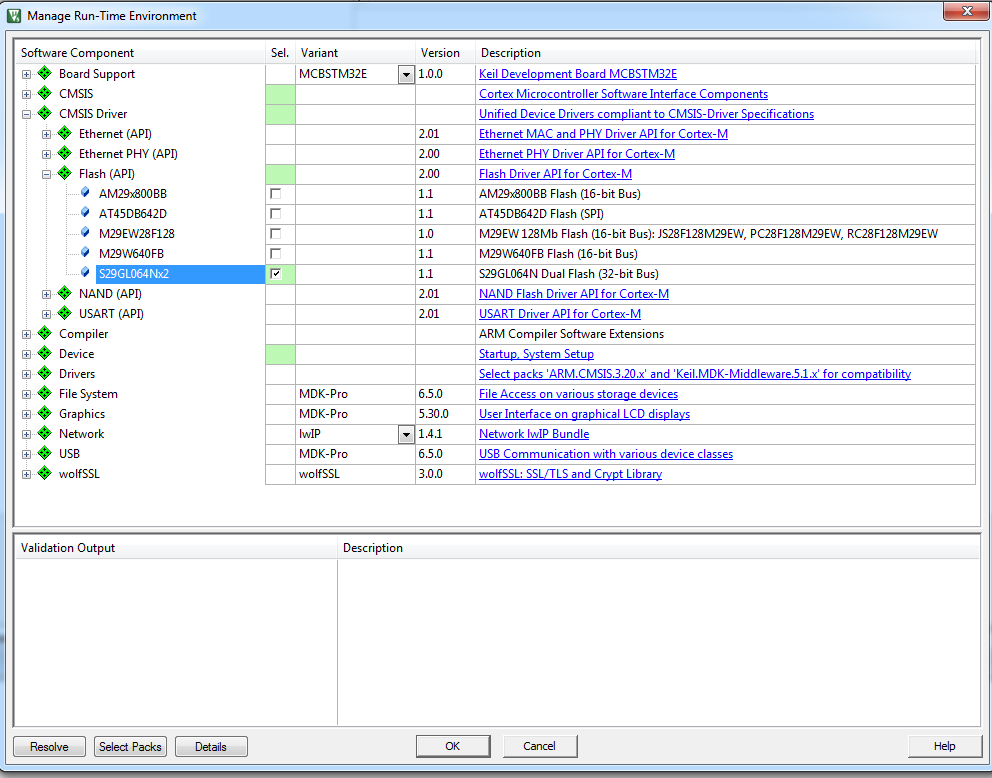
Этот документ написан для того чтобы структурировать мысли по данной системе.

Я занимаюсь разработкой различных устройств, и мне часто приходиться программировать контроллеры. И если честно, то культура программирования у многих разработчиков аппаратуры низкая. И я бы не сказал, что это связано с низким уровнем знаний и умений, для меня лично это связано с тем, что основной упор идет на отладку аппаратной части и поиска ошибок на физическом уровне, где иногда помогают даже танцы с бубном. Поэтому оформление кода занимает самое последнее место в списке дел, работает и хорошо. При этом обычно бывает так, что периферия на плате меняется, а контроллер остается тот же, или наоборот меняется контроллер, а периферия немного изменяется. Добавляются новые функции, которые раньше не были прописаны в ТЗ, или в зависимости от ревизии и использования платы, не устанавливаются различные микросхемы. Поэтому очень часто, копируешь проверенные куски кода из интернета и других проектов в новый и со временем проект превращается в что то ужасное и очень не гибкое. Например, если необходимо отключить часть функционала на плате нужно ввести в проект макросы define, которые будут определять ревизию платы, и это еще хороший вариант, так как при этом некоторые мои коллеги просто начинают новый проект. И ошибки найденные в новом, никак не переносятся в предыдущие версии. Так же достаточно часто, код частично подрезанный, начинает вести не так как планировалось. Поэтому я для себя решил определить некоторые требования к коду прошивки контроллера, чтобы избежать многих проблем. И основное требование это модульность.

Что из этого вышло.

В данной статье я буду рассматривать работу с контроллерами ARM, а точнее Cortex-M с RTOS в Keil. Так как на более мощные контроллеры уже можно устанавливать linux, а это уже другая история. Возможно я изобретаю очередной велосипед , но я не нашел ничего подобного. Я решил написать Middle-Middleware для RTOS-CMSIS и оформить это в виде pack, со всеми плюшками Manage Run-Time Enviroment и Configuration Wizard.



И так требования:

- проект разбивается на модули, модули это неделимая единица, которая исключается из проекта целиком

- модули должны иметь общий интерфейс взаимодействия, т.е. обмениваться данными по определенным правилам наподобие openmax (https://www.khronos.org/openmax/) .

- каждый модуль должен поддерживать визуальную настройку через Configuration Wizard

- создание нового модуля должно быть предельно простым через User Code Template

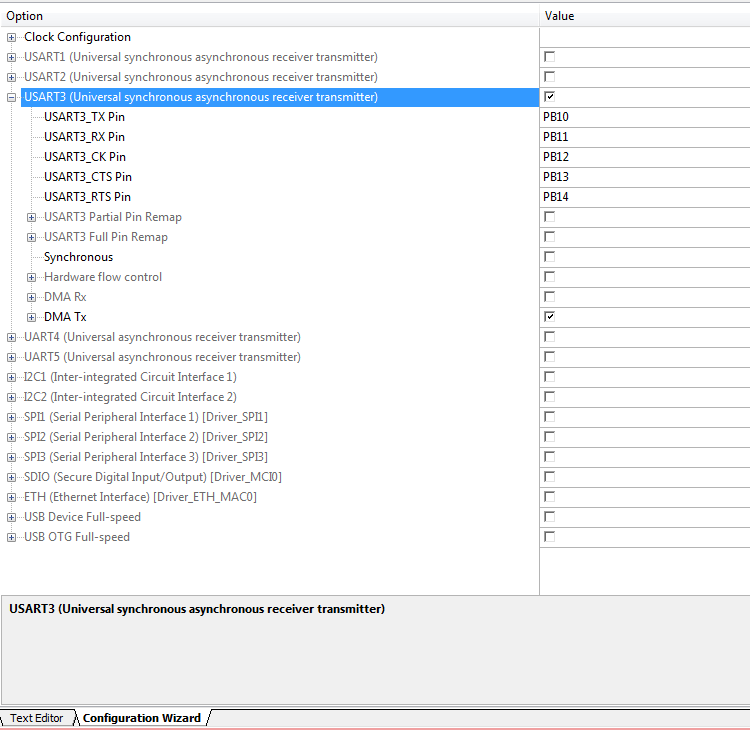
- каждый модуль должен иметь набор тестов по типу PHPUnit.

- модули могут иметь зависимости от других модулей и варианты исполнений в зависимости от цели(Target)

- модули можно добавлять в в общий PACK

**Configuration Wizard**

Пару слов о Configuration Wizard. Configuration Wizard – это возможность менять .h файл, с помощью понятного графического интерфейса. Т.е. если я хочу задействовать один из UART на контроллере, достаточно просто поставить одну галочку. И в моем проекте, будет глобальная переменная с указателем на данный UART, при этом это будет работать через прерывания и DMA .



Взаимодействие модулей

Режим тестирования.

Данный режим тестирования, я считаю самой основной причиной написания данного pack. Обычно программа контроллера пишется, сразу и целиком, со временем начинаешь понимать что это трудный путь , когда происходит переполнения аппаратных буферов, нехватка ресурсов и т.п. Ошибку в рабочей системе очень тяжело найти и исправить. Иногда тесты пишутся в самом коде, и ты выводишь сначала устройство в рабочий режим, а потом генерируешь необходимое событие и пытаешься разобраться, в чем причина. Один такой эксперимент может занимать достаточно много времени. И в 90% случаев можно было бы обойтись синтетическим тестом. Но это очень не удобно делать в рабочей программе, а на чистые тесты в новом проекте не хватает времени, т.к. приходиться конфигурировать кучу переферии. Поэтому я решил придумать удобный инструмент для тестирования любого подключенного модуля к проекту , чтобы он мог запускаться как один так и с зависимостями.

Пример

Как установить