Техническое задание на разработку прошивки для микроконтроллера ESP-32

Цели разработки:

Разработать прошивку для микроконтроллера ESP-32, выполняющую функцию дозирующего растворы устройства, посредством управления через step dir драйвер 8-ю шаговыми двигателями. Устройство будет принимать команды через TCP для управления моторами, в команде будет приходить скорость в шагах в секунду, и количество шагов.

Детали реализации:

Настройка подключения к wi-fi сети

При первом старте устройство запускается в режиме wi-fi точки доступа, и запускает веб интерфейс настройки подключения(captive portal), в котором на данный момент будут поля для ввода SSID, пароля, Ip адреса, адрес шлюза, маски подсети. При этом вэб интерфейс должен запускаться именно как captive portal, т.е при подключении к точке доступа страница откроется автоматически.

Настройки подключения должны храниться в eeprom, также на данный момент настройки ip адреса будут статическими, но необходимо заложить переход на динамический адрес, отслеживаться адреса устройств в будущем будут через бд на стороне одноплатного компьютера..

Работа с tcp

Устройство будет принимать команды и отправлять уведомления через tcp, список команд и уведомлений, прикреплен в таблице к данному документу. Команды и уведомления представляют из себя данные в формате json, путь, будет передаваться в параметре path(далее путь) , в столбцах "параметр" перечислены дополнительные параметры - аргументы команд или данные, которые должны передаваться вместе с уведомлением.

Команды будут приниматься tcp сервером на стороне устройства. Уведомления будут отсылаться в потоке, поток будет открываться если к серверу поступила команда, в которой параметр path имеет значение "events", данная команда имеет опциональный аргумент destination, куда будет передаваться путь, с которого необходимо получать уведомления. Если же параметр destination не указан, в поток будут посылаться все уведомления которые есть на устройстве. Уведомления будут посылаться при определенных состояниях, или периодически, далее будет описано более подробно при каких условиях будут высылаться уведомления.

Если отправлена несуществующая команда или передан несуществующий путь уведомлений, устройство посылает ответ, с параметром errors, хранящим строку с описанием ошибки (command not found, destination not found).

Так-же должна быть предусмотрена возможность подключаться к серверу больше чем с одного клиента.

OTA

Должна быть реализована возможность прошить устройство удаленно, OTA должен быть реализован на библиотеке AsyncElegantOTA.

Подключаемая нагрузка

К устройству будет подключено через step dir драйвера 8 шаговых двигателей.

Выходы микроконтроллера, подключаемые к step входам драйверов должны настраиваться через отдельный файл, на этапе сборки программы. На dir входы всех драйверов следует выделить 1 отдельный пин микроконтроллера, он также должен настраиваться через отдельный файл, на этапе сборки программы.

Параметры, хранимые в eeprom

В постоянной памяти устройства, для подключения к wifi сети должны храниться настройки подключения, на данный момент т.к. настройки ip адреса статические, помимо ssid и пароля необходимо хранить ip адрес, маску подсети и адрес шлюза. Также из-за того что в будущем адреса будут динамическими, необходимо заложить переход на dhcp.

Команды устройства

На команды где не указан ответ, ответ по умолчанию параметр message со значением OK, в случае удачной операции, в случае неудачи - параметр errors, со значением в виде строки с описанием ошибки.

/actions/start

По данной команде устройство запускает задачу пройти количество микрошагов, переданных в параметре steps команды, со скоростью, в микро шагах в секунду, переданной в параметре velocity команды, для двигателя motor.

/actions/stop

По данной команде устройство останавливает задачу для двигателя motor. Если на данный момент для двигателя нет текущих задач, в ответе отдать сообщение с ошибкой.

/actions/state

Когда поступает данная команда, в ответ на данную команду устройство должно вернуть массив Json объектов, структура объектов которые будут передаваться в данном массиве описана в api. Json объект хранит в себе состояния задач на каждом моторе. Массив передается в параметре values. Размер массива соответствует количеству моторов. Если на моторе нет никаких задач, все параметры объекта должны иметь значение 0.

/service/settings/network

Когда поступает данная команда, устройство производит перенастройку wifi сети, и подключается к указанной точке доступа, применяя параметры сети переданные в аргументах команды. Если попытка соединения к новой точке доступа не удалась, устройство возвращается к последним удачным настройкам wifi сети, которые должны храниться в постоянной памяти устройства. В противном случае, при удачном подключении к новой сети, настройки перезаписываются.

/service/settings/network/value

Когда поступает данная команда, в ответ на данную команду устройство должно вернуть Json объект, структура которого описана в api. Json объект хранит в себе текущие настройки сети на устройстве.

/service/reload

Данная команда служит для перезагрузки устройства.

Уведомления, посылаемые устройством

/actions/state

Данное уведомление посылается раз в секунду, и хранит в себе массив Json объектов, структура объектов которые будут передаваться в данном массиве описана в api. Json объект хранит в себе состояния задач на каждом моторе. Массив передается в параметре values. Размер массива соответствует количеству моторов. Если на моторе нет никаких задач, все параметры объекта должны иметь значение 0.

/service/settings/network/value

Данное уведомление посылается раз в секунду, и хранит в себе Json объект, структура которого описана в api. Json объект хранит в себе текущие настройки сети на устройстве.

/service/info

Данное уведомление посылается раз в секунду, и хранит в себе статус сети на устройстве

Приложение

Работа событий