Team Ausstattung - Dokumentation

Makus Denny, Maisch Lukas 04/2024

Überblick

In diesem Bericht wird das Ergebnis des Projekts im Rahmen des Moduls Mikrocomputertechnik dokumentiert, dass sich mit der Entwicklung eines steuerbaren Einkaufswagens befasst. Das Projekt wurde von verschiedenen Teams bearbeitet, wobei unser Team sich auf die Ausstattung des Wagens fokussierte. Das Ziel bestand darin, das Fahrzeug mit Blinkern und einer Hupe auszustatten.
Abbildung 1 zeigt den Stand nach Ablauf des Projektes. Unsere Lösung umfasst selbstentwickelte Blinker und einen DF-Player mit einem 16 Ω-Lautsprecher. Der Haupt-Mikrocontroller, ein STM32F407G, ist über GPIO-Pins mit einem ESP32 verbunden. Dieser ist wiederum Teil des CAN-Netzwerks des Einkaufswagens. Beide Mikrocontroller, sowie die beiden Blinker werden über externe 5V, die durch das BMS-Team zur verfügung gestellt werden, angetrieben. Sowohl für die Blinker als auch für die Hupe wurden Header und Source Dateien entwickelt, dazu unter dem Kapitel Code mehr.
Unser System kann je nach empfangenen CAN-Package folgende Aktionen ausführen: Blinken, Blinken und Fahrtanzeige, Standleuchte bzw "Bremsleuchte", Warnblinker, Hupen. Dabei kommen die Farben Grün, Orange, Weiß und Rot zum Einsatz. Es können bei Bedarf jedoch noch andere Farben dargestellt werden.

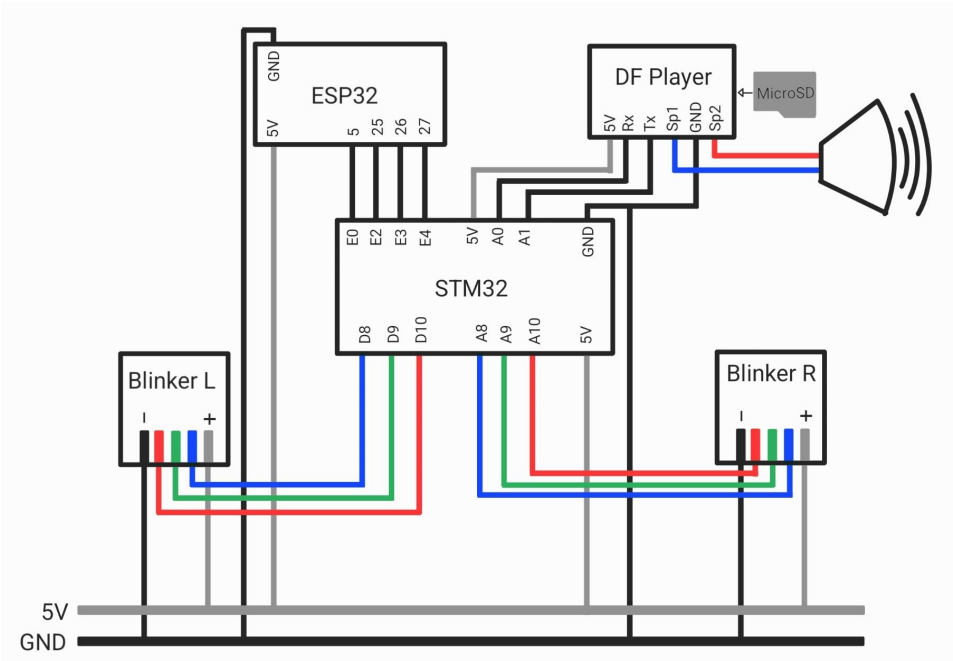


Abbildung 1: System-Schaltplan

Komponenten

Hier werden die spezifischen Komponenten beschrieben, die von unserem Team im Rahmen des Einkaufswagenprojekts entwickelt wurden. Die folgenden Abschnitte bieten eine detaillierte Betrachtung der von uns entwickelten Blinker, implementierten Hupe/Lautsprecher sowie der GPIO-CAN Schnittstelle. Eine Übersicht über alle Verwendeten Bauteile sowie gegebenenfalls deren Datenblättern ist im Abschnitt Datenblatt zu finden.

Blinker

Unsere Blinker basieren auf RGB-LEDs mit gemeinsamer Anode, sowie NPN-Transistoren. Die verwendeten Komponenten und die genutzte Schaltung wurden durch Versuche mit verschiedenen LEDs ausgewählt. Diese unterscheiden sich unter anderem in ihrem Abstrahlwinkel, der Helligkeit und dem internen Aufbau (gemeinsame Anode vs. gemeinsame Kathode). Die Blinker erfordern eine Betriebsspannung von 5V und bestehen pro Blinker aus vier RGB-LEDs.
Für jeden Blinker ist der Schaltkreis aus Abbildung 2 implementiert. Die Anode jeder RGB-LED ist mit der Versorgungspannung verbunden. Die einzelnen Kathoden jeder Farbe (Rot/Blau/Grün) aller RGB-LEDs sind über NPN-Transistoren mit dem Mikrocontroller vebunden. Die Transistoren werden über Vorwiderstände geschalten, welche zur Reduzierung des Basisstroms dienen. Zudem ist jeder LED-Kathode ein Widerstand in Reihe geschalten, um den Strom der durch die LED fließt zu begrenzen. Außerdem kann so die Leuchtkraft der einzelnen Farben eingestellt werden. Die Anschlüsse der NPN-Transistoren sind wie folgt konfiguriert: Der Collector ist mit den Vorwiderständen der Kathoden der LEDs verbunden, die Basis ist mit dem STM32 verbunden und der Emitter mit Ground verbunden.

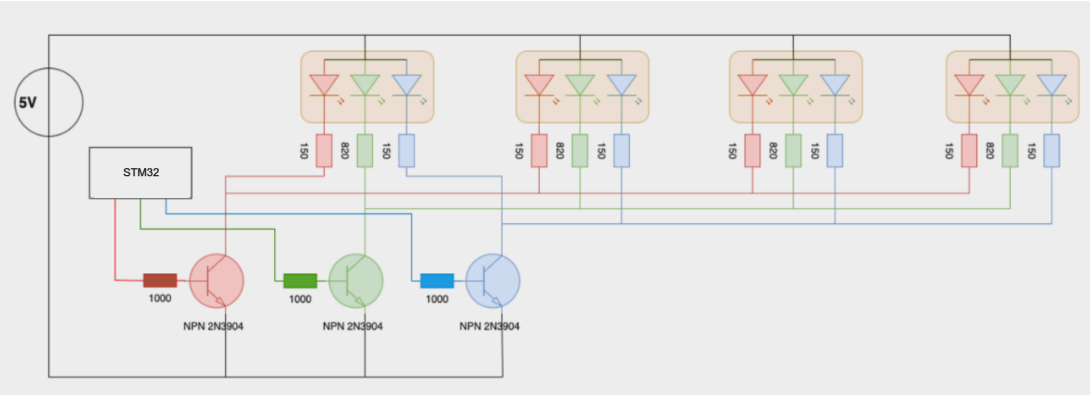


Abbildung 2: Blinker-Schaltplan

Der Schaltplan zeigt die Verschaltung der RGB-LEDs und NPN-Transistoren sowie die Anbindung an den Mikrocontroller. Die Konfiguration ermöglicht die unabhängige Steuerung der Farben für jeden Blinker. Im Kapitel Code wird erklärt, wie die LEDs Softwaretechnisch gesteuert werden können. Bei der bisherigen Konfiguration wurden jeweils die gleichen Pins und folgende Ports verwendet:
Blinkerrechts: GPIOA / Blinkerlinks: GPIOD
Rot: Pin 8 / Grün: Pin 9 / Blau: Pin 10

Lautsprecher / Hupe

Für die akustische Ausgabe in unserem autonomen Einkaufswagen wurde ein DF-Player verwendet, der über UART mit dem STM32 verbunden ist. Der DF-Player ist mit einer 32GB µSD-Karte ausgestattet, die die entsprechenden Sounddateien enthält. An den DF-Player ist ein einfacher 16 Ω-Lautsprecher angeschlossen.
Der DF-Player ist ein kompakter und benutzerfreundlicher MP3-Player. Er ermöglicht die Wiedergabe von MP3-Dateien von einer µSD-Karte und bietet eine einfache Schnittstelle zur Steuerung über UART. Der DF-Player wird über UART-Befehle vom STM32 gesteuert. Je nach gesendetem Befehl führt der DF-Player eine entsprechende Aktion aus. Beispiele für unterstützte Befehle sind: Pause, Play, Play Sound, Play Ordner on Playback usw.
Wichtig ist die korrekte Formatierung der SD-Karte im FAT32-MBR-Format sowie die Benennung der MP3-Dateien. Diese sollten gemäß den Spezifikationen des DF-Players benannt sein, wie zum Beispiel "0001.mp3". Durch die Integration des DF-Players in unser System ist es möglich beliebige MP3-Dateien über den Lautsprecher abzuspielen.
Bei der bisherigen Konfiguration wurden folgende Pins und Ports verwendet:

UART STM-Board:	**Lautsprecher:**
Port: A	Spk 1: Lautsprecher (+)
Rx: Pin 0 || Tx: Pin 1	Spk 2: Lautsprecher (-)

GPIO-CAN-Schnittstelle

Um mit dem CAN-Netzwerk des autonomen Einkaufswagens zu interagieren, wird ein zusätzlicher ESP verwendet. Dieser ESP nutzt die bereitgestellte Bibliothek von Team Schnittstellen, um CAN-Pakete zu empfangen und zu senden. Er empfängt vier verschiedene Pakettypen: Horn (Hupe), left indicator (linker Blinker), right indicator (rechter Blinker) und throttle (Gaspedal). Je nach empfangenem Paket werden entsprechende GPIO-Pins des ESP auf HIGH oder LOW gesetzt. Diese Zustände werden dann vom STM32 abgefragt und entsprechend verarbeitet.
Bei der bisherigen Konfiguration wurden folgende Pins und Ports verwendet:

STM:	**ESP:**
Port: E	Pins:
horn: Pin 4	27
throttle: Pin 3	5
left indicator: Pin 2	26
right indicator : Pin 0	25

Code

LEDs

void LED_Init(void) Initialisiert die LED-Steuerung, konfiguriert die GPIO-Pins für die LEDs und aktiviert die entsprechenden Clocks für die Ports.

void LED_On(uint8_t colour, uint8_t Blinker) Schaltet eine bestimmte Farbe bei dem angegebenen Blinker ein. Die Funktion schaltet, unabhängig vom Ausgangszustand, den Blinker in die gewünschte Farbe

void LED_Off(uint8_t colour, uint8_t Blinker) Schaltet die angegebene Farbe des Blinkers aus. Sollte eine Mischfarbe davor eingeschalten sein, bleiben die restlichen Farben am leuchten.

void LED_Toggle(uint8_t colour, uint8_t Blinker) Ändert pro Aufruf den Zustand der Farbe am gewählten Blinker (Ein/Aus).

void LED_AlternateColour(uint8_t colour1, uint8_t colour2, uint8_t Blinker) Wechselt zwischen zwei Farben eines bestimmten Blinkers bei jedem Aufruf. Bei erstem Aufruf wird colour1 geschalten.

uint8_t Colour_Status(uint8_t colour, uint8_t Blinker) Überprüft den Status der LED einer bestimmten Farbe und eines bestimmten Blinkers (Ein/Aus).

void LED_BothOn(uint8_t colour) Schaltet beide Blinker (links und rechts) in einer bestimmten Farbe ein.

void LED_BothOff(uint8_t colour) Schaltet von beiden Blinker (links und rechts) eine bestimmte Farbe aus.

Hupe

void DF_Init() Initialisiert den DFPlayer Mini für die Wiedergabe von Audiodateien von einer TF-Karte.

void Send_cmd(uint8_t cmd, uint8_t Parameter1, uint8_t Parameter2) Sendet einen Befehl an den DFPlayer Mini über UART. Werte für cmd, Parameter1 und Parameter2 sind dafür aus dem Befehlsset des DF-Players zu entnehmen.

Main

void processInputs(GPIO_PinState left_indicator, GPIO_PinState right_indicator, GPIO_-PinState driving) Die Funktion erwartet die Zustände der linken Blinker-LED, der rechten Blinker-LED und des Fahrbetriebs als Eingangsparameter. Die Zustände werden kombiniert, um den aktuellen Status der Fahrzeugbeleuchtung zu ermitteln. Für jeden Fall werden spezifische LED-Aktionen ausgeführt, wie das Einschalten oder Umschalten der LEDs in verschiedenen Farben und für verschiedene Blinker. Ein Zähler t wird verwendet, um den Zustand der Blinker über einen bestimmten Zeitraum zu überwachen und gegebenenfalls Änderungen vorzunehmen. Eine Flag z wird verwendet, um zu verhindern, dass Änderungen zu oft und zu schnell erfolgen. Es wird ein Standardfall behandelt, falls kein spezifischer Fall zutrifft oder eine Fehlerbehandlung erforderlich ist.

void checkBlinkers(GPIO_PinState blinkerLeft, GPIO_PinState blinkerRight) Die Funktion erwartet die Zustände der linken und rechten Blinker-LEDs als Eingangsparameter. Sie überprüft, ob mindestens einer der beiden Blinker eingeschaltet ist. Die entsprechenden Befehle zum Abspielen und Pausieren des Blinker-Sounds werden über die Funktion Send_cmd gesendet. Eine Flag soundPlayed wird verwendet, um zu verfolgen, ob der Sound gerade abgespielt wird oder nicht, um zu vermeiden, dass der Sound wiederholt abgespielt wird.

void callback(void) Diese Funktion wird alle 50ms von einem Timer als Callbackfunktion aufgerufen und führt die von uns entwickelten Funktionen aus. Die Funktion überprüft den Zustand des Hupensignals und speichert den aktuellen sowie vorherigen Zustand in den Variablen horn_state und pervios_pin_state. Wenn der Hupenschalter von "nicht gedrückt" auf "gedrückt" wechselt, wird ein Hupe-Sound abgespielt. Anschließend wird die Funktion checkBlinkers gefolgt von proccessInputs aufgerufen.

Datenblatt

Mikrocontroller

Eigenschaft	STM32F407VG	ESP32
Modell	STM32F407VG	ESP32
Prozessor	ARM Cortex-M4	Tensilica Xtensa LX6
Taktfrequenz	168 MHz	240 MHz (max.)
Speicher	1 MB Flash, 192 KB SRAM	512 KB Flash, 520 KB SRAM

Transistoren

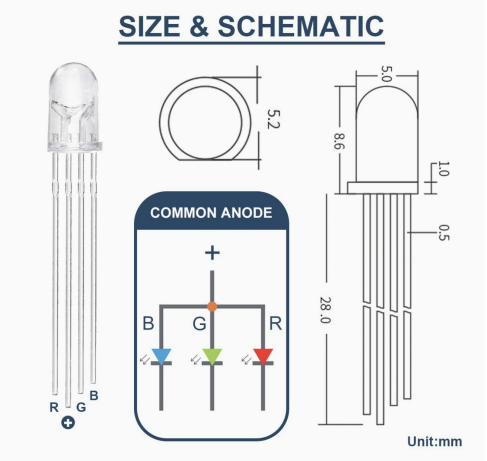
Eigenschaft	Spezifikation
Name	2N3904
Transistortyp	NPN
Format	TO-92
Max. Kollektor-Emitter-Spannung	40V
Max. Kollektorstrom	200mA

Widerstände

Schaltkreis	Wert
Rot	150 Ω
Grün	820 Ω
Blau	150 Ω
Basis	1 kΩ

LEDs

Eigenschaft	Spezifikation
Typ	RGB-LEDs
Anschluss	Common Anode
Farben	Rot, Grün, Blau
Spannungen	R: 2-2.2V G,B: 3-3.2V
Max. Strom	20mA



DF-Player

Eigenschaft	Spezifikation	Eigenschaft	Spezifikation
Modell	MP3-TF-16	Ausgang	24-bit DAC output
Versorgungsspannung	5V	Lautsprecher	16 Ω, 1W
Samplingrates	8, ... , 44.1KHz	Dokumentation	Hier