НАЦИОНАЛНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО КОМПЮТЪРНИ ТЕХНОЛОГИИ И СИСТЕМИ - гр. ПРАВЕЦ ПРИ ТУ - СОФИЯ

ДОКУМЕНТАЦИЯ НА ДИПЛОМЕН ПРОЕКТ ДЪРЖАВЕН ИЗПИТ ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ - ЧАСТ ПО ТЕОРИЯ НА ПРОФЕСИЯТА

НА ТЕМА

Подвижна камера, която качва изображения в онлайн платформа

По професия код 481020 “Системен програмист” специалност код 4810201 “Системно програмиране”

Изработил: Стефан Пламенов Пенчев 19425

Ръководител: маг. инж. Момчил Петков

СЪДЪРЖАНИЕ

Увод

Обикновено се приема, че вградените системи (embedded systems) са електронни цифрови системи, програмирани да изпълняват определени функции в реално време. В зависимост от предназначението си, се характеризират с различна сложност – от елементарни с един микроконтролерен чип, до комплексни системи, съставени от множество устройства, периферия и мрежови модули. Като съвременна тенденция при производството им се очертава стремежът към минимизиране на размерите, масата, консумираната енергия и цената им и увеличаване на надеждността и функционалността на системите.

Основната идея на една такава вградена система е да предостави функционален хардуер и да го свърже с кореспондиращ софтуер, така че да изпълним конкретна функционалност. Вградените системи намират широко приложение в областта на телекомуникациите, потребителската електроника, транспортните средства и промишлеността също така има в различни стопански отрасли.

Имайки в предвид сложността на една вградена система и комплексността на електронно устройство запознаването с възможностите на налични модули за разработка звучи прекалено сложно за един ученик или дете. От друга страна, ако една такава система е представена по интересен и разбираем начин,то тя може да заинтригува ползвателя ,така че да може да отвори врати към нови знания и умения.Като пример са количките с дистанционно управление.Те служат за развлечение както на малки деца така и на по-възрастни хора. Какъв по-добър начин за демонстрация на комбинация от код и хардуер от това да се представи проста на вид количка, която заплита съобразителност, умения за разпределение на процеси и знания за живота на една програма изградена върху вградена система.

Глава I: Литературен обзор и съществуващи решения

1.1. Вградена система

Вградена система (Embedded System) представлява компютърна система, която е вградена в устройство или система с цел да изпълнява определени функции или задачи. Тези системи се срещат в различни устройства от електрониката, механиката, автомобилостроенето, битовата техника и много други индустрии.

Ето някои от ключовите характеристики на вградените системи:

1. Оптимизирани Ресурси: Вградените системи са проектирани да работят с ограничени ресурси като памет, процесорна мощност и енергия. Те обикновено имат по-малко ресурси от стандартните компютърни системи.

2. Операционна Система: Макар че някои вградени системи могат да работят и без операционна система, повечето използват някаква форма на вградена операционна система (Embedded OS), която осигурява управление на хардуера и изпълнение на приложенията.

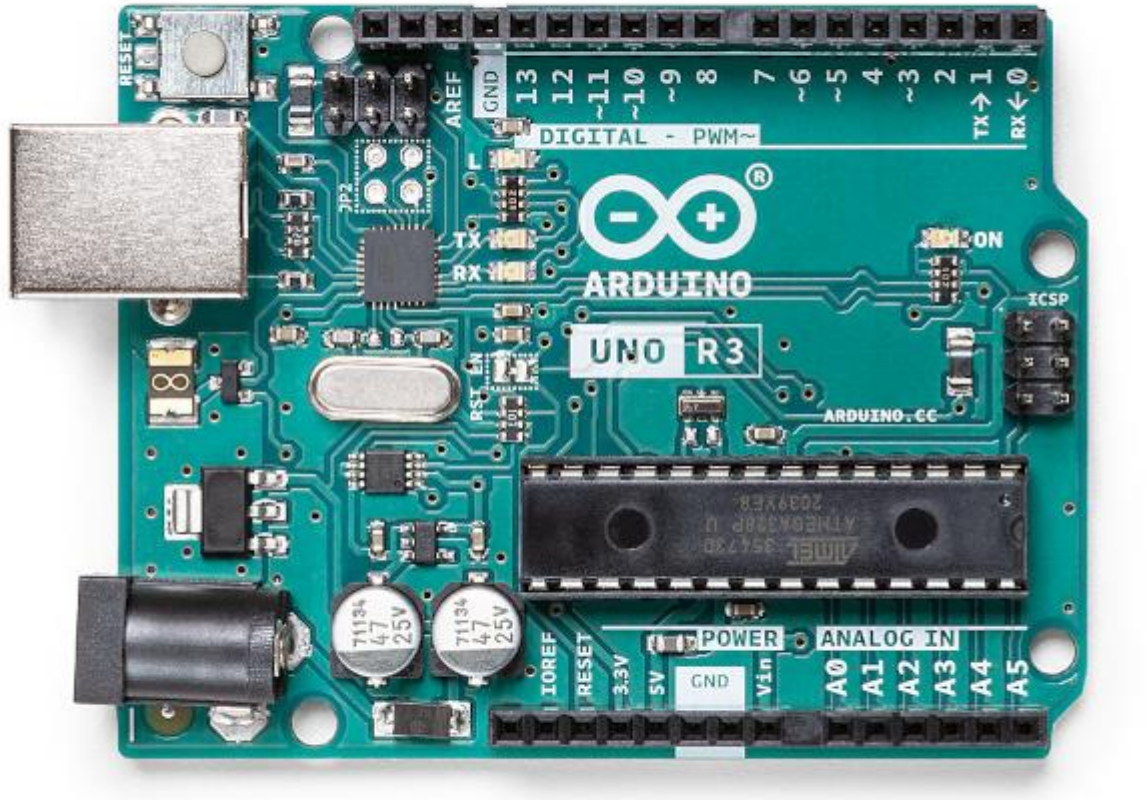
3. Специализиран Софтуер: Вградените системи често използват специализиран софтуер, който е написан за конкретните им нужди. Този софтуер може да бъде много оптимизиран и ефективен за специфичните задачи, които системата трябва да изпълнява.

4. Направени за Интеграция: Вградените системи са проектирани да бъдат лесно интегрирани в по-големи системи или устройства. Те често имат малки размери и консумират малко енергия.

5. Реално Време: Някои вградени системи се изискват да работят в реално време, където отговорът трябва да бъде предоставен в определен интервал от време. Тези системи са известни като системи в реално време (Real-Time Systems).

Примери за вградени системи включват микроконтролери като Arduino и Raspberry Pi, автомобилни компютри, системи за управление на промишлено оборудване, медицински уреди и домашни електронни устройства като телевизори, мобилни телефони и домашни

рутери. Вградените системи са важна част от съвременната технология и играят ключова роля в различни аспекти на нашия ежедневен живот.



Фигура 1.1

1.1.2. Вградени системи – как работят

Вградените системи работят по различен начин в зависимост от конкретните си приложения, изисквания и архитектура. В общи линии, вградените системи изпълняват следните основни стъпки:

1. Зареждане на Софтуер: Първата стъпка в работата на вградената система е зареждането на софтуер, който да се изпълнява. Това може да бъде вградена операционна система (Embedded OS), специализирано приложение или дори просто фърмуер, който управлява устройството.

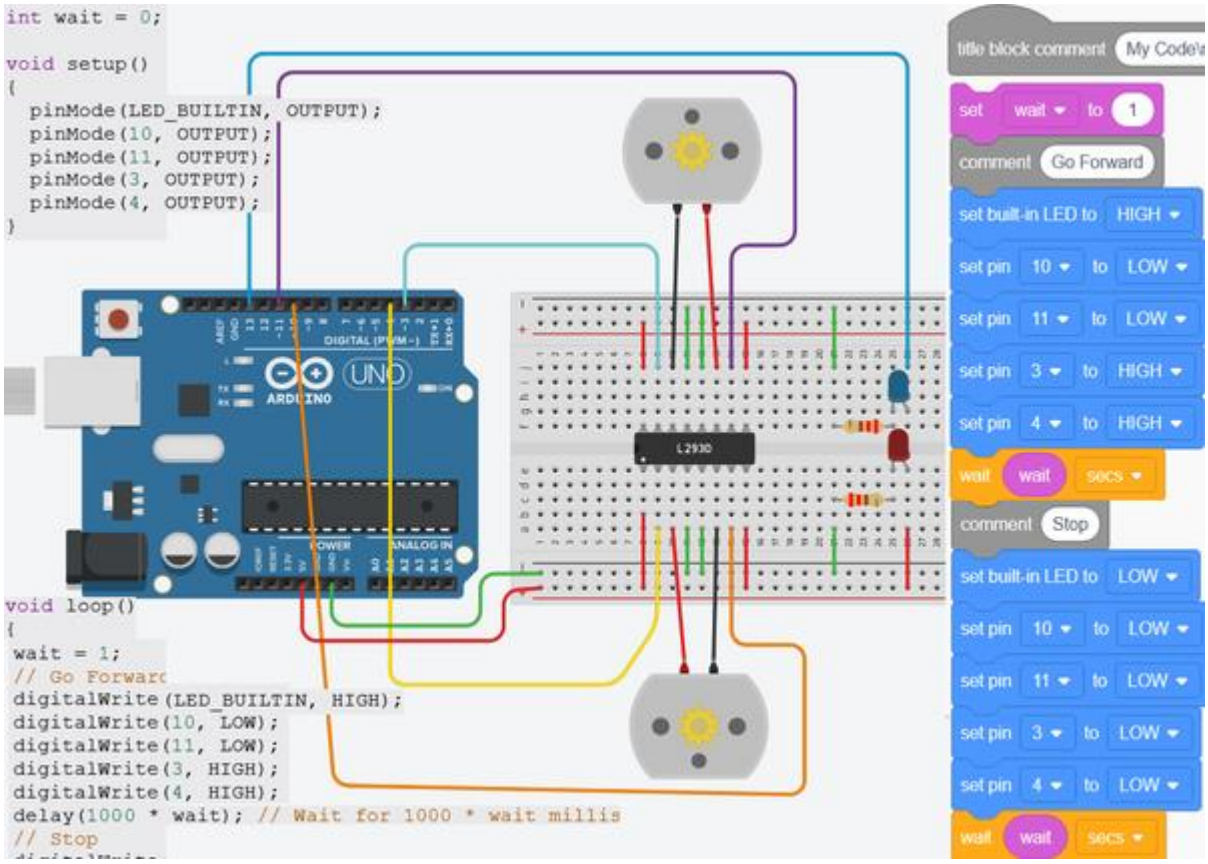
2. Изпълнение на Програмата: След като софтуерът е зареден, вградената система започва да изпълнява програмата си. Тази програма може да включва различни операции като събиране на данни от сензори, обработка на информация, управление на периферни устройства и други.

3. Взаимодействие с Периферни Устройства: Вградените системи често взаимодействат с различни периферни устройства като сензори, мотори, актуатори и комуникационни интерфейси. Те прочитат данни от тези устройства, ги обработват и изпращат обратно контролни сигнали.

4. Управление на Ресурсите: Вградените системи управляват своите ресурси като процесорно време, памет и периферни устройства. Те осигуряват достъп до ресурсите за различните части на програмата и гарантират правилното функциониране на системата.

5. Отговор на Събития: Някои вградени системи са изисквани да реагират на събития в реално време. Това може да включва бързо изпълнение на определени операции при получаване на входни сигнали или изпращане на реакция във форма на управляващи сигнали към външни устройства.

Вградените системи се проектират да бъдат ефективни, надеждни и да изпълняват своите задачи в рамките на специфични ограничения на ресурси и време. Те са от съществено значение за функционирането на множество устройства и системи в различни области като автомобилостроенето, медицината, промишлеността и други.



Фигура 1.2

1.1.3. Характеристики на вградените системи

Характеристиките на вградените системи (Embedded Systems) могат да варират в зависимост от конкретните изисквания и приложения на системата, но обикновено включват следните аспекти:

1. Ресурси:
   - Ограничени Ресурси: Вградените системи обикновено имат ограничени ресурси като процесорна мощност, оперативна памет (RAM), флаш памет (ROM), енергия и външни периферни устройства.
   - Енергийна Ефективност: Много вградени системи трябва да бъдат енергийно ефективни, особено ако са батерийно захранвани или работят на места с ограничени енергийни ресурси.

2. Хардуерни Характеристики:
   - Малък Размер: Вградените системи често трябва да бъдат малки и компактни, за да могат да бъдат интегрирани в различни устройства и системи.

- Надеждност: Поради своето предназначение за вграждане в различни устройства и околности, вградените системи трябва да бъдат надеждни и да функционират безпроблемно за дълъг период от време.
- Периферни Устройства: Вградените системи често имат вградени периферни устройства като сензори, актуатори, комуникационни интерфейси и други, които са необходими за изпълнение на конкретните задачи.

3. Софтуерни Характеристики:
- Операционна Система: Много вградени системи използват вградени операционни системи (Embedded OS) или специализиран софтуер за управление на хардуера и изпълнение на приложенията.
- Оптимизация: Софтуерът за вградените системи често е оптимизиран за конкретните хардуерни ресурси и изисквания, за да бъде ефективен и енергийно ефективен.

4. Комуникация:
- Вътрешна Комуникация: Вградените системи често трябва да взаимодействат вътрешно между различни компоненти и периферни устройства, което може да изисква различни комуникационни протоколи като UART, SPI, I2C и други.
- Външна Комуникация: Много вградени системи изискват външна комуникация с други устройства или системи, което може да включва безжични или жични мрежи, като Wi-Fi, Bluetooth, Ethernet и други.

5. Сигурност:
- Защита на Данните: Вградените системи често трябва да предоставят сигурност на данните и комуникациите си, особено ако обработват чувствителна информация или се свързват с други устройства или мрежи.

Тези характеристики са от съществено значение при проектирането, разработването и използването на вградени системи, като вземат под внимание конкретните изисквания

Фигура 1.3

1.1.4. Структура на вградените системи

Структурата на вградените системи може да варира в зависимост от конкретните изисквания и приложения на системата, но обикновено включва някои основни компоненти и слоеве. Ето обобщена структура на вградените системи:

1. Хардуерен Слой:
   - Микроконтролер или Микропроцесор: Централният компонент на вградената система, който извършва изчисления и управлява периферните устройства.
   - Периферни Устройства: Сензори, актуатори, комуникационни интерфейси, памет и други устройства, които са необходими за изпълнение на конкретните функции на системата.

2. Софтуерен Слой:

- Вградена Операционна Система (Embedded OS): Ако е необходимо, вградената система може да използва вградена операционна система, която управлява хардуера и осигурява интерфейс за приложенията.
- Приложения и Управляващ Софтуер: Софтуерът, който изпълнява конкретните функции на системата, включващ управление на периферни устройства, обработка на данни и комуникация.

3. Комуникационен Слой:
- Вътрешна Комуникация: Механизми за комуникация между компонентите на системата, като например автобуси за данни като I2C, SPI или UART.
- Външна Комуникация: Възможност за връзка с външни устройства или мрежи, като например безжични или жични мрежи за обмен на данни.

4. Захранване и Физическа Интеграция:
- Захранване: Управление на захранването на системата, осигуряване на необходимите напрежения и токове за правилната работа на хардуера.
- Физическа Интеграция: Разположение на компонентите и устройствата във физическото пространство, включително изолация, охлаждане и защита от външни влияния.

5. Сигурност и Надеждност:
- Сигурност на Данните: Защита на данните и комуникациите от неоторизиран достъп и атаки.
- Надеждност: Гарантиране на непрекъсната работа на системата и нейната стабилност дори при неблагоприятни условия или неочаквани ситуации.

Това са основните компоненти и слоеве, които обикновено се срещат в структурата на вградените системи. В зависимост от конкретните изисквания и приложения, структурата може да се различава и да включва допълнителни компоненти или слоеве.

Фигура 1.4

1.1.5. Видове вградени системи

Вградените системи могат да бъдат класифицирани в различни категории в зависимост от техните характеристики, приложения и спецификации. Ето някои от основните видове вградени системи:

1. Вградени системи в Потребителска Електроника:
   - Мобилни Телефони: Смартфоните са типичен пример за вградени системи, които включват процесор, дисплей, камера, сензори и комуникационни модули в един компактен устройство.
   - Телевизори и Аудио Уреди: Вградените системи в телевизорите и аудио уредите управляват функциите на устройствата, включително обработката на аудио и видео сигнали.

2. Вградени системи в Автомобилната Индустрия:
   - Управление на Двигателя: Вградените системи в автомобилите управляват функциите на двигателя, контролират емисиите и оптимизират разхода на гориво.
   - Безопасност и Сигурност: Системите за безопасност в автомобилите като ABS (антиблокираща система за спиране) и ESP (електронна стабилизационна програма) също се основават на вградени системи.

3. Вградени системи в Медицинската Индустрия:
   - Медицински Инструменти: Вградените системи включват медицински уреди като електрокардиографи, монитори за пулс и кръвно налягане, както и устройства за изпитвания на кръв.
   - Медицински Изображения: Системите за обработка на изображения в медицината, като например CT (компютърен томограф) и MRI (магнитно резонансно изображение), използват вградени системи за събиране и обработка на данни.

4. Вградени системи в Промишлеността:
   - Управление на Производствения Процес: Вградените системи контролират и управляват производствените процеси в различни индустрии, като например производството на автомобили, храни и фармацевтика.
   - Автоматизация на Заводи и Машини: Роботизираните системи и автоматизираните машини използват вградени системи за управление на движението, обработката на данни и контрола на операциите.

5. Вградени системи в Домашната Техника:
   - Умни Домашни Уреди: Умните термостати, осветление, камери за видеонаблюдение и други устройства са вградени системи, които се интегрират в домашната автоматизация.
   - Електронни Кухненски Уреди: Модерните кухненски уреди като хладилници, фурни и микровълнови печки включват вградени системи за управление и контрол на функциите си.

Това са само някои от основните видове вградени системи, които се срещат в различни области на нашия ежедневен живот. Всъщност, вградените системи играят важна роля в почти всеки аспект на съвременната технология и индустрия.

Фигура 1.5

1.1.6 Вградени системи в играчките

Вградените системи играят важна роля в развитието на интерактивните играчки, като им дават възможност да предложат по-забавни, образователни и иновативни функции. Ето някои от начините, по които вградените системи се използват в играчките:

1. Интерактивност и Сензори: Много съвременни играчки са оборудвани с различни видове сензори, които реагират на докосване, звук, движение или светлина. Тези сензори позволяват на играчките да реагират на действията на детето и да предлагат персонализирано и интерактивно изживяване.

2. Звукови и Светлинни Ефекти: Вградените системи позволяват на играчките да възпроизвеждат музика, звуци и светлинни ефекти, които подсилват забавлението и атмосферата на играта.

3. Образователни Играчки: Някои образователни играчки използват вградени системи, за да предложат интерактивни уроци и игри, които помагат на децата да учат цифри, букви, форми и други образователни материали.

4. Управление с Дистанционно Управление: Някои играчки, като дистанционно управляеми автомобили или дронове, използват вградени системи за да предоставят контрол и манипулация от разстояние.

5. Виртуална Реалност и Разширена Реалност: Някои играчки използват вградени системи за създаване на виртуални или разширени реалности, които пренасят децата в измислени светове и им предоставят интерактивни и вълнуващи преживявания.

6. Игри със Сървърна Връзка: Някои съвременни играчки се свързват към интернет или към специални сървъри, които позволяват на децата да играят онлайн игри и да се включват във виртуални общности.

Вградените системи играят ключова роля в еволюцията на играчките, като предоставят нови и иновативни начини за забавление, обучение и развитие на децата. Те представляват важен елемент от модерната играчка индустрия и ще продължат да се развиват и интегрират в бъдеще.

Фигура 1.6

1.2 Съществуващи решения - примери

Количките с дистанционно управление са типичен пример за вградени системи, които се използват за забавление и развлечение. Те включват различни компоненти като микроконтролер, радио приемник и предавател, мотори, сензори и други, които работят заедно, за да позволят на потребителя да управлява количката от разстояние. Ето няколко примера за колички с дистанционно:

1. RC Автомобили и Камиони: Това са модели на автомобили и камиони, които се управляват от разстояние с помощта на дистанционно устройство. Те често имитират реалните превозни средства и се използват за състезания или просто за забавление.

2. Дистанционно Управляеми Дронове: Дроновете с дистанционно управление са популярни за хоби и професионални приложения. Те могат да бъдат използвани за въздушно снимане и видеозаснемане, спортни състезания или просто за развлечение.

3. RC Бъгита и Монстър Тракове: Тези са категории колички с дистанционно, които са специално проектирани за каране извън пътя. Те имат по-големи гуми и по-мощни мотори, които им позволяват да преодоляват различни трудни терени и препятствия.

4. RC Лодки и Самолети: Лодките и самолетите с дистанционно управление са още един тип развлекателни устройства, които се използват за забавление на вода или въздух. Те могат да бъдат използвани за състезания или просто за плаване и летене.

5. Дистанционно Управляеми Танкове: Тези са модели на военни танкове, които се управляват от разстояние. Те са популярни сред хобистите и се използват за симулиране на военни битки и стратегии.

Тези са само някои от примерите за колички с дистанционно управление, които са налични на пазара. Те са изключително популярни сред хобистите и ентусиастите, които се забавляват със сглобяването, тестването и управлението им.

Фигура 1.7

Глава II: Използвани технологии

2.1 L293D Motor Driver

L293D е интегрална схема (IC) за управление на мотори, която се използва широко в различни проекти, свързани с роботика, автоматизация, моделиране на превозни средства и други подобни приложения. Тази схема е специално създадена за управление на DC мотори и биполярни стъпкови мотори. Ето основните характеристики и функции на L293D:

Предназначение и Използване:
1. Управление на Мотори: L293D се използва за контролиране на скоростта и посоката на движение на DC мотори и стъпкови мотори.
2. Биполярни Стъпкови Мотори: Специално е проектирана да управлява биполярни стъпкови мотори, които се използват често в роботика за точно движение.
3. Развитие на Проекти: L293D се използва в различни DIY (направи си сам) проекти като роботи, RC (дистанционно управление) коли, лентови принтери и други.

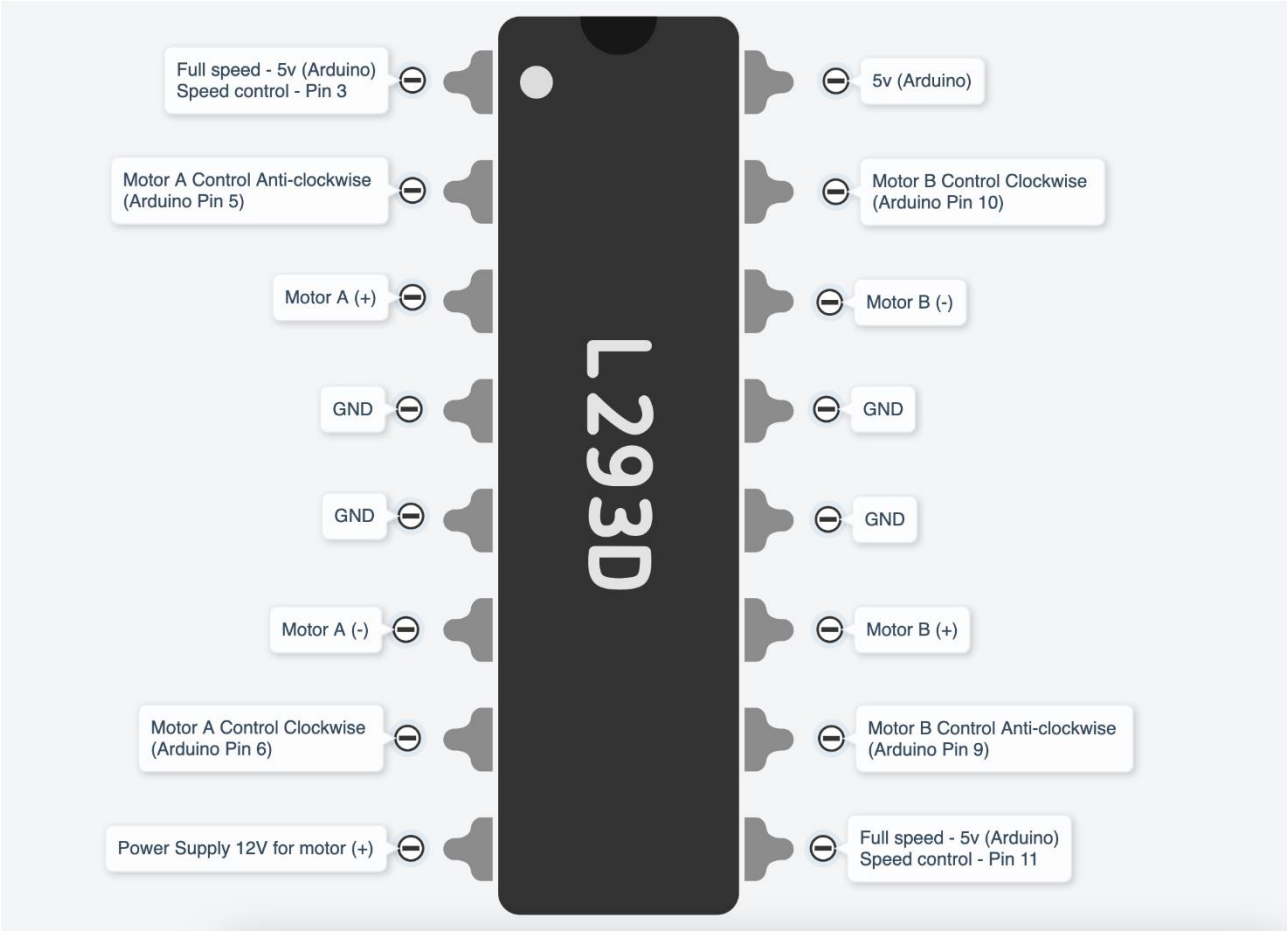
Принцип на Работа:
1. Двупосочен H-Мост: L293D включва два двупосочни H-моста, които позволяват контролирането на посоката на тока през моторите.
2. Входни Сигнали: За управление на моторите, L293D приема входни сигнали от микроконтролер или други източници, които указват посоката на движение и скоростта на моторите.
3. Защита от Претоварване: Интегрираният топлоотводник позволява на L293D да издържа на високи токове и предпазва схемата от прегряване при интензивна употреба.

4. Диоди за Защита: L293D е оборудван със защитни диоди, които предпазват схемата от обратни токове, които могат да възникнат при изключването на моторите.

Как Работи:

- Когато се подава подходящ сигнал на управляващите входове на L293D, схемата прекъсва или пропуска ток от захранването към моторите, което ги върти в желаната посока и със желаната скорост.

L293D е полезен и лесен за използване компонент, който осигурява удобно и сигурно управление на мотори в различни приложения.



Фигура 2.1

2.2 ESP 32-CAM

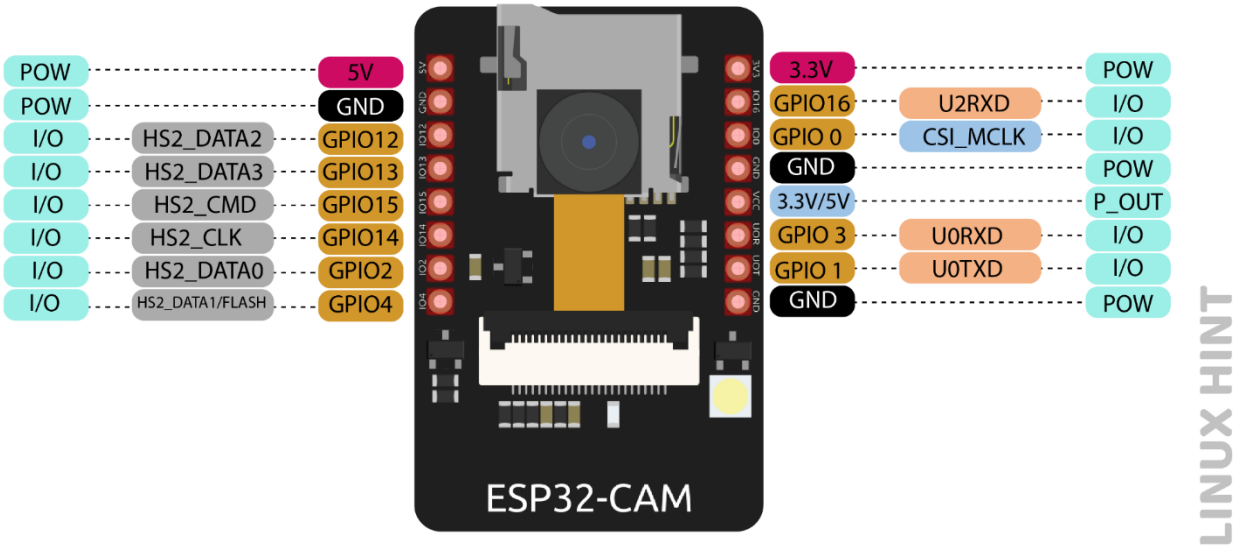
ESP32-CAM е малък модул, базиран на ESP32 микроконтролер, който интегрира камера и Wi-Fi комуникация в компактен пакет. Този модул е проектиран специално за IoT (интернет на нещата) и проекти, които изискват възможност за заснемане на изображения и видео, както и безжична комуникация.

Предназначение и Използване:
1. Видеонаблюдение: ESP32-CAM се използва за създаване на системи за видеонаблюдение, които могат да бъдат контролирани и мониторирани чрез Wi-Fi връзка.
2. IoT Проекти: Модулът се използва в различни IoT проекти, като умни домове, контрол на растения, системи за безопасност и други.
3. Машинно Виждане: Използва се за реализиране на проекти, свързани с машинно виждане, като детекция на движение, разпознаване на лица и други.

Характеристики и Възможности:
1. Камера: Интегрирана камера с възможност за заснемане на изображения и видео.
2. Мощен Микроконтролер: Базиран на ESP32, който предлага висока производителност и широки възможности за програмиране.
3. Wi-Fi Комуникация: Вграден Wi-Fi модул, който позволява безжична комуникация с други устройства и облачни услуги.
4. Малък Размер: Компактен дизайн, който прави ESP32-CAM подходящ за вграждане в различни проекти.
5. Лесна Интеграция: Поддържа различни разработъчни среди и е лесен за програмиране с Arduino IDE или други среди за разработка.

Как Работи:
- ESP32-CAM се програмира чрез Arduino IDE или други подходящи инструменти за разработка на микроконтролери.
- Посредством програмиране се задават инструкции за управление на камерата и обработка на изображенията.
- Модулът може да бъде свързан към локална Wi-Fi мрежа или да изпраща данни към облачни услуги за мониторинг и управление на данните.

ESP32-CAM е мощен инструмент за реализиране на различни IoT и проекти за машинно виждане, като предоставя възможност за създаване на устройства за видеонаблюдение, умни домове и други иновативни приложения.



Фигура 2.2

2.3 Захранване

Захранването на количка с дистанционно управление обикновено се осъществява чрез вградени батерии или презареждаеми батерии. Ето някои от ключовите аспекти на захранването на такава количка:

1. Батерии: Повечето колички с дистанционно управление използват батерии като източник на енергия. Те могат да бъдат стандартни батерии, които се заменят след изтощаване, или презареждаеми батерии, които могат да бъдат зареждани и използвани многократно.

2. Тип на батериите: В зависимост от модела и размера на количката, тя може да използва различни типове батерии. Най-често използваните са AA или AAA батерии за по-малките модели, докато по-големите модели могат да изискват по-големи и по-мощни батерии.

3. Зареждане на батериите: Ако количката използва презареждаеми батерии, те могат да се зареждат директно през вградените им зарядни устройства или чрез външни зарядни станции. Обикновено зареждането се осъществява чрез включване на количката в електрически контакт или чрез USB порт на компютър или зарядно устройство.

4. Издръжливост на батериите: Времето, за което батериите могат да поддържат работа на количката, варира в зависимост от няколко фактора, включително размера и мощността на батериите, интензивността на използването на количката и дали тя използва енергоспестяващи технологии.

5. Спестяване на енергия: Някои модели колички с дистанционно управление са оборудвани с функции за спестяване на енергия, като автоматично изключване след известен период на неактивност или режими на пестене на енергия, които ограничават консумацията на батерия при по-ниска скорост или интензивност на движение.

Захранването на количка с дистанционно управление е важен аспект от нейната функционалност и удоволствие от използването, поради което е важно да се внимава при избора и поддръжката на батериите, за да се осигури непрекъсната работа на устройството.

Фигура 2.3

2.4 HTML

HTML (HyperText Markup Language) е основният език за създаване на уеб страници. Той представлява структуриран език, който използва различни елементи и тагове за дефиниране на съдържанието и структурата на уеб страниците. HTML работи с други технологии, като CSS (Cascading Style Sheets) за стилизиране на страниците и JavaScript за добавяне на интерактивност.

Ето някои от основните аспекти на HTML в контекста на използваните технологии в проект:

1. Структура на уеб страниците: HTML се използва за дефиниране на структурата на уеб страниците. Той включва различни елементи като заглавия, параграфи, списъци, таблици, форми и други, които се използват за организиране на съдържанието на страницата.

2. Семантични елементи: HTML предлага семантични елементи, които не само предоставят структура, но и смисъл на съдържанието. Например, `<header>`, `<footer>`, `<nav>`, `<article>`, `<section>` и други елементи се използват за ясно дефиниране на различните части на страницата.

3. Форми и въвеждане на данни: HTML предоставя възможност за създаване на форми, които позволяват на потребителите да въвеждат и изпращат данни към уеб сървъра. Елементи като `<input>`, `<textarea>`, `<select>` и други се използват за създаване на различни видове полета за въвеждане.

4. Съдържание и мултимедия: HTML позволява включването на различни видове съдържание в уеб страниците, като текст, изображения, видео, аудио и други мултимедийни елементи. Тагове като `<img>`, `<video>`, `<audio>` и други се използват за вграждане на тези ресурси.

5. Свързване с CSS и JavaScript: HTML работи с CSS за стилизиране на съдържанието и JavaScript за добавяне на интерактивност. Чрез включването на външни CSS и JavaScript файлове или вграждането на стилове и скриптове директно в HTML, можем да постигнем желаните ефекти и функционалности.

6. Валидност и съвместимост: При разработката на уеб проект, е важно HTML кодът да бъде валиден и съвместим с различните уеб браузъри. Използването на стандартизирани HTML елементи и правилно създадена структура помага за постигане на тази цел.

7. Отзивчив дизайн и мобилна съвместимост: Съвременни уеб проекти често използват HTML и CSS за създаване на отзивчив дизайн, който се приспособява към различните размери на екраните, включително мобилни устройства. Това се постига с помощта на медийни заявки (media queries) и флексибилни стилове.

В общи линии, HTML е основният строителен блок на уеб страниците и е от съществено значение за разработката на функционални, естетични и достъпни уеб проекти. Съчетан с CSS и JavaScript, HTML играе ключова роля в създаването на уеб преживяване, което отговаря на потребностите на потребителите и целите на проекта.

Фигура 2.4

2.5 CSS

CSS (Cascading Style Sheets) е език за стилизиране на уеб страници. Той се използва за дефиниране на външния вид и визуалния стил на HTML елементите в уеб приложение. CSS се използва в комбинация с HTML и често с JavaScript, за да създаде динамични и атрактивни уеб страници. В проект може да се използват различни технологии, които подпомагат и разширяват възможностите на CSS:

1. CSS препроцесори: Тези инструменти, като например Sass (Syntactically Awesome Style Sheets) и Less, предлагат по-напреднали функции и синтаксис, които улесняват писането и управлението на CSS кода. Те позволяват използването на променливи, вложени правила, миксини, функции и други конструкции, които правят стиловете по-модулни и поддържаеми.

2. CSS фреймуърки: Фреймуърките като Bootstrap, Foundation и Bulma предлагат готови компоненти, стилове и грид системи, които ускоряват разработката на уеб приложения и осигуряват консистентен външен вид. Те са полезни за бърз старт на проект и за създаване на респонсивни и мобилни уеб страници.

3. CSS методологии: Тези методики, като например BEM (Block, Element, Modifier), SMACSS (Scalable and Modular Architecture for CSS) и OOCSS (Object-Oriented CSS), предлагат принципи и правила за организация на CSS кода, което улеснява мащабируемостта, поддръжката и сътрудничеството във вградения екип.

4. CSS анимации и преходи: CSS предлага възможности за създаване на анимации и преходи, които добавят интерактивност и визуален ефект на уеб страниците. Тези анимации могат да бъдат контролирани с CSS свойства като `transition`, `animation` и `transform`, както и да бъдат създадени с помощта на CSS библиотеки като Animate.css или библиотеки за анимация като GreenSock Animation Platform (GSAP).

5. CSS Grid и Flexbox: CSS Grid и Flexbox са модели за изграждане на макети, които предлагат по-гъвкаво и мощно управление на позиционирането на елементите в уеб страниците. Те позволяват създаването на сложни макети и реагиране на промени в размера на прозореца или устройството.

6. CSS променни: CSS Variables позволяват декларирането и използването на променливи в CSS кода, което улеснява повторната употреба на стилове и динамичната промяна на дизайна. Те могат да бъдат дефинирани в глобален обхват или в обхват на конкретни елементи.

7. Предпроцесори за автоматизация: Инструменти като PostCSS и Autoprefixer позволяват автоматично генериране на вендър префикси и оптимизации на CSS кода, което подобрява съвместимостта с различни браузъри и намалява времето за разработка.

В контекста на използваните технологии в проект, CSS играе ролята на език за стилизиране, който работи с HTML и JavaScript, за да създаде атрактивен, функционален и лесен за управление уеб интерфейс. Изборът на конкретни технологии зависи от нуждите на проекта, предпочитанията на екипа за разработка и изискванията за бъдещото развитие и поддръжка на уеб приложението.

Фигура 2.5

2.6 Arduino IDE

Arduino IDE (Integrated Development Environment) е софтуерна среда за програмиране, специално създадена за разработка на проекти с микроконтролери от серията Arduino. Тя предоставя интуитивен интерфейс и инструменти, които улесняват програмирането и качването на код към Arduino платформата. Ето подробностите за Arduino IDE и неговата роля в контекста на използваните технологии в проект:

1. Инсталиране и настройка:
   - Arduino IDE е безплатен софтуер, който може да бъде свален от уебсайта на Arduino и инсталиран на компютър с операционна система Windows, macOS или Linux.
   - След инсталацията, Arduino IDE се настройва за работа с конкретния модел Arduino платка, като се избира правилният тип и порт за връзка.

2. Редактор на код:
   - Arduino IDE предоставя текстов редактор, където програмистът може да пише и редактира програмния код.
   - Редакторът осигурява основни функции като подчертаване на синтаксиса, автоматично довършване и номерация на редовете, което улеснява програмирането.

3. Библиотеки и примери:
   - Arduino IDE включва богата библиотека с готови функции и примери за различни сензори, актуатори и комуникационни модули.
   - Тези библиотеки са от основно значение за бързото и лесно разработване на проекти, като се осигурява готова реализация на много от необходимите функционалности.

4. Компилация и качване на код:
   - След като програмистът завърши писането на кода, Arduino IDE компилира програмата в машинен код, който може да бъде изпълнен от микроконтролера.
   - Компилираният код се качва (upload) към Arduino платката чрез USB връзка или друго подходящо средство за комуникация.

5. Монитор на сериен порт:

- Arduino IDE предоставя интегриран монитор на сериен порт, който позволява на програмиста да изпраща и получава данни от Arduino платката по време на работа.

- Това е полезен инструмент за отстраняване на грешки (debugging) и за проследяване на работата на устройството.

6. Интеграция със среди за разработка:

- Arduino IDE може да се интегрира с други софтуерни среди за разработка (IDE), като Visual Studio Code, като се добавят допълнителни разширения или конфигурации.

В контекста на използваните технологии в проект, Arduino IDE е ключово средство за програмиране и управление на Arduino микроконтролерите, които често се използват за управление на сензори, актуатори и други периферни устройства в различни проекти, включително IoT (Интернет на нещата), роботика, автоматизация и други.

Фигура 2.6

Глава III: Проектиране и разработка

3.1 Свързване на L293D

Характеристики на L293D: Преди да започнете, убедете се, че сте запознати с характеристиките на L293D, включително максималните стойности на тока, входното напрежение и т.н.

Изберете изходите за вашия двигател: L293D има четири изхода (1, 2, 3 и 4), като всеки от тях може да управлява по един двигател. Всяка от тези изходи има възможност да се управлява в различни посоки.

Свържете входните сигнали: L293D има контролни пинове, които приемат сигнали от микроконтролер или друг източник на сигнали. Тези пинове обикновено са означени като EN1, EN2, IN1, IN2, IN3 и IN4.

Свържете двигателя: Свържете двигателя към изходите на L293D. Уверете се, че сте свързали правилно положителния и отрицателния краища на двигателя към подходящите терминали на моста.

Захранване на моста: L293D изисква външно захранване за работа. Свържете подходящо захранване към пиновете VCC1 и VCC2 на моста. Уверете се, че захранването е в съответствие с характеристиките на L293D.

Свързване на земята: Не забравяйте да свържете земята на вашата система към терминала GND на L293D, за да се осигури правилна работа на схемата.

Тестване и отстраняване на неизправности: След като всичко е свързано правилно, тествайте вашата схема и проверете дали всичко работи коректно. Ако има проблеми, прегледайте връзките и уверете се, че всичко е свързано правилно.

3.2 Свързване на ESP 32-CAM

Захранване: ESP32-CAM се захранва чрез входовете за захранване VCC и GND. Уверете се, че предоставяното напрежение е в съответствие със спецификациите на модула.

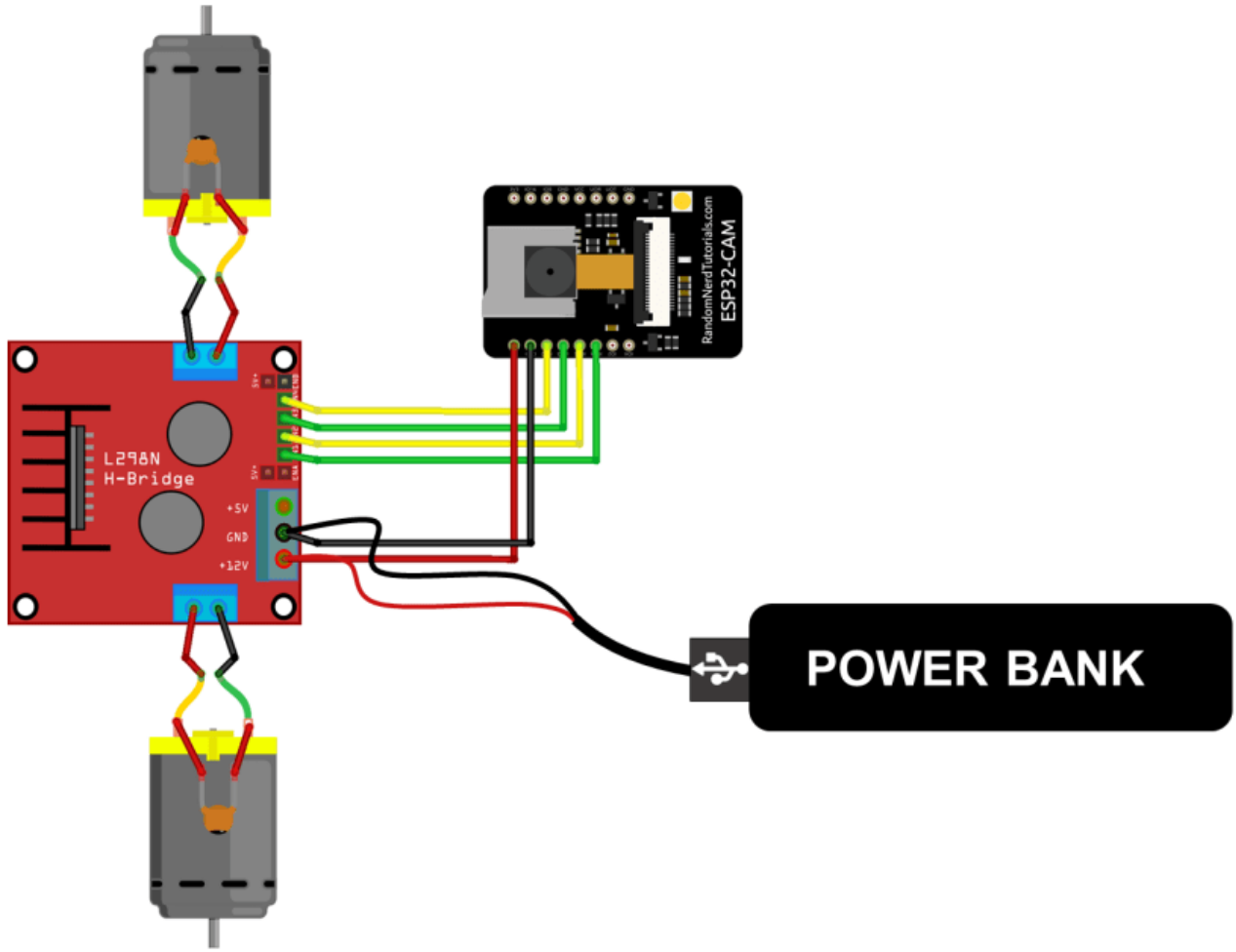
Свързване на камерата: Камерата е вградена в модула и обикновено се свързва вече. Ако не е така, свържете кабела на камерата към съответните пинове на ESP32-CAM модула.

Свързване на сериен порт: Ако искате да използвате сериен комуникационен порт за отстраняване на проблеми или за връзка с компютър, свържете TX и RX пиновете на ESP32-CAM към съответните пинове на вашето устройство.

Свързване на други периферни устройства: Ако желаете да свържете други периферни устройства към ESP32-CAM, използвайте свободните GPIO пинове на модула. Уверете се, че конфигурирате GPIO пиновете правилно в софтуера си.

Програмиране и контрол: Използвайте средата за програмиране, която предпочитате (например Arduino IDE) и програмирайте ESP32-CAM с вашите желани функционалности за управление на камерата и комуникация с други устройства.

Тестване и отстраняване на проблеми: След като всичко е свързано и програмирано, тествайте функционалността на вашата система и проверете за възможни проблеми. Ако се сблъскате с проблеми, прегледайте връзките и програмния код, за да ги отстраните.



Фигура 3.1 Схема на цялата система

L298N Motor Driver	ESP32-CAM
IN1	GPIO 14
IN2	GPIO 15
IN3	GPIO 13
IN4	GPIO 12

Фигура 3.2 Връзка на пиновете между L293D и ESP32-CAM

3.3 Създаване и дизайн на Web Server

Избор на технологии: Изберете технологиите, които ще използвате за създаване на уеб сървър. Някои от най-популярните технологии включват Node.js със Express, Python с Flask или Django, Ruby с Ruby on Rails и други.

Инсталация и настройка на средата за разработка: Инсталирайте средата за разработка за избраната технология и настройте проекта си. Това включва инсталиране на съответните библиотеки и инструменти за разработка.

Създаване на основен уеб сървър: Създайте основен уеб сървър, който слуша за заявки от клиентите и връща отговори. В този етап обикновено се създава файл с основния код на сървъра.

Дефиниране на маршрути: Дефинирайте маршрути, които сървърът ви ще поддържа. Тези маршрути са свързани с определени URL адреси и обработват заявките на клиентите за тези адреси.

Изграждане на функционалността на сървъра: Добавете логика и функционалност към вашия сървър. Това може да включва работа с бази данни, обработка на форми, управление на сесии и други.

Създаване на потребителски интерфейс: Създайте HTML, CSS и JavaScript файлове за дизайна на вашата уеб страница. Тези файлове ще бъдат заредени от браузъра на клиента и ще осигурят интерактивност и визуален дизайн на приложението ви.

Интегриране на потребителския интерфейс със сървъра: Интегрирайте потребителския интерфейс с вашия уеб сървър. Това включва зареждане на статични файлове (HTML, CSS, JavaScript) и обработка на заявки за динамично съдържание.

Тестване и отстраняване на грешки: Тествайте вашето приложение, за да се уверите, че всичко работи правилно. Отстранете всякакви грешки и проблеми, които откриете по време на тестовете.

Разгласяване и публикуване: Когато вашето приложение е готово за продукция, разгласете го на уеб сървъра си и го направете достъпно за потребителите.

```
static const char* _STREAM_CONTENT_TYPE = "multipart/x-mixed-replace;boundary=" PART_BOUNDARY;
static const char* _STREAM_BOUNDARY = "\r\n--" PART_BOUNDARY "\r\n";
static const char* _STREAM_PART = "Content-Type: image/jpeg\r\nContent-Length: %u\r\n\r\n";

httpd_handle_t camera_httpd = NULL;
httpd_handle_t stream_httpd = NULL;

static const char PROGMEM INDEX_HTML[] = R"rawliteral(
<html>
  <head>
    <title>ESP32-CAM Robot</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style>
      body { font-family: Arial; text-align: center; margin:0px auto; padding-top: 30px;}
      table { margin-left: auto; margin-right: auto; }
      td { padding: 8 px; }
      .button {
        background-color: #2f4468;
        border: none;
        color: white;
        padding: 10px 20px;
        text-align: center;
        text-decoration: none;
        display: inline-block;
        font-size: 18px;
        margin: 6px 3px;
        cursor: pointer;
        -webkit-touch-callout: none;
        -webkit-user-select: none;
        -khtml-user-select: none;
        -moz-user-select: none;
        -ms-user-select: none;
        user-select: none;
        -webkit-tap-highlight-color: rgba(0,0,0,0);
      }
      img {  width: auto ;
        max-width: 100% ;
        height: auto ;
      }
    </style>
  </head>
  <body>
    <h1>ESP32-CAM Robot</h1>
    <img src="" id="photo" >
    <table>
      <tr><td colspan="3" align="center"><button class="button" onmousedown="toggleCheckbox('forward');" ontouchstart="toggleCheckbox('forward');" onmouseup="toggleCheckbox
      ('stop');" ontouchend="toggleCheckbox('stop');">Forward</button></td></tr>
      <tr><td align="center"><button class="button" onmousedown="toggleCheckbox('left');" ontouchstart="toggleCheckbox('left');" onmouseup="toggleCheckbox('stop');"
      ontouchend="toggleCheckbox('stop');">Left</button></td><td align="center"><button class="button" onmousedown="toggleCheckbox('stop');" ontouchstart="toggleCheckbox
      ('stop');">Stop</button></td><td align="center"><button class="button" onmousedown="toggleCheckbox('right');" ontouchstart="toggleCheckbox('right');"
      onmouseup="toggleCheckbox('stop');" ontouchend="toggleCheckbox('stop');">Right</button></td></tr>
      <tr><td colspan="3" align="center"><button class="button" onmousedown="toggleCheckbox('backward');" ontouchstart="toggleCheckbox('backward');"
      onmouseup="toggleCheckbox('stop');" ontouchend="toggleCheckbox('stop');">Backward</button></td></tr>
    </table>
   <script>
   function toggleCheckbox(x) {
     var xhr = new XMLHttpRequest();
     xhr.open("GET", "/action?go=" + x, true);
     xhr.send();
   }
   window.onload = document.getElementById("photo").src = window.location.href.slice(0, -1) + ":81/stream";
  </script>
  </body>
</html>
)rawliteral";
```

Фигура 3.3 Това тук е кода за html страницата в ArduinoIDE, който съдържа бутоните за контрол на количката и поле, на което се показва видеото, заснето от нея.

3.4 Конфигурация на пиновете и заявките

```
#define MOTOR_1_PIN_1 14
#define MOTOR_1_PIN_2 15
#define MOTOR_2_PIN_1 13
#define MOTOR_2_PIN_2 12
```

Фигура 3.4 Тук се дефинират пиновете на L293D в ArduinoIDE

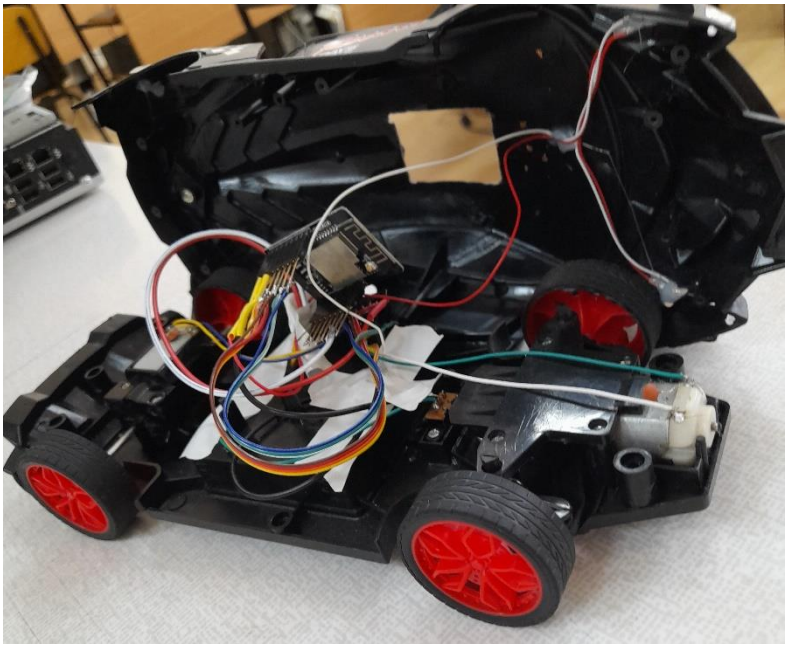
```
if(!strcmp(variable, "forward")) {
    Serial.println("Forward");
    digitalWrite(MOTOR_1_PIN_1, 1);
    digitalWrite(MOTOR_1_PIN_2, 0);
    digitalWrite(MOTOR_2_PIN_1, 1);
    digitalWrite(MOTOR_2_PIN_2, 0);
 }
 else if(!strcmp(variable, "left")) {
   Serial.println("Left");
   digitalWrite(MOTOR_1_PIN_1, 0);
   digitalWrite(MOTOR_1_PIN_2, 1);
   digitalWrite(MOTOR_2_PIN_1, 1);
   digitalWrite(MOTOR_2_PIN_2, 0);
}
else if(!strcmp(variable, "right")) {
   Serial.println("Right");
   digitalWrite(MOTOR_1_PIN_1, 1);
   digitalWrite(MOTOR_1_PIN_2, 0);
   digitalWrite(MOTOR_2_PIN_1, 0);
   digitalWrite(MOTOR_2_PIN_2, 1);
}
else if(!strcmp(variable, "backward")) {
   Serial.println("Backward");
   digitalWrite(MOTOR_1_PIN_1, 0);
   digitalWrite(MOTOR_1_PIN_2, 1);
   digitalWrite(MOTOR_2_PIN_1, 0);
   digitalWrite(MOTOR_2_PIN_2, 1);
}
else if(!strcmp(variable, "stop")) {
   Serial.println("Stop");
   digitalWrite(MOTOR_1_PIN_1, 0);
   digitalWrite(MOTOR_1_PIN_2, 0);
   digitalWrite(MOTOR_2_PIN_1, 0);
   digitalWrite(MOTOR_2_PIN_2, 0);
}
```

Фигура 3.5 Тук са заявките за контрол на количката написани в ArduinoIDE, те се състоят от група else if statements и позволяват да се задвижи в зададената посока.

3.5 Реализация

За реализацията на проекта използваме количка Lamborghini, която оригинално се управлява с дистанционно с антена, но сега се управлява с Wi-Fi. По-точно се управлява като се изпращат сигнали към L293D Motor Driver през заявките показани в **Фигура 3.5**.

Фигура 3.6



Фигура 3.7 Снимка на количката и начина, по който са свързани ESP32, L293D и захранването.

Заключение

Вградените системи играят ключова роля в различни области на живота, като промишлеността, медицината, транспорта и дори в домашното околие. Използвайки напреднали технологии като микроконтролери и сензори, вградените системи могат да предложат автоматизирани и интелигентни функции, които оптимизират процесите и подобряват управлението.

Количката с камера, управлявана дистанционно, представлява иновативно приложение на вградените системи в областта на транспорта и домашното обслужване. Чрез интегрирането на камера и микроконтролер, количката може да предложи следните възможности:

Визуално Наблюдение: Камерата позволява на оператора да вижда околната среда от разстояние и да наблюдава процесите, които се извършват около количката.

Дистанционно Управление: Чрез микроконтролера и безжичната комуникация, операторът може да управлява движението на количката от разстояние, без да е необходимо да бъде на място.

Автоматизирани Функции: Вградените системи могат да бъдат програмирани да извършват различни автоматизирани действия, като например избягване на препятствия, следване на линии или дори самостоятелно навигиране в пространството.

Данни и Анализ: Количката може да събира данни от околната среда с помощта на сензори или камерата и да ги предоставя на оператора за анализ и вземане на решения.

Заключително, количката с камера, управлявана дистанционно, е пример за интелигентно използване на вградените системи за оптимизиране на процесите и подобряване на ефективността в различни области на приложение. Тази технология предлага значителен потенциал за автоматизация и разширяване на възможностите за контрол и мониторинг в реално време.

Използвана литература

1. https://randomnerdtutorials.com/esp32-cam-car-robot-web-server/
2. https://en.wikipedia.org/wiki/Arduino
3. https://en.wikipedia.org/wiki/ESP32
4. https://wiki.eprolabs.com/index.php?title=L293D_H-bridge_Motor_Driver_Shield
5. https://www.arduino.cc/en/software/
6. https://www.w3schools.com/html
7. https://randomnerdtutorials.com/esp32-web-server-arduino-ide/
8. How to Program / Upload Code to ESP32-CAM AI-Thinker (Arduino IDE) | Random Nerd Tutorials
9. https://www.tinkercad.com/
10. https://www.instructables.com/Rebuild-an-RC-Car-With-ESP32-Arduino-Code/