

Глава 7. GIT UART

Ежелев Г. И.

12 декабря 2025 г.

Структура проекта

В нашем проекте уже появилось несколько файлов с совершенно разным функционалом - это и управление тактированием и временем, и управление **GPIO** - General Purpose Input Output, и ADC, а скоро еще и **UART** - это цифровой протокол передачи данных.

Настала пора навести порядок в проекте. Начнем с файловой структуры проекта, откройте проводник, чтобы убедиться что у вас так же:

- ▶ **Корень** - в нем лежит файл *.uvprojx - файл-проект.
- ▶ Папка **src**(source - источник/исходник) - в ней лежит весь разработанный вами код и библиотеки.
- ▶ Папка **Objects** - все скомпилированные программные компоненты. Каждый файл - отдельный объект, созданный компилятором. Далее их будет собирать воедино **Linker** от слова Link - связь/ссылка.

Возникает довольно логичный вопрос - как именно распределять файлы и папки. Сейчас вам может показаться что это вопрос из секции подушнить чтобы понтонуться, но вспомните любой свой проект - в какой-то момент он становится таким громоздким, что в нем очень трудно разобраться вам самим. В таком случае жесткая формальная структура приходит на помощь:

- ▶ Layer-first - подход, в котором все файлы распределяются по папкам в зависимости от их "слоя в проекте". Нам важно не то, какие функции используют этот файл, а то чем он является сам - все интерфейсы в папке `interfaces`, вся обработка датчиков в `sensors` и т.д..
- ▶ Feature-first - здесь все файлы, ответственные за одну "фичу" группируются в одну папку:
`camera/{uart.cpp, camera.cpp, gpio.cpp}`
- ▶ В нашем случае лучше использовать именно первый вариант, т.к. мы делаем универсальный проект.

Очевидно, что **каждая пара *.cpp и *.h лежат в отдельной папке** с таким же названием.

В каждом файле должен быть `#include "project_config.h"`

Каждый файл должен быть **добавлен в группу** (папки в левой части экрана)

Каждая папка должна быть указана в
`options for target`(значок волш палочки) → C C++
→ include paths

И все подпапки

GIT

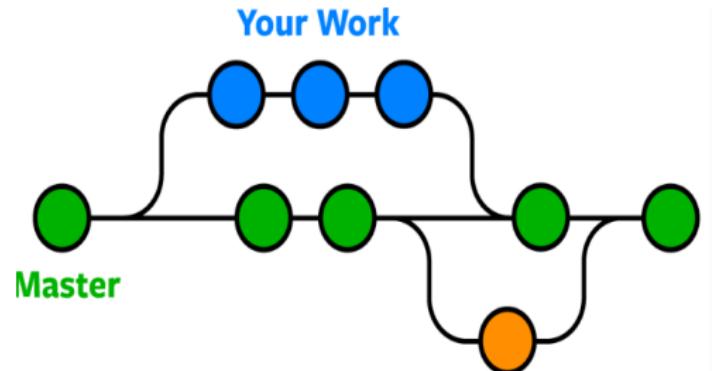
Следующая проблема ой, а куда я жмав, все сломалось.

Чтобы не тратить много времени имеет смысл сохранять **стабильные версии** проекта. Можно сделать zip и отправить себе в телеге способ невероятно удобный.

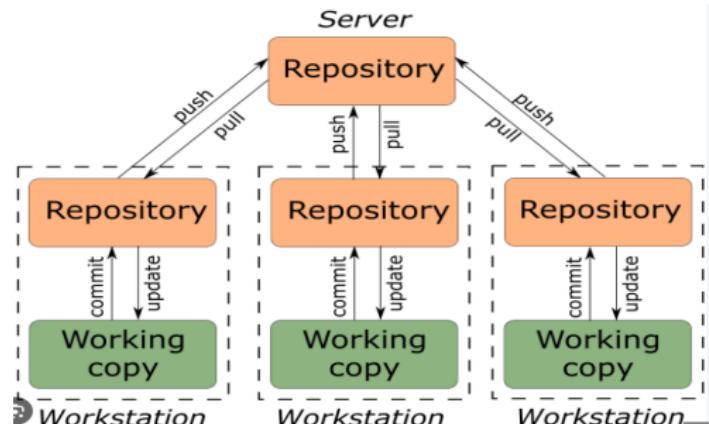
Но если брать более адекватный вариант то это GIT распределенная система управления версиями.

- ▶ В ней ваш проект состоит из **веток(branches)** каждая из них таймлайн вашего проекта.
- ▶ Каждая ветка **таймлайн версий проекта**.
- ▶ Несколько веток позволяют разрабатывать проект параллельно и из каждой брать по кусочку
- ▶ В каждой ветке **сохранены все версии** вы можете откатиться к любой

GIT



Someone Else's Work



Очень советую самостоятельно углубиться в эту тему.

Сейчас только самое важное. Подробности:

[https://www.perplexity.ai/search/
kak-sdelat-pervyi-push-na-gith-80L1XRYNSeKcql5k0RofHw](https://www.perplexity.ai/search/kak-sdelat-pervyi-push-na-gith-80L1XRYNSeKcql5k0RofHw).

UART (Universal Asynchronous Receiver/Transmitter) — универсальный асинхронный приёмопередатчик — это интерфейс для последовательной передачи данных между устройствами **без общего тактового сигнала**. Он преобразует параллельные данные из шины микроконтроллера в последовательный поток битов и обратно, используя две линии: **TX (передача)** и **RX (приём)**. Каждый байт передается в формате кадра:

- ▶ Изначально линия находится в состоянии покоя - **HIGH**
- ▶ Стартовый бит всегда - **LOW**. По спадающему фронту происходит **синхронизация устройств**.
- ▶ Момент чтения каждого следующего бита определяется этим фронтом и **baud rate** - скорость передачи, которая на обоих устройствах должна быть одинакова.

UART

- ▶ Далее идет (обычно) 8 бит данных начиная с LSB (Low significant bit), опциональный PARITY BIT (бит четности - простая проверка корректности приема байта) и 1 или 2 STOP BITS - возврат линии в покой.
- ▶ Одна логическая последовательность байтов называется пакет. У пакета, обычно есть явное начало и конец, чтобы было понятно, какой байт что значит, например:
255, X, Y, 254
- ▶ UART имеет стандартные скорости - загуглите какие
- ▶ устройства подключаются крест-накрест tx1 в rx2 и tx2 в rx1

UART - особенности

UART не самый крутой протокол и имеет множество недостатков:

- ▶ Проблема рассинхронизации
- ▶ Много служебных битов
- ▶ Невозможно подключить более двух устройств для **full duplex**(двухсторонней) коммуникации.
- ▶ Не самая высокая скорость по сравнению с другими протоколами

Все это с лихвой компенсируется его простотой в использовании

