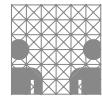


64-211 Übung Eingebettete Systeme



Aufgabenblatt 04 Termine: KW 20, KW 21

Gruppe	
Name(n)	Matrikelnummer(n)

4 Steuerung externer Komponenten

Die Steuerung von Aktoren ist ein Standard-Einsatzfall für eingebettete Systeme. Dabei werden verschiedene Typen von Aktoren zur Erledigung unterschiedlicher Aufgaben eingesetzt. Die Beispiele reichen von sehr kleinen Anwendungen (Steuerung des Antriebs eines RC-Fahrzeugs, Steuerung der Ruder eines Flugzeugs etc.) bis hin zu hochkomplexen verteilten Systemen (Prozesskontrolle in industriellen Fertigungsstraßen, Regelung des Kühlkreislaufs eines Atomkraftwerks etc.).

Gegenstand dieser Aufgabe ist die Entwicklung der Steuerung eines Gleichstrommotors. In Aufgabe 4.1 werden Sie ein über lediglich zwei Taster zu bedienendes Benutzerinterface und den für die Ablaufsteuerung erforderlichen Automaten entwickeln, während Sie in Aufgabe 4.2 die Ausgaben an den Benutzer implementieren und in Aufgabe 4.3 schließlich die Ansteuerung des Treiber-ICs des Motors realisieren werden.

Aufgabe 4.1 Steuerung eines Gleichstrommotores

Entwickeln Sie ein Programm, das die Ansteuerung eines Gleichstrommotors ermöglicht. Implementieren Sie dafür eine Benutzerschnittstelle, die es erlaubt, mit nur zwei Tastern sowohl die Leistung des Motors zu regeln, als auch zwischen den Rotationsrichtungen umzuschalten. Ihren Code aus Aufgabe 3.2 bzw. 3.3 zum Entprellen der Taster und zur Erkennung einer Doppel- bzw. Parallelbetätigung der Taster können Sie unmittelbar wiederverwenden. Erstellen Sie ein Programm für den Mikrocontroller, das eine Steuerung des Gleichstrommotors nach folgendem Schema ermöglicht:

- Doppelbetätigung der Taster
Wenn beide Taster gleichzeitig betätigt werden, dann soll zyklisch zwischen den beiden Zuständen (Bedien-Modi) umgeschaltet werden:
 1. *Einstellung der Rotationsrichtung*
 2. *Einstellung der Leistung*

- Einzelbetätigung der Taster

1. Im Zustand *Einstellung der Rotationsrichtung* haben die Taster folgende Funktion:¹

- Taster 1 verändert die Rotationsrichtung wie folgt: $CW \rightarrow STOP \rightarrow CCW$
- Taster 2 verändert die Rotationsrichtung wie folgt: $CCW \rightarrow STOP \rightarrow CW$

2. Im Zustand *Einstellung der Leistung* haben die Taster folgende Funktion:

- Taster 1 regelt die Leistung um eine von Ihnen definierte Schrittweite hoch, und nach Erreichen des Maximalwertes wird dort verblieben.
- Taster 2 verhält sich entgegengesetzt zur Funktion des Tasters 1.

Die **Initialisierung des Systems** soll im Modus *Einstellung der Rotationsrichtung* mit der Rotationsrichtung *STOP* erfolgen.

Aufgabe 4.2 Ausgaben des Systems

Ausgaben des Systems sollen über eine Dreifarben-LED und über die serielle Schnittstelle getätigt werden.

Die RGB-LED soll zur Anzeige des aktuellen Bedien-Modus des Systems eingesetzt werden:

1. *Einstellung der Rotationsrichtung* = „Grün“
2. *Einstellung der Leistung* = „Rot“
3. Ist in der Rotationsrichtung *STOP* gewählt, soll zusätzlich noch „Blau“ aktiviert werden.

Beachten Sie die Anordnung der Anschlüsse der **RGB-LED** (siehe Abbildung 1). Die Farbkanäle (Red, Green, Blue) der LED besitzen eine **gemeinsame Anode** (der „Pluspol“). Schließen Sie diese an VCC an. Schließen Sie jeden einzelnen Kanal über den zugehörigen Vorwiderstand an einen **PWM-fähigen** Anschlusspin des Arduinos an. Da bei der Simulation mit Proteus die PWM-Signale nicht in Realzeit generiert werden können, verwenden Sie bitte bei den Vorwiderständen das analoge Simulationsmodell (Edit Properties/Model Type - ANALOG) und bei den jeweiligen LEDs das digitale Simulationsmodell. Dann wird die langsam voranschreitende Simulationszeit berücksichtigt und die Helligkeit der LED angepasst. Sonst würde die LED entsprechend der voranschreitenden Simulationszeit im Takt des PWM-Signals blinken. Sie können für die Intensität aber auch nur die Werte 0 und 255 verwenden oder der Einfachheit halber ggf. gleich `digitalWrite()`.

Welche Auswirkung hat die gemeinsame Anode der RGB-LED auf die Ansteuerung der drei Farbkanäle mit der `analogWrite()` bzw. `digitalWrite()` Funktion?

Benutzen Sie für den Anschluss der RGB-LED in Abhängigkeit der Versorgungsspannung V_{CC} Ihres Aufbaues folgende Vorwiderstände:²

¹CW clock-wise : im Uhrzeigersinn

CCW counter clock-wise: gegen den Uhrzeigersinn

²Vgl. Sie hierzu die Aufgabe 1.2 des Aufgabenblattes 1 und berücksichtigen die unterschiedlichen Durchlassspannungen U_D der RGB-Dioden (vgl. **Datenblatt**).

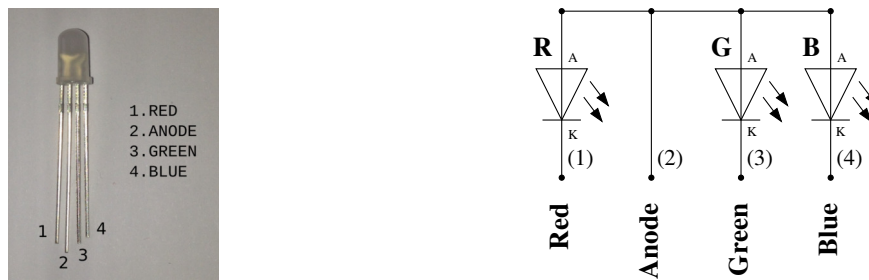


Abbildung 1: Anordnung der Anschlüsse der RGB-LED.

Farbe	U_D	V_{CC}		Proteus
		3.3 V	5.0 V	(RGBLED-CA)
Red	1,7 V	270 Ω	560 Ω	160 Ω
Green	3,1 V	27 Ω	330 Ω	100 Ω
Blue	3,1 V	27 Ω	330 Ω	100 Ω
		$I_D \approx 6mA$		$I_D \approx 20mA$

Tabelle 1: Die Vorwiderstände der jeweiligen Farben werden zum Schutz der Ausgangstreiber des Arduino für einen Durchlassstrom (I_D) von $\approx 6mA$ (vgl. [Datenblatt RGB-Led](#)) bemessen. Da das Proteus-Modell der RGB-LED (RGBLED-CA) deutlich abweichende Kennlinien aufweist und auch erst bei $I_D \approx 20mA$ ausreichend Licht emittiert wird, verwenden Sie bitte für die Simulation die Widerstandswerte der dritten Spalte.

Für die **Ausgaben des Systems** über den seriellen Port gilt:

- Informieren Sie den Benutzer immer über den Zustand des Systems, indem Sie Veränderungen der Kontrollwerte über den seriellen Monitor der Arduino IDE (sinnvoll präsentiert) ausgeben. Sollte eine Aktion des Benutzers mit keiner Veränderung der Kontrollwerte in Verbindung stehen, z.B. Veränderung der Leistung während Rotationsrichtung *STOP* eingestellt ist, erzeugen Sie eine entsprechende Ausgabe auf der seriellen Konsole.

In [Abb. 2](#) finden Sie ein Beispiel dafür, wie diese Information präsentiert werden könnte.

Aufgabe 4.3 Ansteuerung des Gleichstrommotors

Für die unmittelbare Ansteuerung des Gleichstrommotors verwenden wir ein Arduino Erweiterungsboard, das auf dem integrierten Schaltkreis **TB6612FNG** basiert. Dieses IC enthält die Steuerlogik und die Leistungsendstufen zur Ansteuerung der Motoren. Die Leistungsendstufen sind in Form einer **H-Brücke** verschaltet. Nähere Informationen zum Funktionsprinzip einer H-Brücke finden Sie unter: en.wikipedia.org/wiki/H_bridge.

Informieren Sie sich über die Anschlussbelegung (siehe [Abbildung 4](#)) und lesen Sie das zugehörige [Datenblatt](#). Für die Lösung der Aufgabe wird nahegelegt, den Versuchsaufbau gemäß [Abbildung 3](#) zu verdrahten. Für die Simulation mit Proteus verwenden Sie bitte anstelle des Breakout-Boards mit dem Motortreiber TB6612FNG direkt das Device TB6612FNG.

```

/dev/ttyACM0
[DIRECTION MODE] Motor direction: CCW - (STOP) - CW | Motor power: 0 %
-----
[POWER MODE] Motor direction: CCW - (STOP) - CW | Motor power: 0 %
ATTENTION: Motor is stopped! Please select direction.
-----
[DIRECTION MODE] Motor direction: CCW - (STOP) - CW | Motor power: 0 %
[DIRECTION MODE] Motor direction: CCW - STOP - (CW) | Motor power: 0 %
-----
[POWER MODE] Motor direction: CCW - STOP - (CW) | Motor power: 0 %
[POWER MODE] Motor direction: CCW - STOP - (CW) | Motor power: 3 %
[POWER MODE] Motor direction: CCW - STOP - (CW) | Motor power: 7 %
[POWER MODE] Motor direction: CCW - STOP - (CW) | Motor power: 11 %
[POWER MODE] Motor direction: CCW - STOP - (CW) | Motor power: 15 %
-----
[DIRECTION MODE] Motor direction: CCW - STOP - (CW) | Motor power: 15 %
[DIRECTION MODE] Motor direction: CCW - (STOP) - CW | Motor power: 0 %

```

Autoscroll Newline 115200 baud

Abbildung 2: Kommunikation des Systemzustandes über den seriellen Monitor.

Beachten Sie bei der Verdrahtung des Breakout-Boards bzw. des TB6612FNG Bausteins folgende Hinweise:

- Stellen Sie zunächst die Stromversorgung des TB6612FNG Bausteins sicher. Der Anschluss VCC Motor des Breakout-Boards bzw. die Pins 24, 14 und 13, also VM1, VM2 und VM3 des TB6612FNG-IC dienen der Versorgung der Motortreiber (2.5 – 13.5 V), während VCC Logik bzw. Pin 20 (Vcc) die Spannungsversorgung der Logik (2.7 V – 5.5 V) darstellt. Auf dem Breakout-Board sind alle GND-Anschlüsse boardseitig verbunden, während seitens des TB6612FNG-IC Motortreiber-GND (Pin 3, 4 - PGND1 und Pin 9, 10 - PGND2) und Logik-GND (Pin 18 - GND) getrennt sind.
- Da der TB6612FNG Baustein zwei H-Brücken enthält, haben Sie die Wahl zwischen zwei Kanälen: Kanal A (rot markierte Anschlüsse des Breakout-Boards, bzw. die Anschlüsse A01, A02, PWMA, AIN1, AIN2) und Kanal B (blau markierte Anschlüsse des Breakout-Boards, bzw. die Anschlüsse B01, B02, PWMB, BIN1, BIN2). Benutzen Sie einen der beiden Kanäle.
- Verbinden Sie den Kontrollanschluss PWM mit einem PWM-fähigen Anschlusspin des Mikrocontrollers. Stellen Sie sicher, dass der vom Microcontroller für die Generierung des PWM-Signals des von Ihnen gewählten Anschlusspins vorgesehene Timer nicht bereits durch Ihr Programm verwendet wird (siehe hierzu Aufgabenblatt 3, Abschnitt: 3.1). Die Eingänge IN1 und IN2 ermöglichen die Einstellung der im **Datenblatt** beschriebenen Steuerzustände. Die Anbindung des Gleichstrommotors muss über die Anschlüsse A OUT1,2 oder B OUT1,2 bzw. A01, 2 oder B01, 2 erfolgen.
- Vergessen Sie nicht, Stand-By bzw. STBY mit einem definierten Pegel zu versehen:

Stand-By	= HIGH	schaltet den Baustein ein
	= LOW	schaltet diesen aus

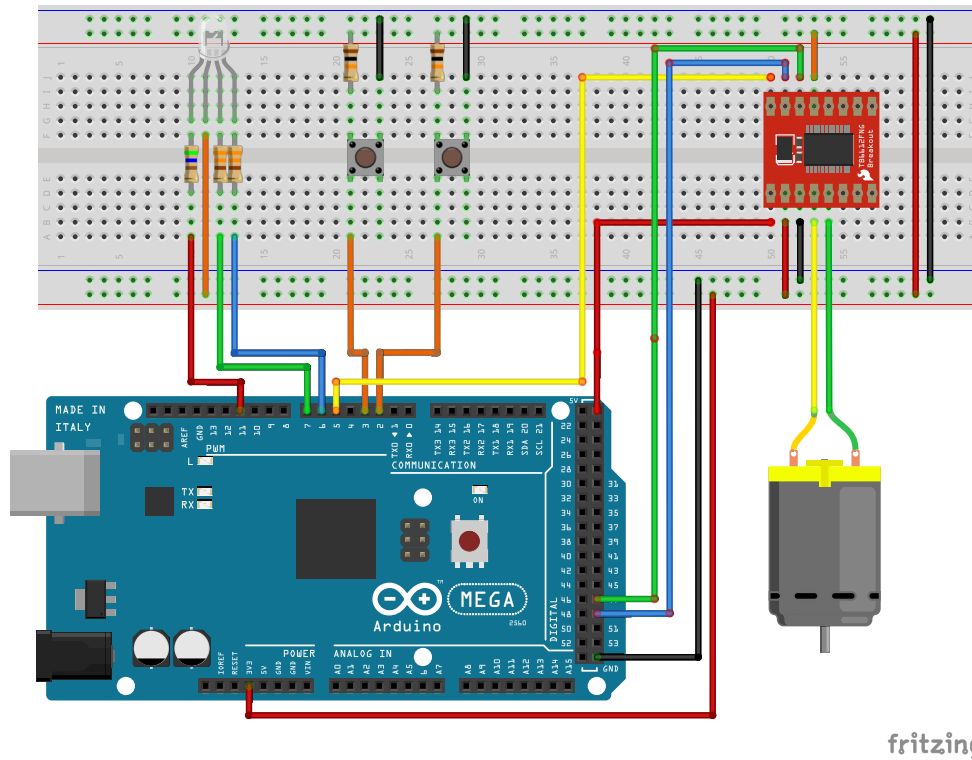


Abbildung 3: Vorschlag für die Verdrahtung des Versuchsaufbaus.

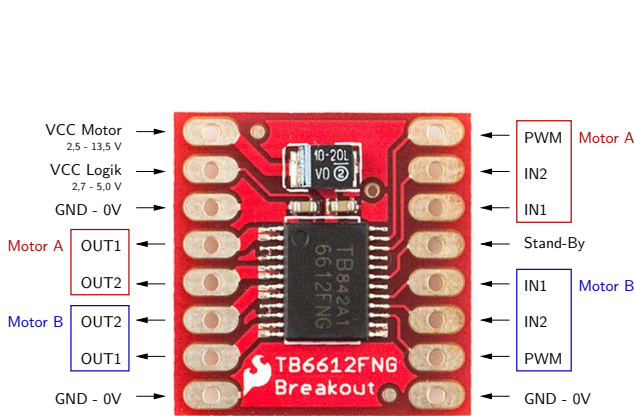


Abbildung 4: Anschlussbelegung des TB6612FNG Breakout-Boards.

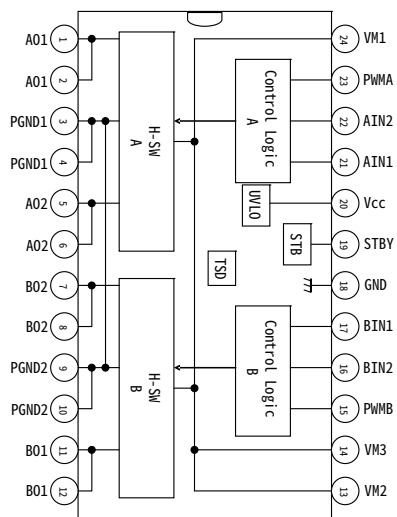


Abbildung 5: Blockdiagramm des TB6612FNG-Motortreibers