CT Praktikum: Finite State Machine (Ampel)

1 Funktion

In diesem Praktikum implementieren Sie eine Finite State Machine (FSM) als Steuerung für die Ampeln einer Strassenkreuzung. Abbildung 1 zeigt das im Praktikum verwendete Ampelmodul. Das Modul kann ans CT-Board angeschlossen werden.

Es können verschiedene Szenarios nachgestellt werden. Zuerst implementieren Sie eine einfache Steuerung mit wenigen Ampeln. Nachher können Sie sich selbst ein Szenario überlegen und umsetzen.

2 Lernziele

- Sie können eine Funktionsbeschreibung in ein einfaches 'UML State Diagram' umsetzen.
- Sie können ein 'UML State Diagram' in C implementieren.

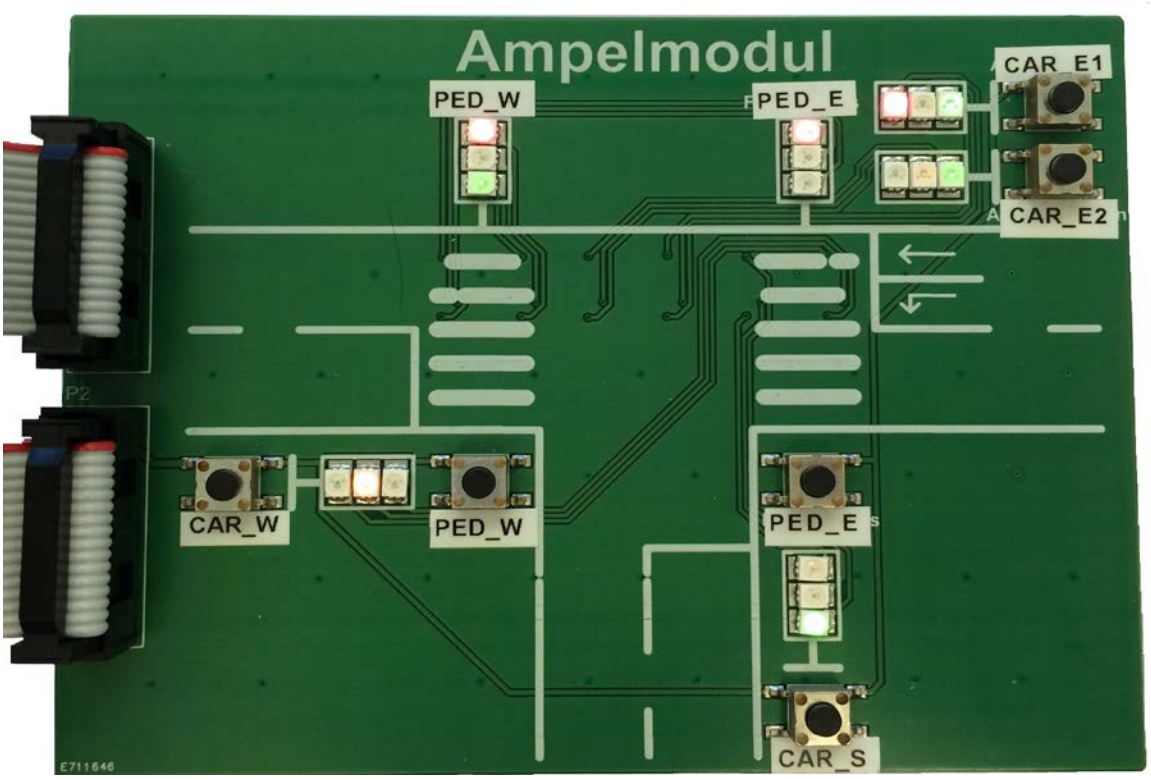


Abbildung 1: Ampelmodul

3 Schema der Schaltung

Das Ampelmodul wird über die GPIO Ports des Microcontrollers angeschlossen. Verwenden Sie Port P5 für die Ausgänge bzw. P6 für die Eingänge (siehe Abbildung 2).

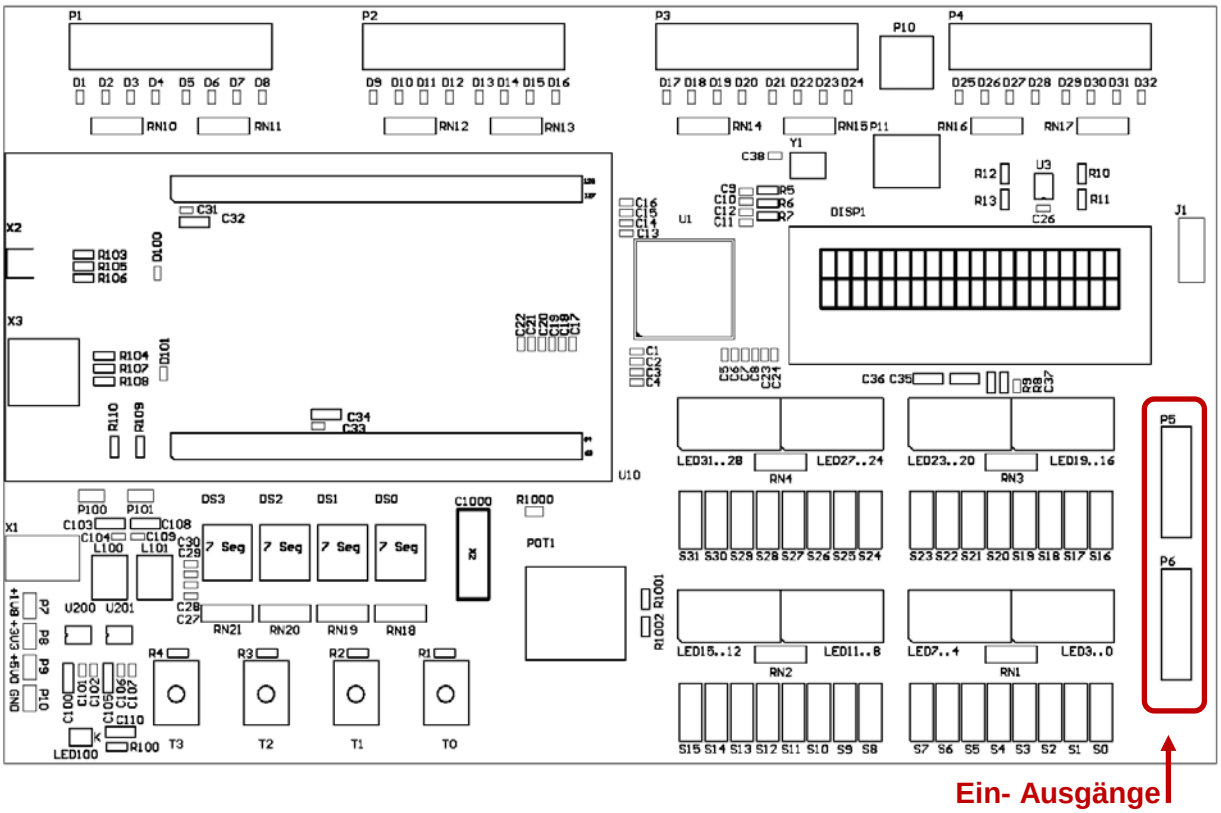


Abbildung 2: Anschluss des Ampelmoduls an CT-Board

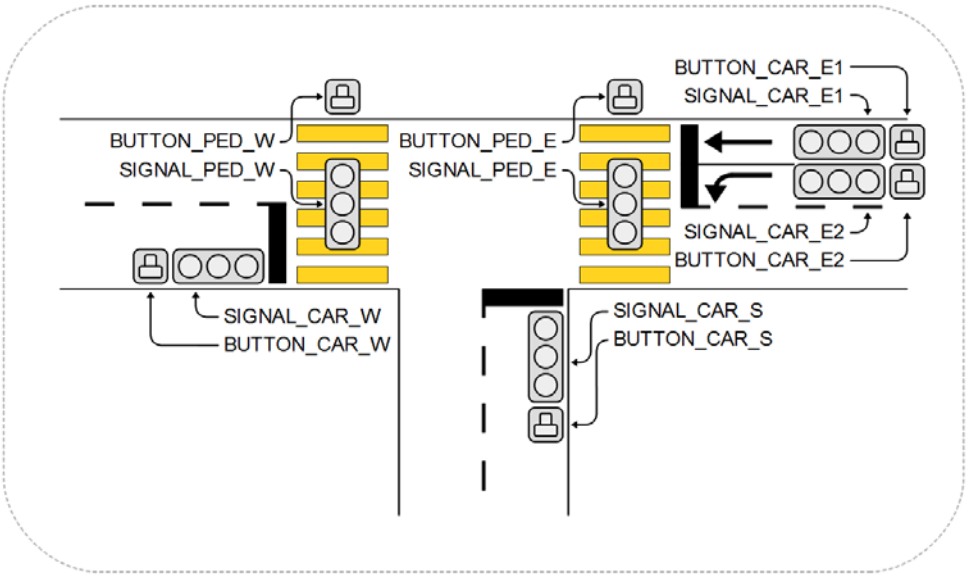


Abbildung 3: Bezeichnung der Ampelelemente

Abbildung 4 und Abbildung 5 zeigen das Schema der Ampelmoduleingänge und -ausgänge.

Die Eingänge sind „active-high“. Eine logische „**1**“ am Eingang „BUTTON_CAR_S“ (Siehe Abbildung 3) bedeutet, dass der entsprechende Taster gedrückt wurde bzw. dass ein Auto an diesem Punkt steht.

Für die Ausgänge ist unten eine Wahrheitstabelle abgebildet.

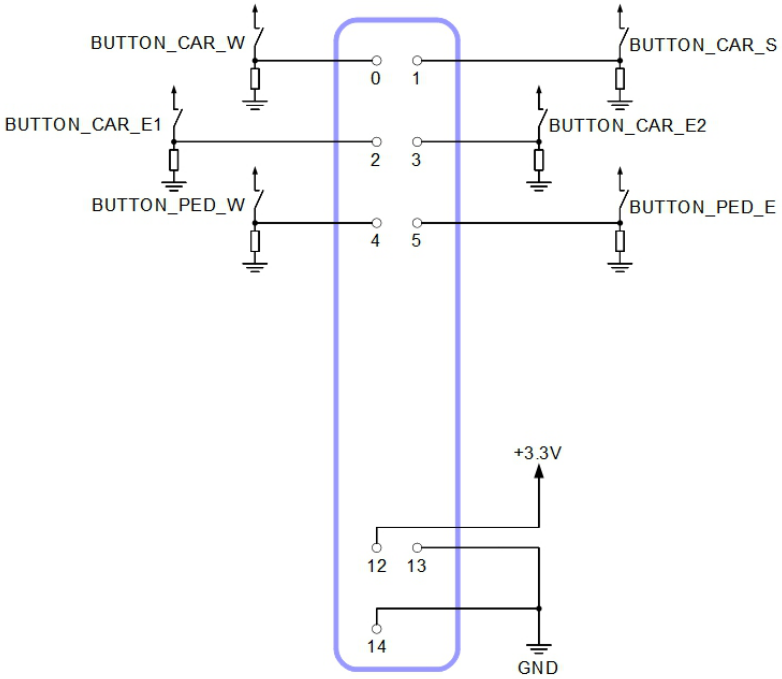
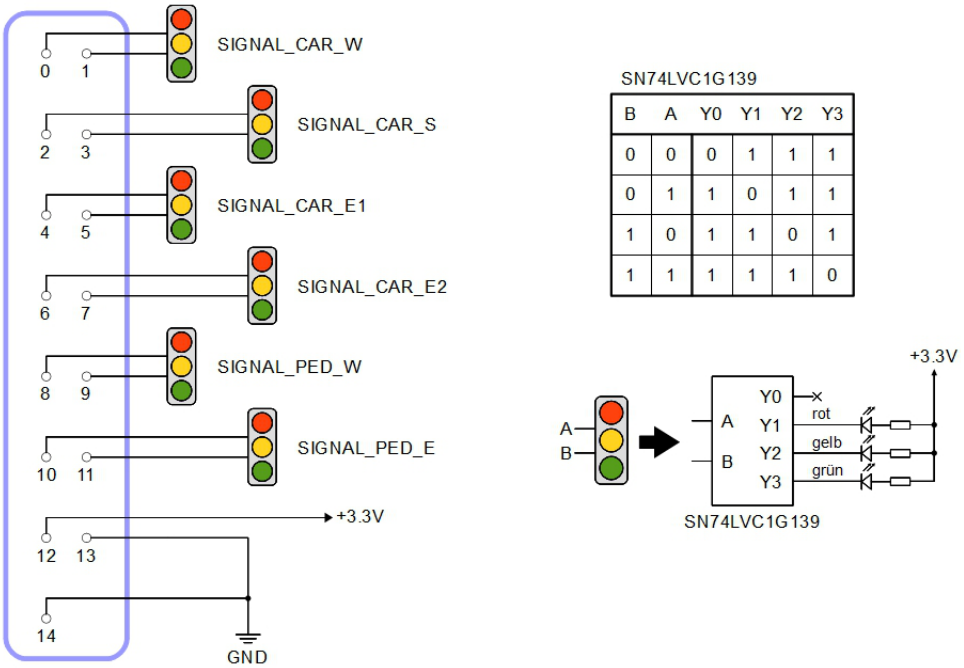


Abbildung 4: Schema der Ampelmoduleingänge



B	A	Y0	Y1	Y2	Y3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Abbildung 5: Schema der Ampelmodulausgänge

4 Aufgaben

Für die Teilaufgaben 4.1 und 4.2 wird von folgendem Szenario ausgegangen: Die Ampel soll eine gerade Strasse und zwei Fussgängerstreifen so steuern, dass jeweils entweder nur den Autos oder den Fussgängern der Weg erlaubt wird:

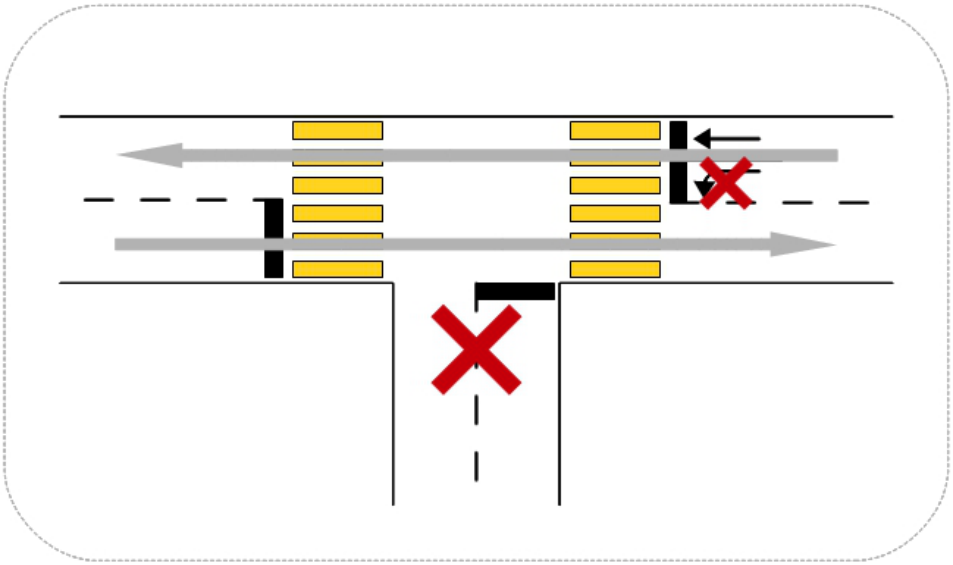


Abbildung 6: Grünphase Autos

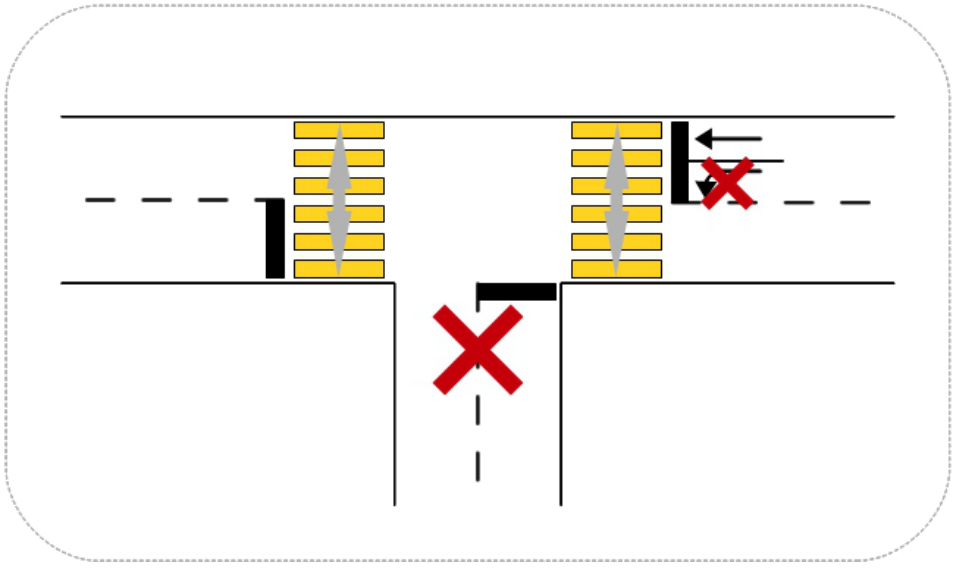


Abbildung 7: Grünphase Fussgänger

4.1 Grün- / Rotphasen

Das erste Szenario ist eine einfache Strasse mit zwei Fussgängerstreifen. Das Szenario umfasst zwei Grünphasen (siehe Abbildung 6 und Abbildung 7).

Ignorieren Sie in diesem Schritt die Gelbphasen und setzen Sie untenstehende FSM um.

- Erweitern Sie im Modul `event_handler` die Funktion `eh_get_event()` so, dass Sie erkennen können, wann eine Taste gedrückt wurde (positive Flanke).
- Erweitern Sie die Funktionen `ah_set_signal(signal_t, color_t)` im Modul `action_handler` damit Sie die entsprechenden Ampeln steuern können.
- Implementieren Sie die Funktion `fsm_handle_event(event_t)` im Modul `state_machine` gemäss dem Diagramm in Abbildung 8.

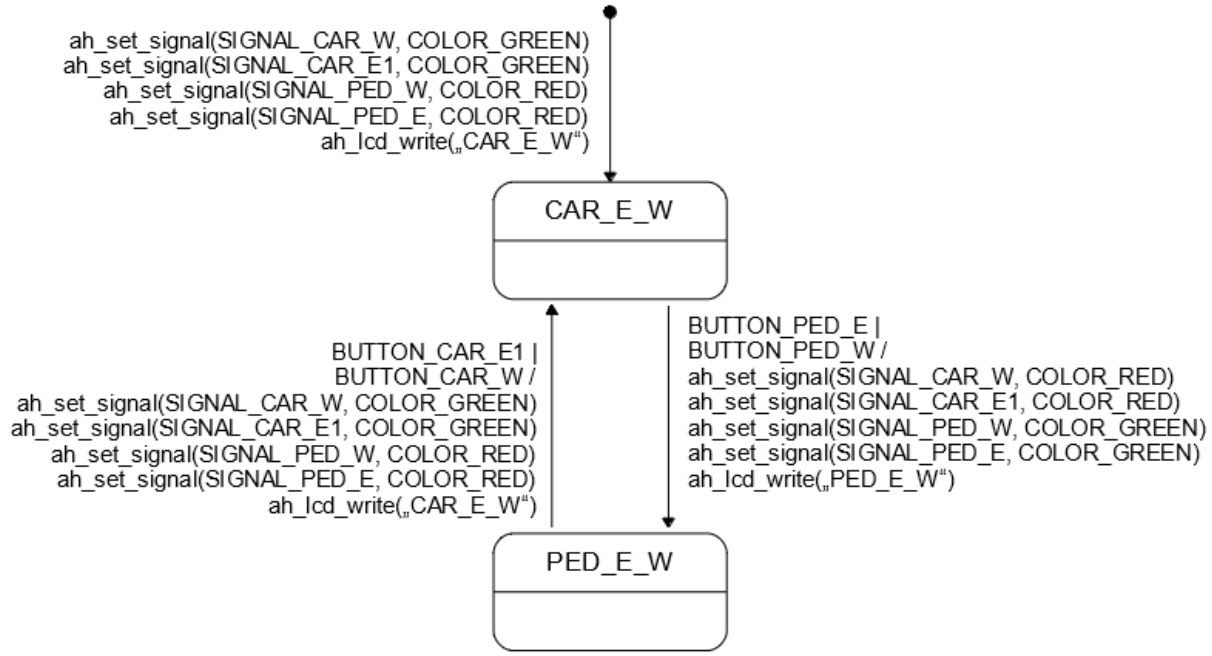


Abbildung 8: UML Diagramm ohne Gelbphase

Hinweis: Das Modul `action_handler` enthält eine Funktion `ah_lcd_write(char text[])`. Mit dieser Funktion können Sie Strings für das Debugging auf den LCD ausgeben. Zeigen Sie damit bei einer Transition den Folgezustand an.

Beispiel: `ah_lcd_write(„CLOSED“);`

4.2 Gelbphasen

Überlegen Sie sich in einem zweiten Schritt wie Sie die Steuerung aus Aufgabe 4.1 um die Gelbphase erweitern können. Die Gelbphase soll etwa 4s andauern, danach wird in die nächste Grün- / Rotphase gewechselt.

- Überlegen Sie sich, wie das UML Diagramm dieser Steuerung aussehen wird und zeichnen Sie es.
- Erweitern Sie die Funktion `eh_get_event()` im Modul `event_handler` so, dass Sie Timer-Events erkennen können. Wenn der aktuelle Wert des Timers gleich 0 ist und der vorherige Wert ungleich 0 ist, soll der Event `TIME_OUT` erzeugt werden. Der Timer Event soll dabei die höchste Priorität haben.
- Erweitern Sie die Funktion `ah_set_signal(signal_t, color_t)` im Modul `action_handler` so, dass Sie die benötigten Gelbphasen verwenden können.
- Erweitern Sie die State Machine so, dass die zusätzlichen Transitionen unterstützt werden.
- Den Timer können Sie mittels der Funktion `timer_start(duration)` starten, und mittels `timer_stop()` stoppen, falls der Timer abgebrochen werden soll.
  Der Funktionsparameter `duration` bezieht sich auf die Anzahl Interrupts eines 100 Hz Timers.

4.3 Eigenes Beispiel

Überlegen Sie sich ein eigenes Szenario. Achten Sie darauf, dass Sie keine allzu komplizierte Kreuzungssituation wählen.

- Suchen Sie die Grün- / Rotphasen in Ihrer Situation.
- Überlegen Sie sich, welche Events eine Transition in welche Grün- / Rotphase provozieren sollen.
- Ergänzen Sie Ihr Diagramm um die Gelbphasen.

Erweitern Sie ihr Programm um die fehlenden Funktionen:

- Zeichnen Sie dazu ein vollständiges 'UML State Diagram'.
- Setzen Sie dieses in C-Code um und testen Sie Ihren Code am Modell.

4.4 Bewertung

Die lauffähigen Programme müssen präsentiert werden. Die einzelnen Studierenden müssen die Lösungen und den Quellcode verstanden haben und erklären können.

Bewertungskriterien	**Gewichtung**
Sie konnten die Grün- / Rotphasen aus Aufgabe 4.1 implementieren.	1/4
Sie konnten die Gelbphasen aus Aufgabe 4.2 implementieren.	1/4
Sie haben ein eigenes Beispiel gemäss Aufgabe 4.3 umgesetzt.	2/4