Ostfalia
Hochschule für angewandte Wissenschaften

Ostfalia Fakultät Fahrzeugtechnik

Embedded Systems Laborprojekt

Grundsoftware für ein Autonomes Modellfahrzeug in Mbed

Vorgelegt von:
[redacted]

Geprüft durch:
Prof. Dr. V. von Holt

Durhcgeführt im:
WiSe 2022

Inhaltsverzeichnis

1 Begründung für die Wahl der technischen Umsetzung

Um unseren Softwareentwurf zu dokumentieren, haben wir uns dafür entschieden mehrere Diagramme zu erstellen. Ein Blockdefinitionsdiagramm beschreibt die grobe Struktur des Programmcodes. Ein Flussdiagramm für die einzelnen Tasks sowie für das Hauptprogramm beschreibt jeweils die einzelnen Programmabläufe.

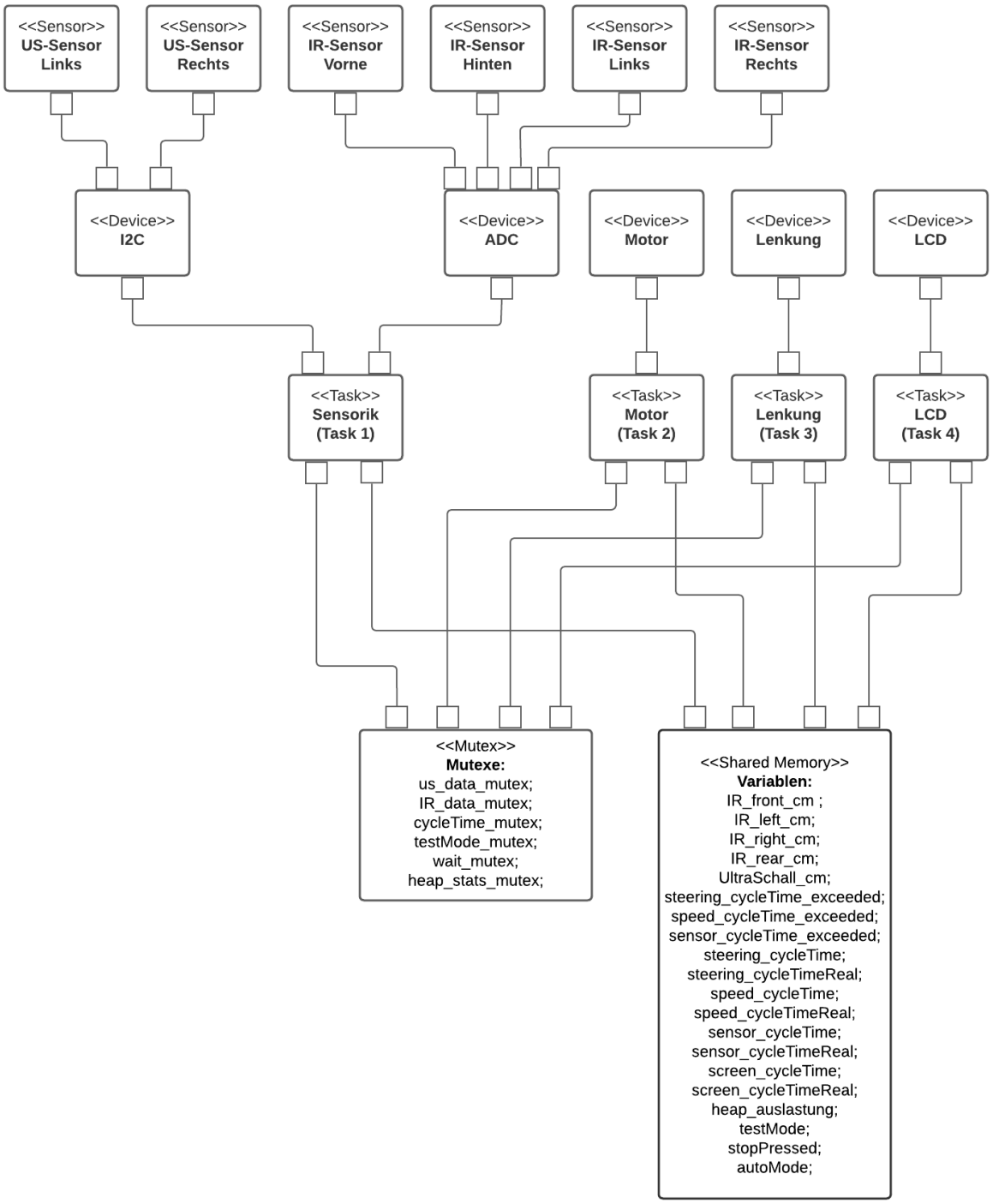
Wie das Blockdiagramm in Abschnitt 2 zeigt, haben wir vier Tasks exklusive der Haupttask bzw. "main" verwendet. Die Gliederung der Tasks erschien uns, aufgrund der vorhandenen Devices sinnvoll. Die Sensorik-Task (Task 1) kümmert sich um das Auslesen der US- und IR-Sensorik über die Devices I2C und ADC. Die Motor-Task (Task 2) kümmert sich um die Ansteuerung des Motors. Die Lenkung-Task (Task 3) kümmert sich um die Ansteuerung der Lenkung. Die LCD-Task (Task 4) kümmert sich um die Auswertung und Ansteuerung des Displays mit Touchpad. Der Informationsaustausch unter den Tasks geschieht über globale Variablen die wie in Abschnitt 2 zu erkennen, in einem globalen Speicher abgelegt sind. Um die globalen Variablen während des Zugriffes einer Task vor der Bearbeitung durch eine andere Task zu schützen, werden sie durch jeweils einem Mutex blockiert. Wir haben uns dafür entschieden verschieden Mutexe, wie in Abschnitt 2 dargestellt zu verwenden, um die Leserlichkeit des Codes zu erhöhen.

Das erste Flussdiagramm in Abschnitt 3 zeigt den Programmstart. Hier wird Übliches vorgenommen, wie die Einbindung von Headerdateien, Deklaration von Tasks und Mutexe, Initialisierung globaler Variablen, usw. Wir haben uns dafür entschieden einige Operationen über Methoden von Klassen in Headerdateien auszulagern, da wir unser Wissen aus vorangegangenen Lehrveranstaltungen einbringen wollten. Ohnehin ist es in komplexeren Projekten wichtig, den Programmcode damit einhergehend übersichtlich zu strukturieren.

Das Flussdiagramm in Abschnitt 4 befasst sich mit der Haupttask bzw. "main". Hier wird zunächst dem Motor 3,5s für die Kalibrierung Zeit gegeben. Anschließend werden die vier Tasks gestartet. Nach dem Start einer Task wir dieser jeweils 200ms Zeit für die Initialisierung gegeben. Nachdem die vier Tasks gestartet wurden geht die Haupttask in eine while(1) Schleife. Diese Schleife lässt die Haupttask stets für 1000ms schlafen, sobald sie wieder erwacht. Dies dient dazu die CPU wach zu halten.

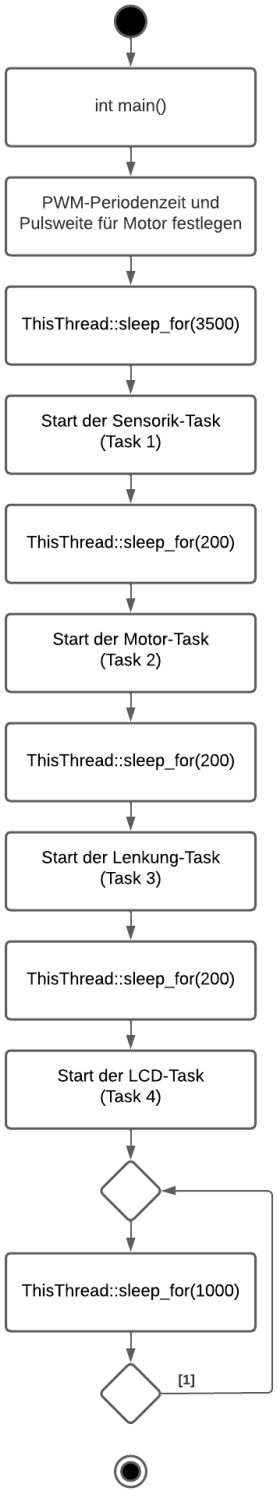
Die weiteren vier Task sind jeweils nach einem ähnlichen Prinzip untereinander aufgebaut. Zunächst werden Initialsierungen von lokalen Variablen der Tasks vorgenommen. Dann gehen die Tasks in eine while(1) Schleife in der u.A. die eigentliche Handlung der Task rotiert durchgeführt wird. Zum Ende jeder Task wird die stets gemessene Zykluszeit je Task mit der Sollzykluszeit abgeglichen. Für die berechnete Zeitdifferenz wir die Task Schlafen gelegt, sodass sie immer dieselbe Zykluszeit hat, sofern diese nicht überschritten wird.

2 Blockdiagramm

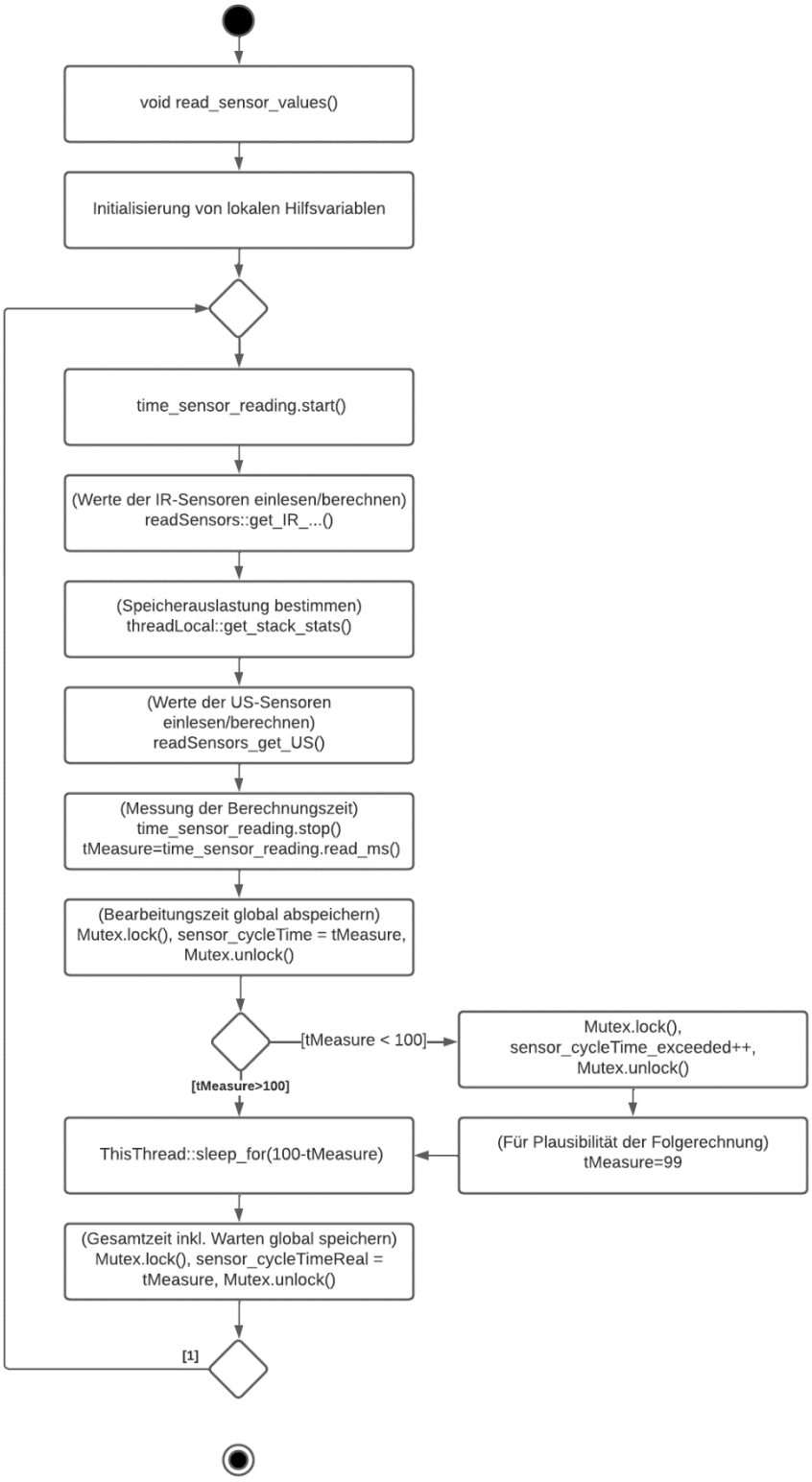


3 Flussdiagramm - Programmstart

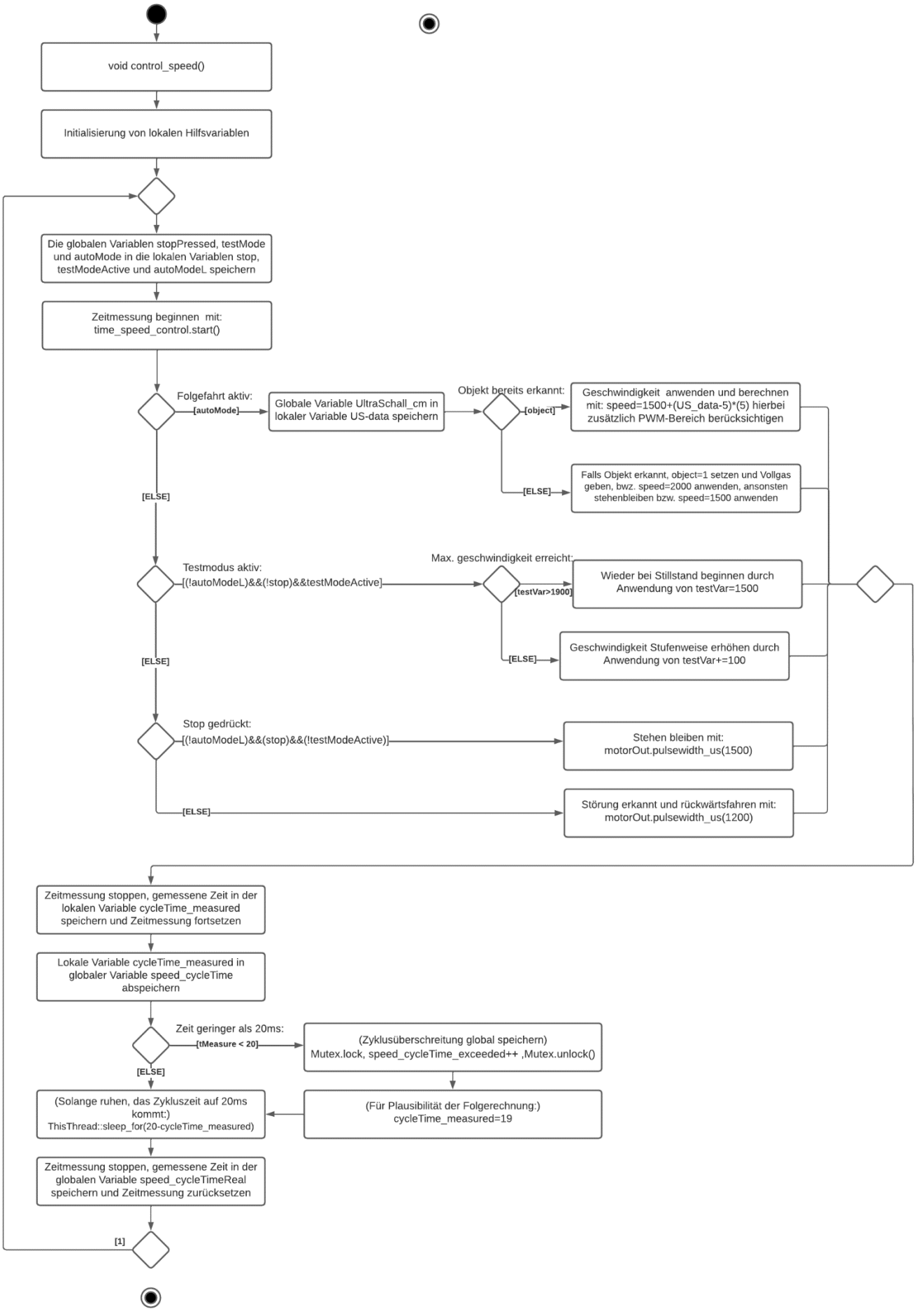
4 Flussdiagramm für Haupttask: int main()



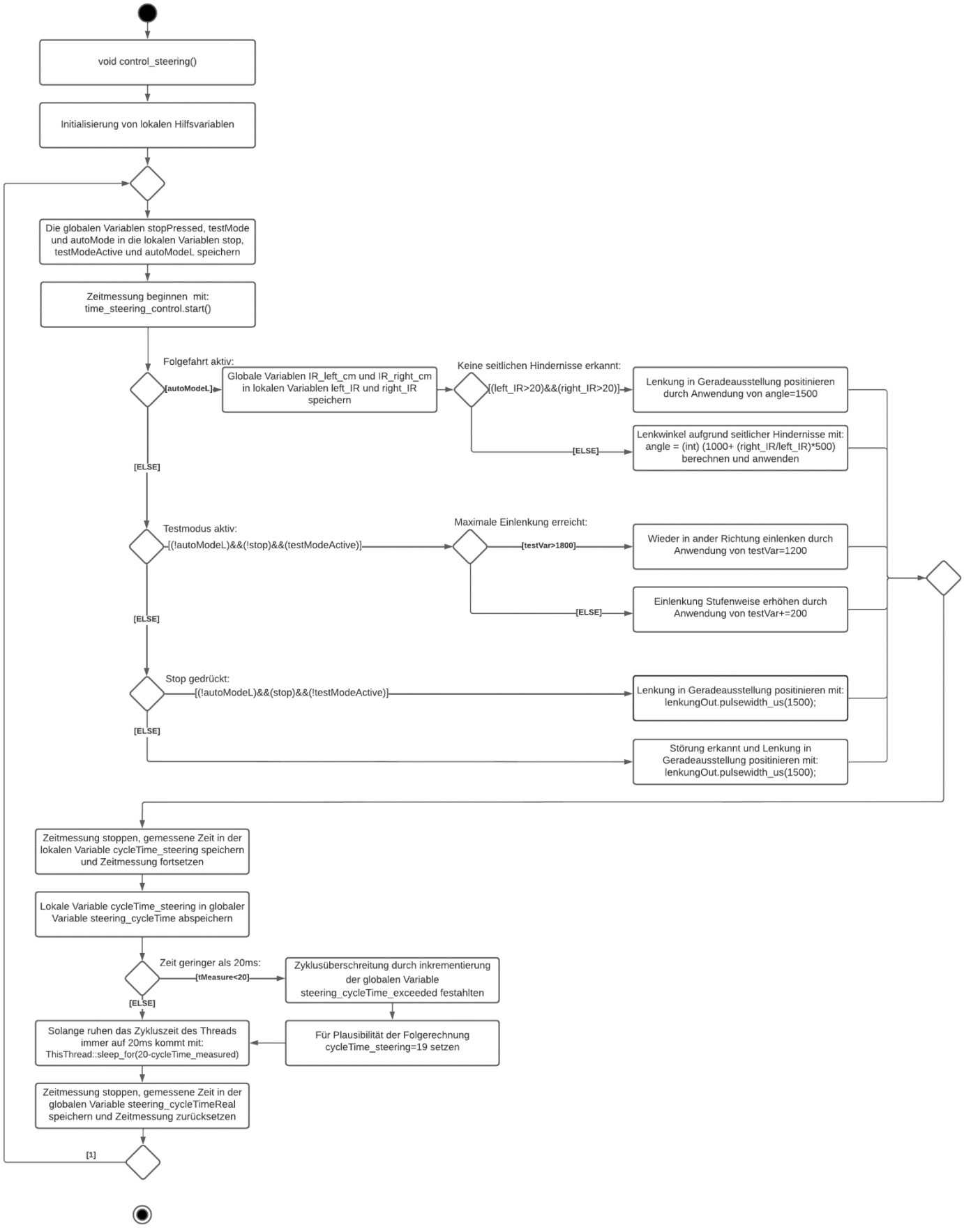
5 Flussdiagramm für Task 1: void read_sensor_values()



6 Flussdiagramm für Task 2: void control_speed()



7 Flussdiagramm für Task 3: void control_steering()



8 Flussdiagramm für Task 4: void control_screen()