Аппаратное обеспечение IoT/CPS
Лекция 8

А. А. Подшивалов
apodshivalov@miem.hse.ru

Энергосберегающие режимы работы

10 лет работы от батарейки

Другой экспонат выставки — «система сбора показаний ЖКХ», выводящая в интернет данные счетчиков воды, электроэнергии или газа. Среди решающих преимуществ прибора — «10 лет работы от батарейки». Моя попытка выяснить у разработчиков, откуда такая уверенность в том, что прибор, изобретенный буквально вот только что, проработает на одной батарейке 10 лет, и испытывали ли эту батарейку эти 10 лет, закончилась крахом: мне объяснили, что это и так с самого начала ясно всем разработчикам этого проекта.

Андрей Колесников, «Коммерсант», Владимир Путин зачекинился в сети 11.06.2014

Практическая задача

- Срок службы водосчётчика — 8 лет (6 лет межповерочный интервал, 2 года срок сохраняемости)
- Батарейка в водосчётчике — 3,6 В, 2400 мА*ч (Li-$SOCl_2$)
- 8 лет $= 24 \times 365 \times 8 = 70080$ часов
- Среднее энергопотребление — не более $2400/70080 = 0,034$ мА

Методы экономии энергии

- Понижение тактовой частоты процессора
- Приостановка работы ядра процессора
- Полное отключение ядра процессора
- Отключение периферийных устройств
- Отключение оперативной памяти

У разных микроконтроллеров — разный набор доступных методов энергосбережения

Режимы работы на примере STM32L

RUN	Работа без каких-либо ограничений	1–20 мА
Low Power RUN	Серьезно ограничена тактовая частота (131 кГц)	0,01–0,1 мА
SLEEP	Приостановка работы ядра с сохранением тактирования	0,1–1 мА
Low Power SLEEP	Приостановка работы ядра с сохранением тактирования 131 кГц	0,005–0,03 мА
STOP	Остановка основных тактовых частот с сохранением RAM	0,001–0,01 мА
STANDBY	Почти полное выключение микроконтроллера	0,0005–0,001 мА

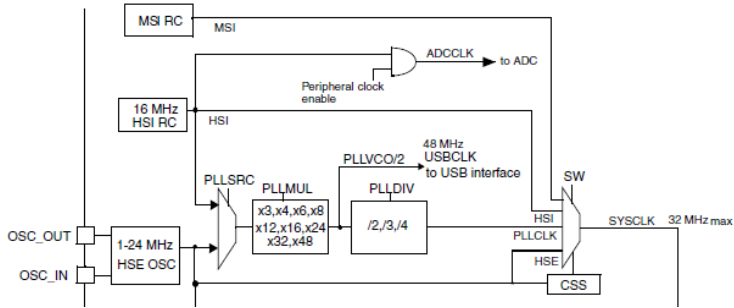
Режимы работы — или кто же не спит в лесу

- **RUN**
  - Ядро процессора, ОЗУ, все модули периферии, которые вы включили
- **SLEEP**
  - ОЗУ и все модули периферии, которые вы включили
- **STOP**
  - ОЗУ, прерывания GPIO, RTC, Low-Power (LP) Timer, LP UART
  - GPIO сохраняют состояние, но...
  - ...неотключенные модули периферии могут управлять GPIO
- **STANDBY**
  - RTC и несколько (1–5) GPIO специального назначения (wake-up pins, tamper protection)
  - Состояние GPIO сброшено (смотрим даташит!)

Переключение режимов (RUN–STOP–RUN)

```
/* флаг PDDS определяет выбор между Stop и Standby, его надо сбросить */
PWR->CR &= ~(PWR_CR_PDDS);
/* флаг Wakeup должн быть очищен, иначе есть шанс проснуться немедленно */
PWR->CR |= PWR_CR_CWUF;
/* стабилизатор питания в low-power режим, у нас в Stop потребления-то почти не будет */
PWR->CR |= PWR_CR_LPSDSR;
/* источник опорного напряжения Vref выключить автоматически */
PWR->CR |= PWR_CR_ULP;
/* с точки зрения ядра Cortex-M, что Stop, что Standby - это режим Deep Sleep */
/* поэтому надо в ядре включить Deep Sleep */
SCB->SCR |= (SCB_SCR_SLEEPDEEP_Msk);
/* выключили прерывания; пробуждению по ним это не помешает */
unsigned state = irq_disable();
/* завершили незавершённые операции сохранения данных */
__DSB();
/* заснули */
__WFI();
/* проснулись - переинициализация рабочих частот, иначе попадем в прерывание на MSI */
init_clk();
/* и восстановили прерывания */
irq_restore(state);
```

Выбор источника тактирования



- В ОС Riot делается в `init_clk()`
- Варианты:
  - MSI — на нем МК запускается, около 4,2 МГц
  - HSI — 16 МГц ± 5%
  - HSE — 1-24 МГц ± 0,005% (50 ppm)

Процедура переключения режимов

- От STOP к RUN, 32 МГц, внешний кварцевый резонатор (HSE)

Пробуждение ядра МК	Включение кварцевого резонатора	Включение PLL	Переключение на PLL
10 мкс ядро 4,2 МГц	2000 мкс HSE 24 МГц Ядро 4,2 МГц	160 мкс PLL 32 МГц Ядро 4,2 МГц	15 мкс Ядро 32 МГц

- Общее время: 2185 мкс
- Точность частоты генератора: $\pm 0,005\%$

Процедура переключения режимов

- От STOP к RUN, 32 МГц, внутренний RC-генератор (HSI)

Пробуждение ядра МК	Включение внутреннего RC-генератора	Включение PLL	Переключение на PLL
10 мкс ядро 4,2 МГц	6 мкс HSI 16 МГц Ядро 4,2 МГц	160 мкс PLL 32 МГц Ядро 4,2 МГц	15 мкс Ядро 32 МГц

- Общее время: 191 мкс
- Точность частоты генератора: $\pm 5\%$

Что учитывают при выборе режима?

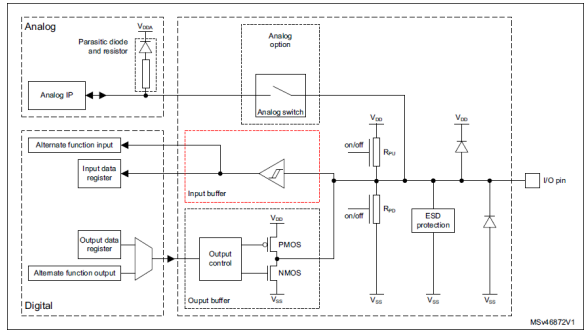
- Доступные периферийные устройства (включая GPIO)
- Энергопотребление в данном режиме
- Производительность в данном режиме (интегральное энергопотребление)
  - На вычислительной задаче RUN — лучше, чем LP RUN
  - На ожидании события LP RUN — лучше, чем RUN
- Время переключения режимов
  - Время выхода процессора из энергосберегающего режима
  - Время запуска генераторов тактовых частот
  - Время настройки тактовых частот и периферии

ВСЁ!

(На самом деле нет)

Важные мелочи (для STM32L1 и не только)

- Триггер Шмитта на GPIO
  - При уходе в STOP переключаем GPIO в Analog IN, иначе получим потребление 100–200 мкА
- Периферийные устройства (в том числе JTAG) непредсказуемым образом управляют GPIO
  - Отключаем Alternate function и устанавливаем нужный уровень самостоятельно
- При уходе в STANDBY GPIO переключаются в режим по умолчанию (Обычно Hi-Z)
  - Добавляем внешние подтяжки



Энергопотребление IoT-системы

Немного цифр

- STM32L151
  - RUN — 9,6 мА, SLEEP — 2,2 мА, LP RUN — 84 мкА, STANDBY — единицы мкА
- TI CC3200 (Cortex-M4 + Cortex-M0 + Wi-Fi)
  - MCU Active, NWP IDLE Connected — 15,3 мА
  - MCU Sleep, NWP IDLE Connected — 12,2 мА
  - Hibernate (standby) — 4 мкА
  - TX — 166–278 мА, RX — 50–60 мА
- nRF52840 (Cortex-M4 + IEEE 802.15.4/BLE)
  - CPU — 3–6 мА, TX — 6–16 мА, RX — 6–10 мА
- SX1276
  - Standby — 1,6 мкА, TX — 20–120 мА, RX — 12 мА

Пример практического расчета

- Водосчётчик с приёмопередатчиком LoRa
  - Потребление в режиме STOP — 4 мкА при 25°C
  - Измерение — 10 раз в секунду, 0,2 мс, 10 мА
  - Передача данных — 1 раз в 4 часа, 1,5 с, 25 мА
- Среднее потребление

$$4 + 10 \times \frac{0,2}{1000} \times 10 \times 10^3 + \frac{1,5}{3600} \times \frac{1}{4} \times 25 \times 10^3 = 27 \text{ мкА}$$

- Технологический разброс — 25%, итого 34 мкА
- Батарейка ER14505 (Li-$SOCl_2$) — 3,6 В, 2400 мА×ч
- $\frac{2400000}{34} = 68571$ часов $= 8,05$ года

Важные мелочи-2

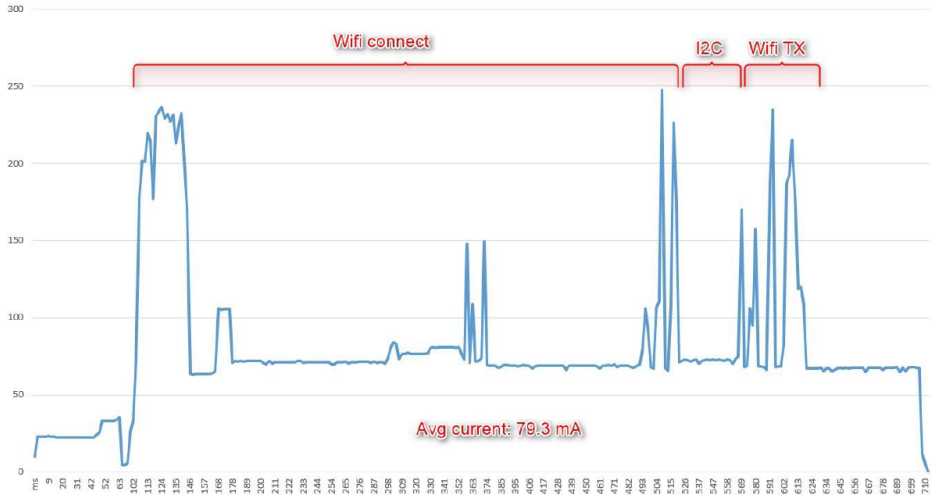
- Энергопотребление растёт с ростом тактовой частоты
  - 0,7 мА @ 4 МГц vs. 7 мА @ 32 МГц
- Энергопотребление растёт с ростом температуры
  - 1 мкА @ 25° C vs. 2 мкА @ 85° C
- Энергопотребление растёт с ростом напряжения питания...
  - ...если в составе нагрузки нет DC-DC преобразователя
- Минимальный ток DC-DC и LDO
  - Не все преобразователи питания стабильно работают при малых нагрузках
  - Не забываем про I_Q

Важные мелочи-3

- Проверяйте энергопотребление всей системы в целом
  - Акселерометр LIS3DH — power-down 0,5 мкА, 1 Гц — 2 мкА, 50 Гц — 11 мкА, 1344 Гц — 185 мкА
- При батарейном питании — не забывайте про свойства батареек и аккумуляторов
  - Емкость падает с понижением температуры — 2400 мА×ч @ 25°C vs. 1600 мА×ч @ −20°C
  - Эффект пассивации (особенно для Li-$SOCl_2$)
  - Максимальный ток нагрузки
  - Максимальное и минимальное напряжение
- Не забываем про передачу данных — и про потребление приемника
  - Wi-Fi — TWT, target wake time
  - LoRaWAN — классы A/B/C, RX1 и RX2
  - mesh-сети и «спящие маршрутизаторы», Contiki MAC
  - Bluetooth Low Energy и передача данных в заданные интервалы времени

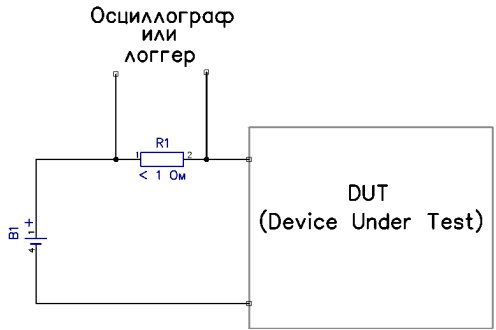
Измерение энергопотребления

ESP8266 — сеанс связи



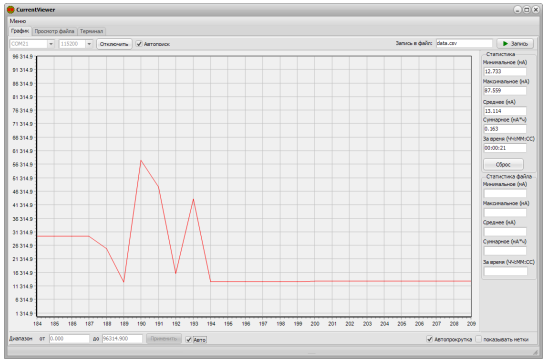
Измерение энергопотребления

- Диапазон токов — от 1 мкА до 1 А (10^6, 20 бит)
- Падение напряжения на шунте R1 — не более 100 мВ
- Разрешение по времени — не хуже 100 мкс (частота дискретизации от 10 кГц)

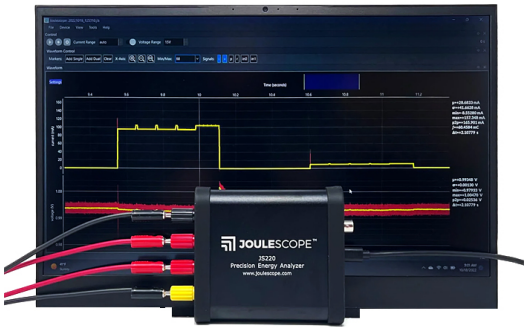


Измерение энергопотребления: UMDK-RF

- Шунт на один диапазон, разрешение 16 мкА, 12 бит (максимальный измеряемый ток 65 мА)
- Частота дискретизации АЦП 100 кГц, выдача в UART усредненных данных с частотой около 10 Гц
- Встроенный отладчик и USB-UART преобразователь



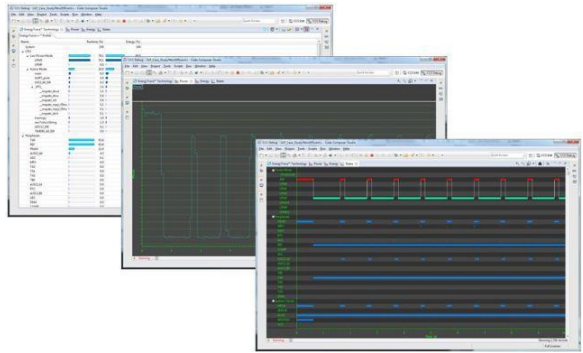
Измерение энергопотребления: Joulescope



- Переключаемые шунты на 7 диапазонов
- Измерение тока от -1 до 3 А, разрядность АЦП 12 бит
- Частота дискретизации 2 МГц, полоса пропускания 250 кГц

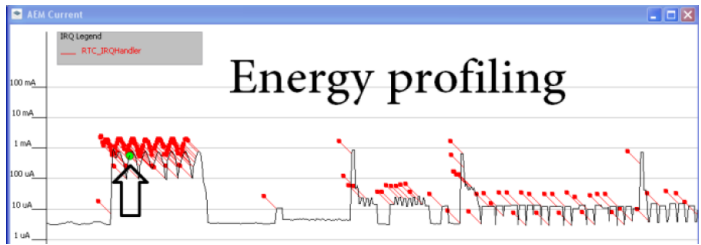
Измерение энергопотребления: TI EnergyTrace

- Измерение скважности импульсов на ключе DC-DC преобразователя
- Измеряет только потребленную энергию (хотя этого часто достаточно)

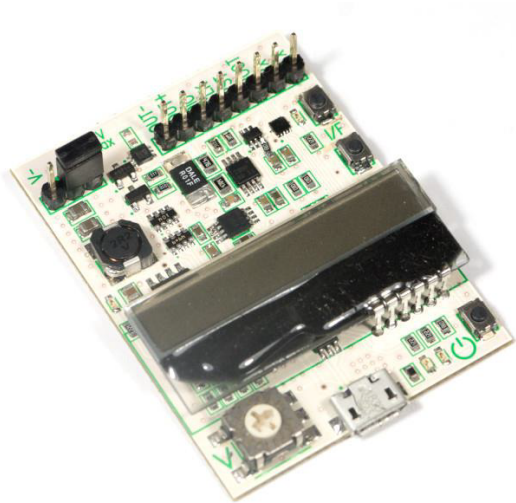


Измерение энергопотребления: SiLabs STKxxxx

- Встроенный двухдиапазонный усилитель шунта
- Фоновое энергопотребление около 5 мкА
- Частота замеров 100 Гц (зелёные платы) или 6250 Гц (чёрные платы)



Измерение энергопотребления: Energymon



- Встроенный трехдиапазонный усилитель шунта
- Встроенный DC-DC или внешний источник питания, измерение напряжения
- Частота замеров 300 кГц, выдача данных в UART — 10-100 Гц
- Отладчик и USB-UART преобразователь можно отключить от МК

Механизмы операционной системы

Riot: `pm_layered`

- Автоматический переход в энергосберегающий режим в потоке IDLE
- Модули ОС, драйверы и программы могут задавать *минимальный разрешённый* энергосберегающий режим
- Сложные схемы (выход из сна на разные частоты и в разные режимы) не реализованы

Память микроконтроллера

Несколько общих слов

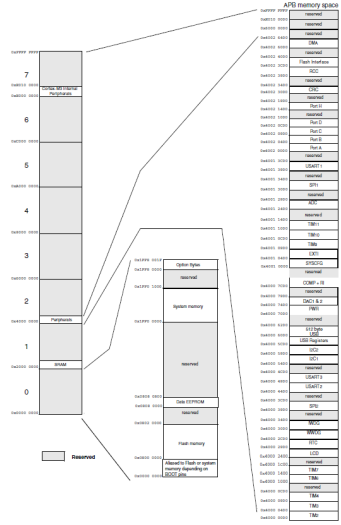
- Гарвардская архитектура — раздельная память программ и данных
- Фон-неймановская архитектура — общая память программ и данных
- Ввод-вывод с отображением на память (memory-mapped I/O)

Организация памяти МК

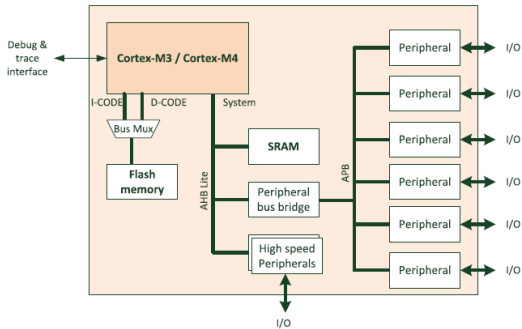
- PIC10/12/16/18
  - Типичный пример гарвардской архитектуры
  - Раздельные ROM и RAM
  - Special Function Registers в адресном пространстве RAM
- 8051
  - IRAM — внутренная RAM, 128 или 256 байт, 8-битная адресация
  - SFR в адресном пространстве IRAM
  - XRAM — до 64 кБайт (16-битная адресация)
  - PMEM — память программ, до 64 кБайт (или больше с переключаемыми страницами)

ARM Cortex-M

- Общий размер адресного пространства у 32-битного ядра — 4 Гб
- В единое адресное пространство проецируются:
  - ROM (встроенная Flash-память)
  - RAM
  - Регистры периферии
- С точки зрения программиста — типичнейший пример фон-неймановской архитектуры



ARM Cortex-M

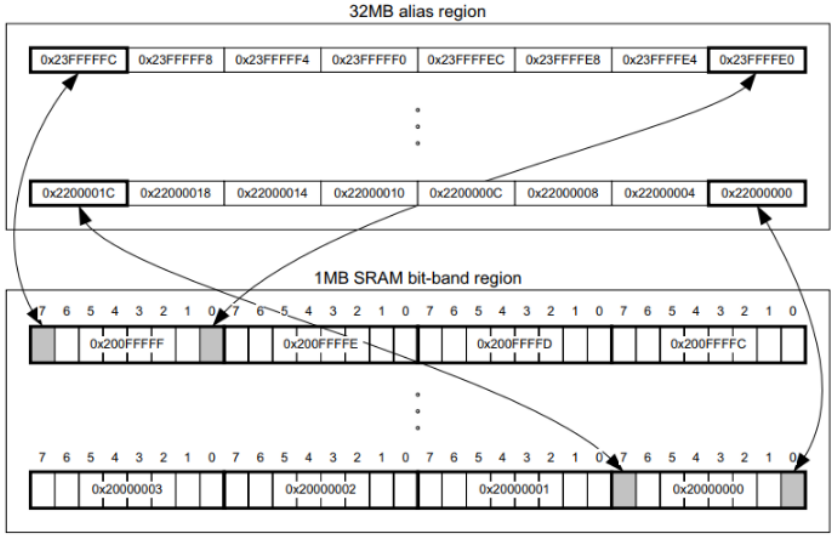


- «Внутри» ядро Cortex-M имеет гарвардскую архитектуру памяти
- Это позволяет одновременно с загрузкой инструкции из ROM загружать данные из RAM
- Выполнение кода из RAM возможно с замедлением примерно в 2 раза
- Для встроенной периферии используются шины AHB (AMBA High-performance Bus) или APB (AMBA Peripheral Bus), описанные в стандарте AMBA (Advanced Microcontroller Bus Architecture)

DMA — контроллер прямого доступа к памяти

- «Асинхронный `memcpy`»
- Может копировать данные с одного адреса в памяти на другой без участия процессорного ядра
- Автоинкремент адресов источника и приемника (настраивается)
- Может управляться от периферии (UART, SPI, I^2C, ADC, ...)
- Счетчик количества операций
- При достижении середины и конца буфера генерирует прерывание

ARM Cortex-M: bit-banding



Виды памяти

Встроенная память МК

- RAM
  - Static RAM — дорогая (6 транзисторов/бит), но не требует «регенерации» и имеет низкое энергопотребление
- ROM
  - Flash
  - EEPROM

Flash-память

- Чтение происходит «строками» по 32, 64 или 128 бит, довольно медленное
  - Контролер памяти скрывает это от программиста
  - Между ядром процессора и flash часто находится «ускоритель flash» — небольшой объем кеш-памяти
- Возможна только запись 0 на место 1
  - Стирание памяти (запись 1 во все биты) производится «страницами», типичный размер — от 256 байт до 64 кБайт
  - Ресурс выражается в циклах стирание-запись, обычно около 10 000 циклов
- Контролер flash-памяти — отдельное периферийное устройство, позволяет стирать flash и записывать туда данные
- Типичный объем — десятки-сотни кБайт (256 кБ в STM32L151CC)

EEPROM

- Возможно побайтовое чтение и запись
- Существенно больший ресурс, чем у flash, 100 000 – 1 000 000 циклов
- Типичный объем — единицы кБайт (8 кБ в STM32L151CC)

FSMC — Flexible Static Memory Controller

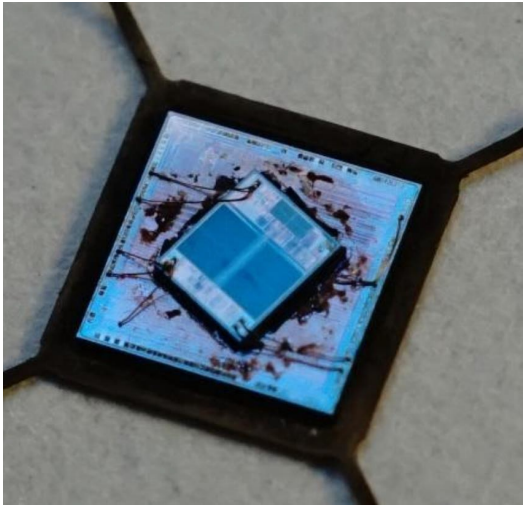
- Внешняя память проецируется в адресное пространство микроконтроллера (адреса с 0x60000000 по 0x9FFFFFFF)
- Шина адреса до 26 бит
- Шина данных до 16 бит
- «Служебные» сигналы read strobe, write strobe, chip select
- Можно подключать не только память, но и все, имеющее схожий интерфейс (например, ЖК-дисплеи с параллельной шиной)

Экзотика

- TI CC3200
  - Встроенная флеш-память очень мала и программисту недоступна
  - Программа хранится во внешней микросхеме памяти с последовательным интерфейсом QSPI, при запуске МК считывается оттуда в оперативную память
- TI CC3220SF
  - 1 Мб встроенной флеш-памяти, XIP (eXecute In Place)
  - Ускоритель не работает, память очень медленная, производительность в 1,5 раза ниже предыдущей модели
  - Внешняя микросхема памяти все равно нужна

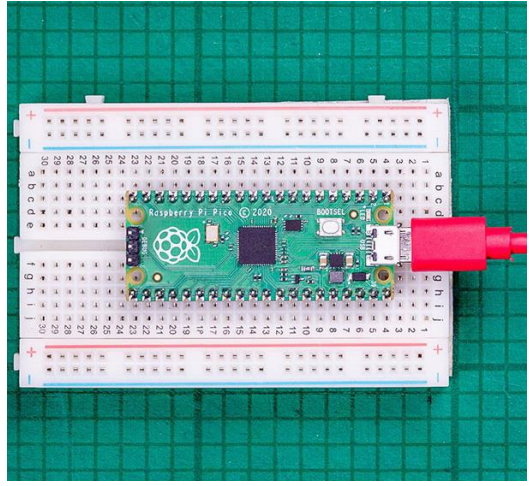
Экзотика-2

- GigaDevice — китайский производитель микросхем флеш-памяти и микроконтроллеров
- GD32 — клоны STM32 на ядре Cortex-M
- GD32V — собственные разработки на базе RISC-V
- В одном корпусе собраны микроконтроллер с периферией и отдельный чип flash-памяти с последовательным интерфейсом



Экзотика-3

- RP2040 — попытка Raspberry Pi Foundation сделать микроконтроллер
- Встроенная flash-память отсутствует, программа загружается в SRAM из внешней микросхемы памяти с интерфейсом QSPI



С точки зрения программиста

*.ld и *.map

- Linker script — файл с расширением `ld`
  - Описывает расположение данных в памяти
  - Обычно секция `.text` и копия `.data` помещаются в ROM
  - Задача обработчика сброса — обнулить `.bss` и скопировать `.data` в RAM
- Map file — файл с расширением `map`

Bootloader

- Выполняющаяся на микроконтроллере программа для записи «прошивки» в собственный Flash
- Очень часто bootloader «зашит» в микроконтроллер уже при изготовлении, и выбор программы для запуска (bootloader либо штатная прошивка) осуществляется переключением логических уровней на нескольких специально выделенных выводах
- Bootloader в STM32 умеет работать с UART и USB (по протоколу DFU, Device Firmware Update)

Постоянная память и RIOT OS

- Драйверы Flash и EEPROM
- Поддержка bootloader
- Имитация EEPROM в Flash-памяти (для некоторых семейств процессоров)
- В EEPROM/Flash можно сохранять настройки устройства, сессионные ключи, идентификаторы, ...

`drivers/include/periph/eeprom.h`

Файловые системы для Flash

- Когда flash-памяти много, можно организовать в неиспользуемой ее части полноценную файловую систему
- Основная проблема — ограниченное количество циклов перезаписи
- Решение — стирать данные только при «переполнении» ФС, поддерживать версии файлов