

Categories A->Z

System Core >

Analog >

ADC

COMP1

COMP2

DAC

OPAMP1

OPAMP2

Multimedia >

Mode

☒ IN0

☐ IN0b

Configuration

Reset Configuration

☒ NVIC Settings ☒ DMA Settings ☒ GPIO Settings

☒ Parameter Settings ☒ User Constants

Search (Ctrl+F)

ADC_Settings

Clock Prescaler

Bank to use

Resolution

Data Alignment

Scan Mode Disabled

Continuous Conversion Mode Enabled

Discontinuous Conversion Mode Disabled

DMA Continuous Requests Disabled

GPIO_Output PC1

GPIO_Output PC2

GPIO_Output PC3

VSSA

VDDA

ADC_IN0 PA0-...

GPIO_Input PA1

SART_TX PA2

1. Portpin als ADC_IN einstellen
z.B. PA0 für Poti

2. Einstellungen für ADC öffnen
unter Analog

3. Continuous Conversion Mode Enable

Stufe 2: Analoge
Geschwindigkeitsvorgabe



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

Aus der
Formelsammlung



2. Initialisierung

main:

```
ldr    R0,=hadc
bl     HAL_ADC_Start
```

//AD-Wandler starten starten

3. Endlosschleife

schleife: //Endlosschleife

...

//AD-Wandler abfragen Ergebnis in R0

```
ldr    r0,=hadc
bl     HAL_ADC_GetValue
```

//Wert vom AD-Wandler holen

...

```
b      schleife
```

Stufe 2: Analoge
Geschwindigkeitsvorgabe



Projekt Schrittmotor mit Siebensegmentanzeige und analoger Geschwindigkeitsvorgabe

HAL_ADC_GetValue
speichert den
Einstellwert in R0
Wir verwenden aber
R5 als
Geschwindigkeits-
variabel



2. Initialisierung

main:

```
ldr    R0,=hadc
bl     HAL_ADC_Start
```

//AD-Wandler starten starten

3. Endlosschleife

schleife: //Endlosschleife

...

)-Wandler abfragen Ergebnis in R0

```
ldr    r0,=hadc
bl     HAL_ADC_GetValue
mov    R5,R0
b      schleife
```

//Wert vom AD-Wandler holen

//Geschwindigkeit in R5

Stufe 2: Analog
Geschwindigkeitsvorgabe

