



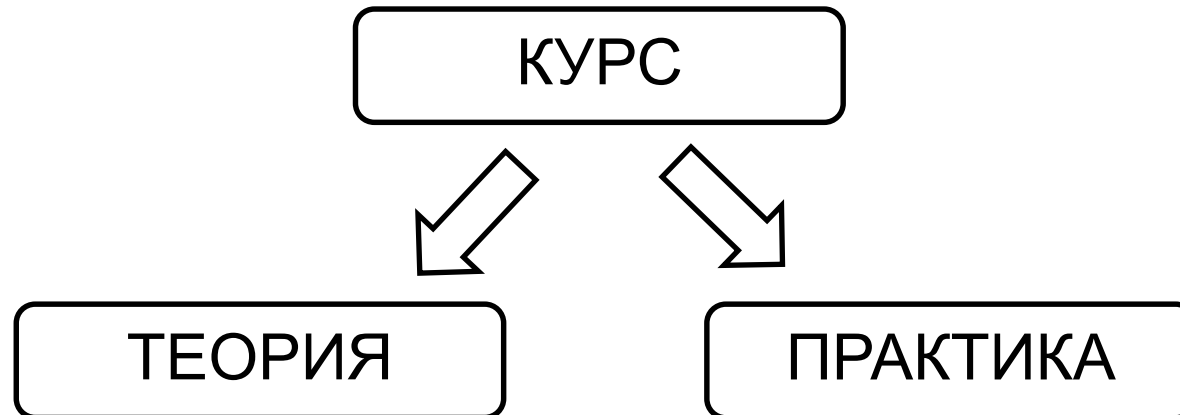
STM &  XILINX.

# «Программирование» встраиваемых систем на базе микроконтроллеров и ПЛИС

Владимир Хрусталеv  
Email : v\_crys@mail.ru

Часть I  
Знакомство с STM32

«В ВУЗе нужно излагать материал на высоком профессиональном уровне. Но поскольку этот уровень проходит значительно выше головы среднего студента, я буду объяснять на пальцах. Это не очень профессионально, зато понятно».  
*Неизвестный преподаватель*



# План курса

1. Часть I. Программирование микроконтроллеров
  1. Основы STM32. Обзор средств разработки. Первая программа.
  2. Таймеры и прерывания. Разрабатываем секундомер.
  3. Внешние интерфейсы STM32. «Связываем» микроконтроллер с компьютером.
2. Часть II. Проектирование микросхем и их отладка на FPGA
  1. Знакомимся с FPGA и SystemVerilog. Обзор средств разработки. Первые опыты.
  2. Цифровая схемотехника. Синхронная и комбинаторная логика. Конечные автоматы. Разрабатываем светофор.
  3. Верификация. Разработка аппаратного модуля UART.
3. Часть III. (опционально) О разработке процессоров. Архитектура и микроархитектура. «Живой» пример: процессор Syntacore SCR1

# Введение в курс

Курс является логическим продолжением занятий по проектированию простых цифровых устройств. Материалы занятий по проектированию устройств выложены здесь:

[https://github.com/v-crys/AD19\\_C1\\_L1](https://github.com/v-crys/AD19_C1_L1)

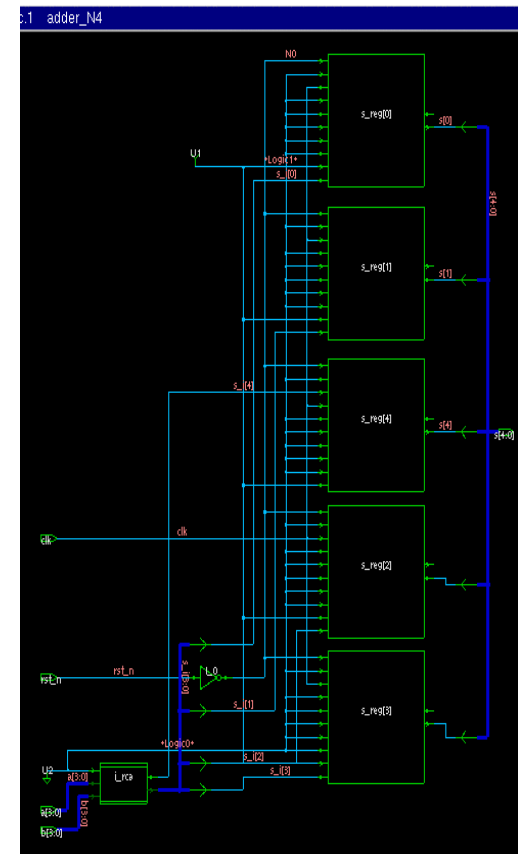
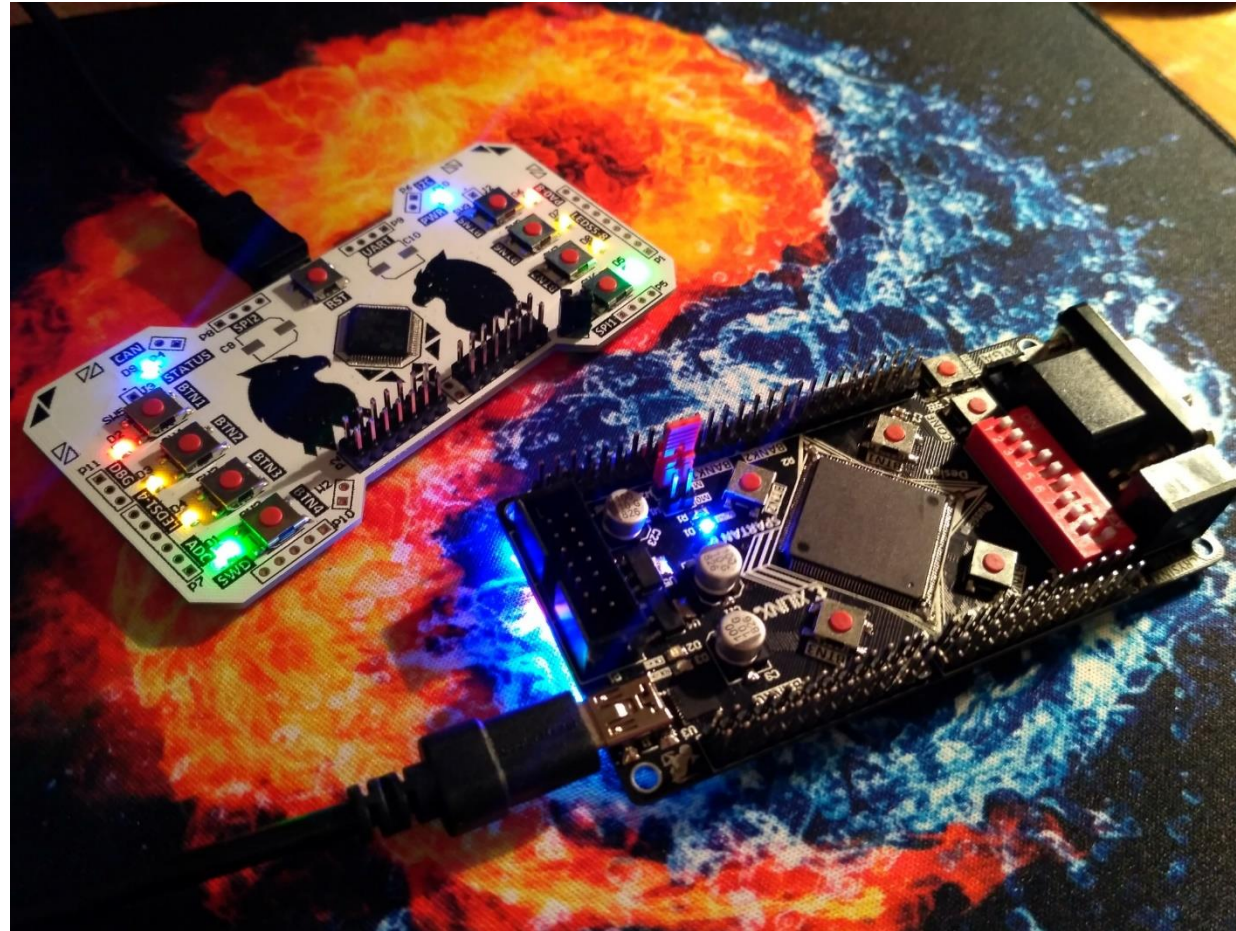
В настоящем курсе будут рассмотрены вопросы разработки прошивки для спроектированного нами устройства. Разработанная прошивка позволит превратить кусок «мертвого» текстолита в полезный гаджет.

Во второй части курса будет рассмотрен вопрос проектирования собственных микросхем и их прототипирование на ПЛИС.

Курс направлен на развитие навыков у студентов в областях аппаратной разработки и программирования встраиваемых систем.

# Mcu VS FPGA

```
C main.c x rds.github.grc
D:\> files > work > stm32 > test > Src > C main.c
1
2 #include "main.h"
3 #include "usb_device.h"
4
5 void SystemClock_Config(void);
6 static void MX_GPIO_Init(void);
7
8 void led_indic(int type);
9 char check_btn();
10
11 int main(void)
12 {
13
14     HAL_Init();
15     SystemClock_Config();
16
17     MX_GPIO_Init();
18     MX_USB_DEVICE_Init();
19
20     led_indic(1);
21     HAL_Delay(1000);
22     while (1)
23     {
24         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_15);
25
26         HAL_Delay(1000);
27         led_indic(0);
28
29         if ((check_btn() & 1) != 0) led_indic(2);
30         if ((check_btn() & 2) != 0) led_indic(1);
31     }
32
33 }
34
35 char check_btn() {
36     char res = 0;
37     char tmp = 0;
38
39     tmp |= ~HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_11) & 1;
40     tmp |= (~HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_12) & 1) << 1;
41     res = tmp;
42     HAL_Delay(5);
43     tmp ^= ~HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_11) & 1;
44     tmp ^= (~HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_12) & 1) << 1;
45
46     res ^= ~tmp;
47
48     return res;
49 }
50
51 void led_indic(int type) {
```



# Занятие №1

## ТЕОРИЯ

# Что такое микроконтроллер

Микроконтроллер это небольшая микросхема, на кристалле которой собран настоящий микрокомпьютер! Это означает, что внутри одной микросхемы смонтировали процессор, память (ПЗУ и ОЗУ), периферийные устройства, заставили их работать и взаимодействовать между собой и внешним миром с помощью специальной микропрограммы, которая храниться внутри микроконтроллера.



# Классификация микроконтроллеров

**МК**

**По разрядности  
шины данных**

- 8
- 16
- 32
- 64

**По назначению**

- общего
- специального (DSP,...)

**По архитектуре**

- Гарвардская
- Фон-Неймановская

**По набору  
инструкций**

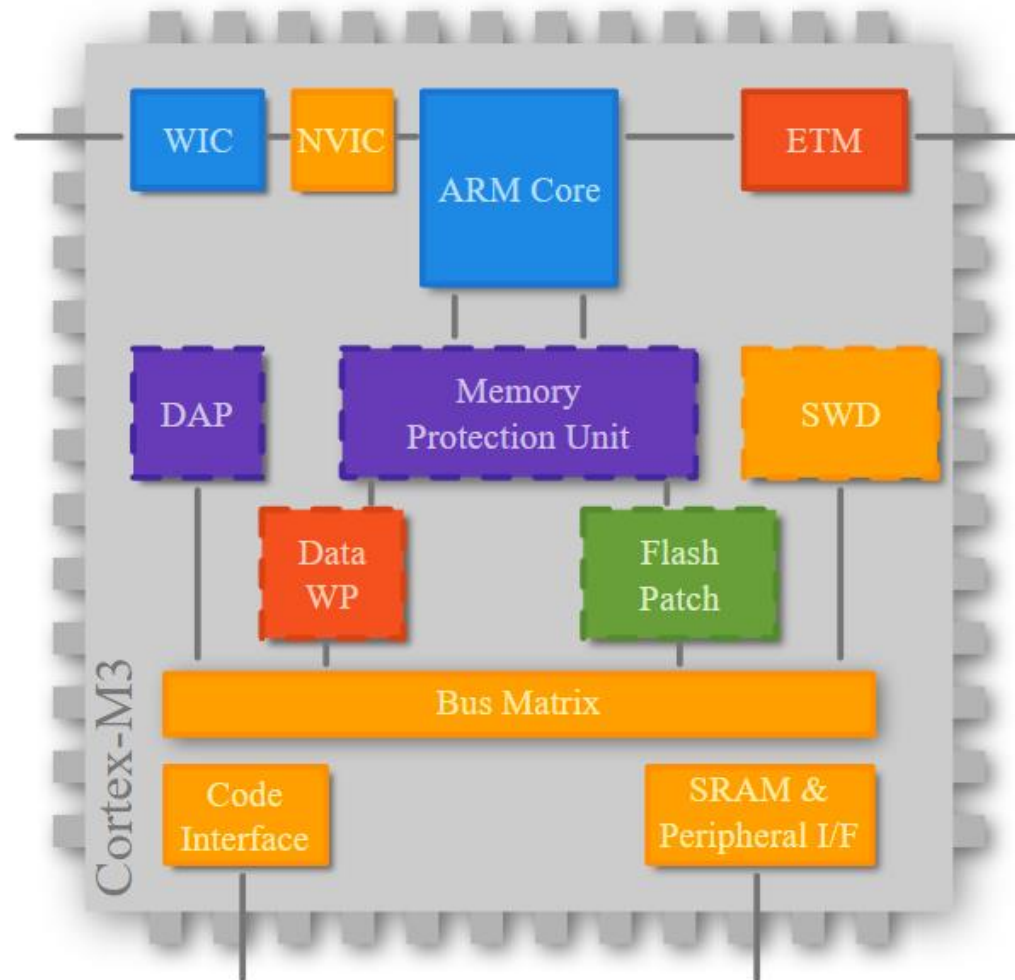
- CISC
- RISC

**Система команд**

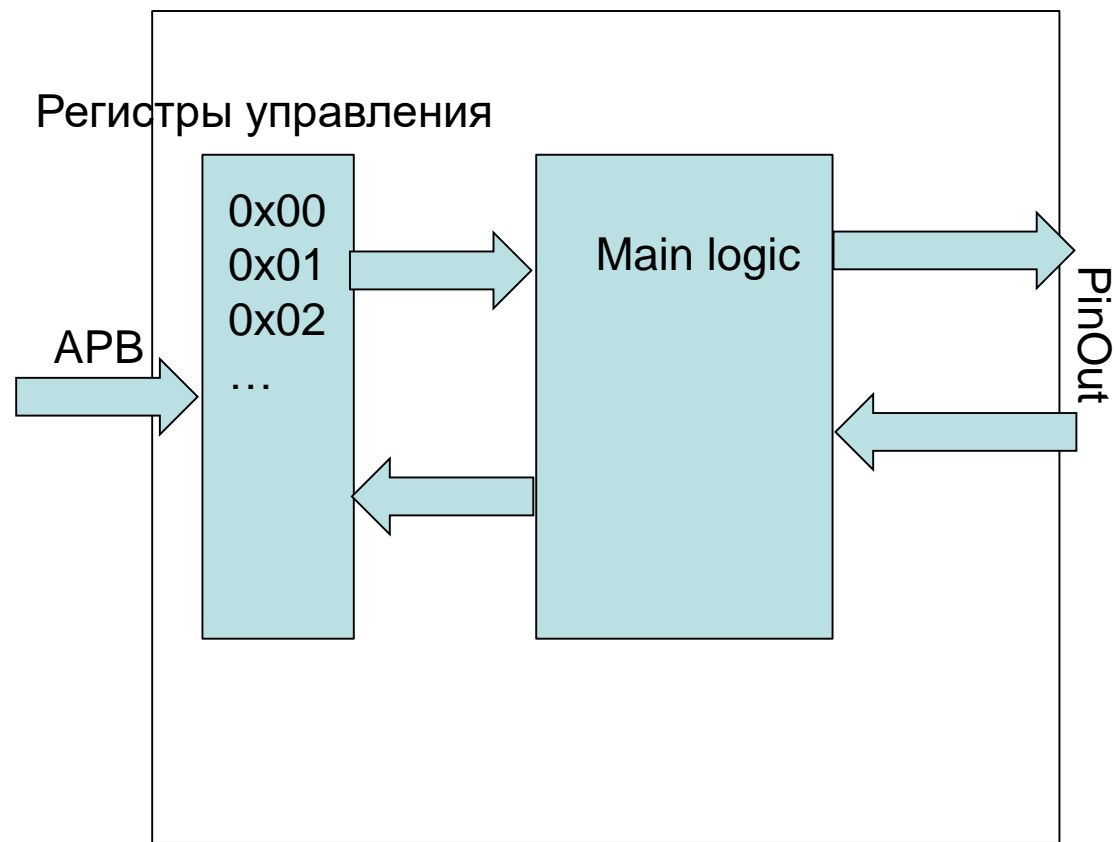
- RISC-V
- ARM
- MIPS



# Внутри МК

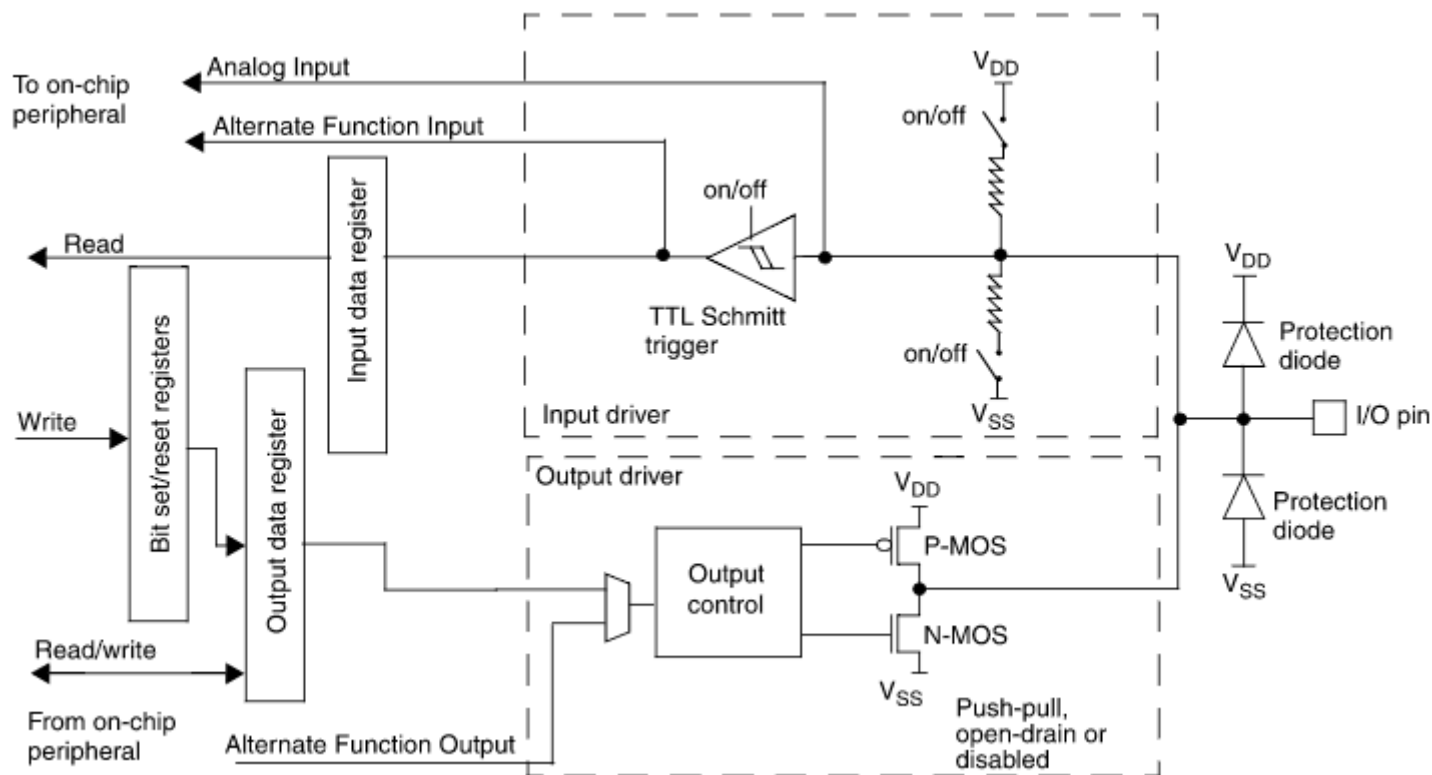


## Пример периферийного блока



# Порты ввода-вывода GPIO

**Порт ввода-вывода – несколько выводов микроконтроллера, управляемых одним регистром. Ниже устройство одного вывода микроконтроллера.**



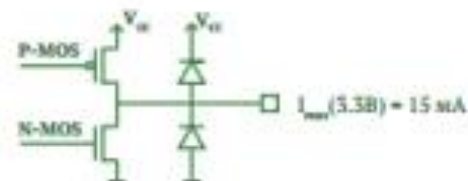
# Режимы GPIO



## Двухтактный \ push-pull

Применение:

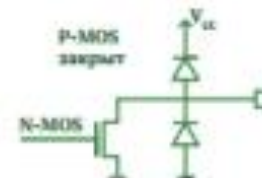
- нагрузка вкл/выкл;
- всевозможные ногодрыги;
- управление индикаторами.



## Открытый сток \ open drain

Применение:

- работа с I2C, 1-Wire;
- управление 5В нагрузкой;
- обеспечение совместимости 3В и 5В логики.

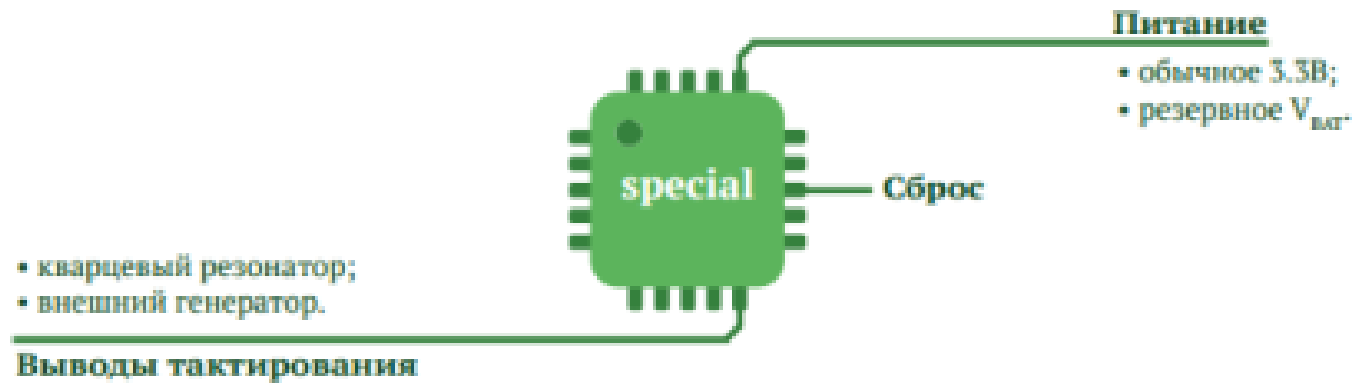


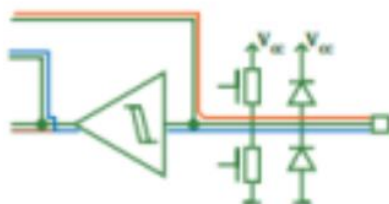
## Альтернативный \ alternative

Применение:

- ЦАП, АЦП;
- USB, UART, SPI и т.д.

# Режимы GPIO





## ■ Альтернативный \ alternative

Применение:

- захват таймеров;
- внешние прерывания;
- внешняя память.

## Плавающий вход \ floating

Применение:

- приём сигналов НЛО.

## ■ Аналоговый \ analog

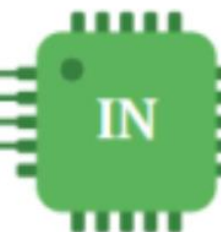
Применение:

- АЦП;
- компараторы;
- операционные усилители.

## ■ Цифровой \ digital

Применение:

- сигналы от кнопок;
- программные шины.



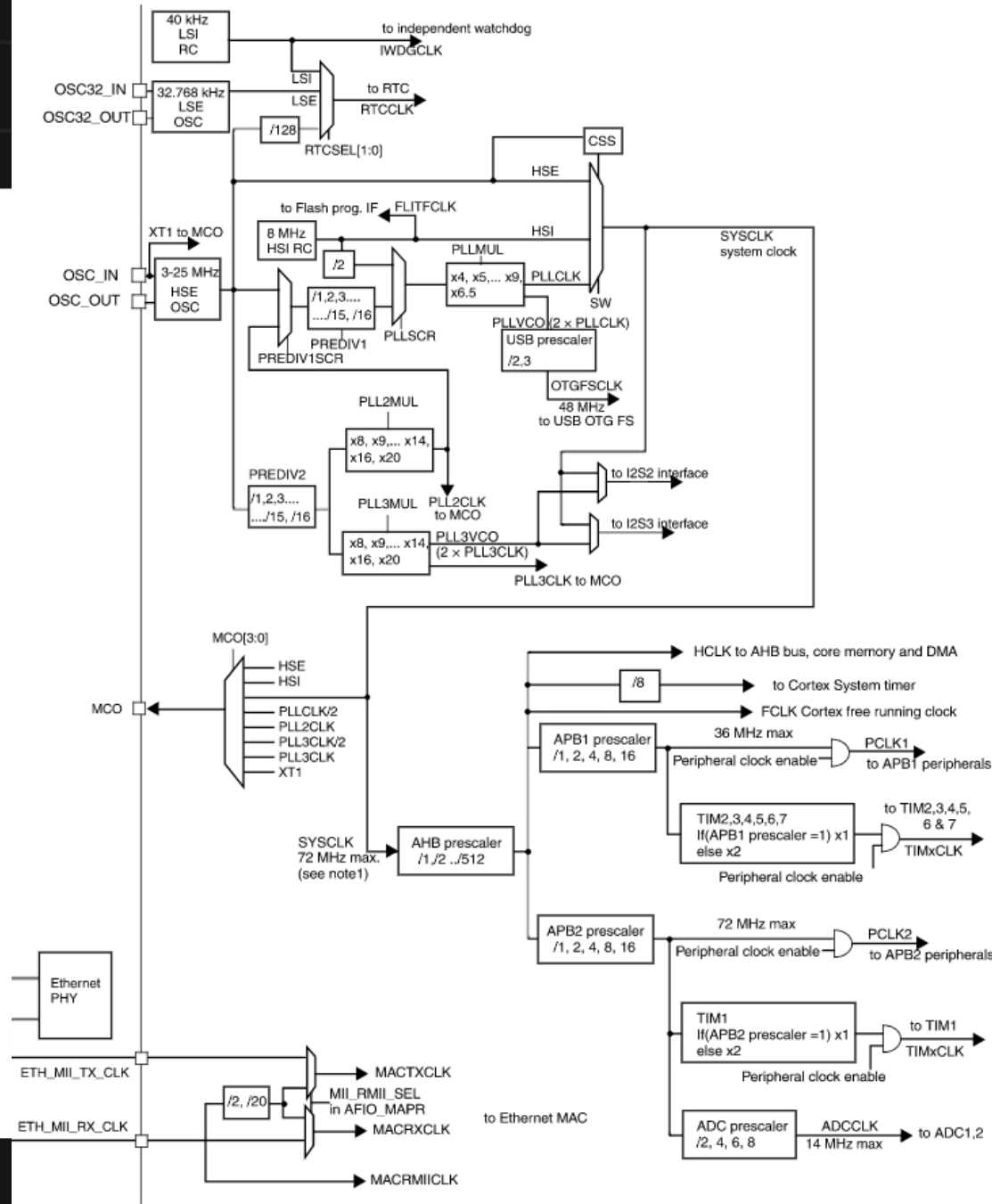
# Система тактирования МК

Большая часть системы тактируется от линии SYSCLK за исключением блоков USB, RTC и т.д. Источником сигнала для неё могут выступать три источника HSI, HSE и PLL.

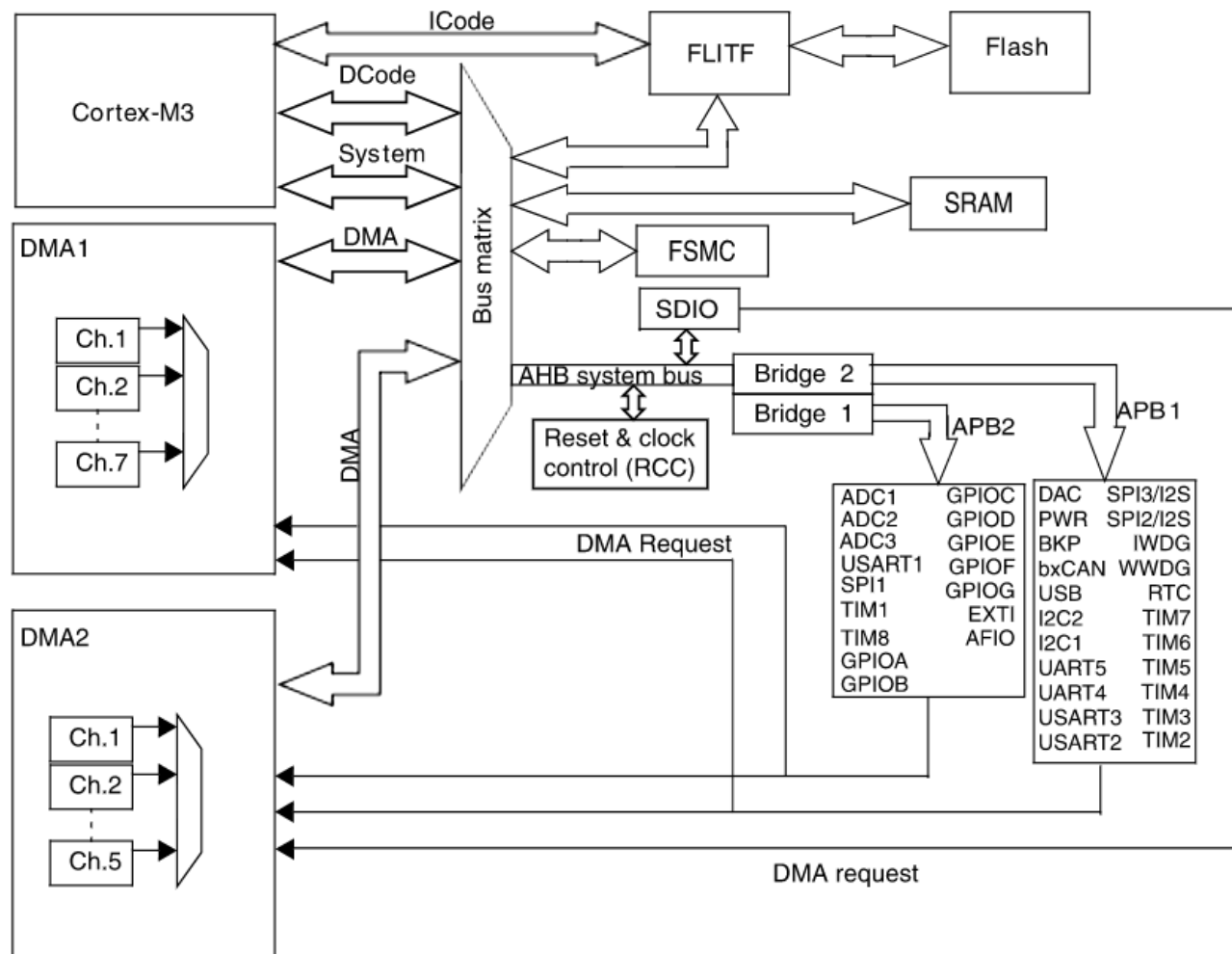
**HSI** (сокр. High Speed Internal) — встроенный RC-генератор тактового сигнала работающий на частоте 8 МГц. Его калибруют на заводе, но при изменении температуры частота может колебаться от 7,3 МГц до 8,7 МГц. При подаче питания он является источником тактового сигнала, далее программа может переключаться на другие источники.

**HSE** (сокр. High Speed External) — внешний генератор (обычно кварцевый или керамический резонатор), который подключаются к соответствующим ножкам МК.

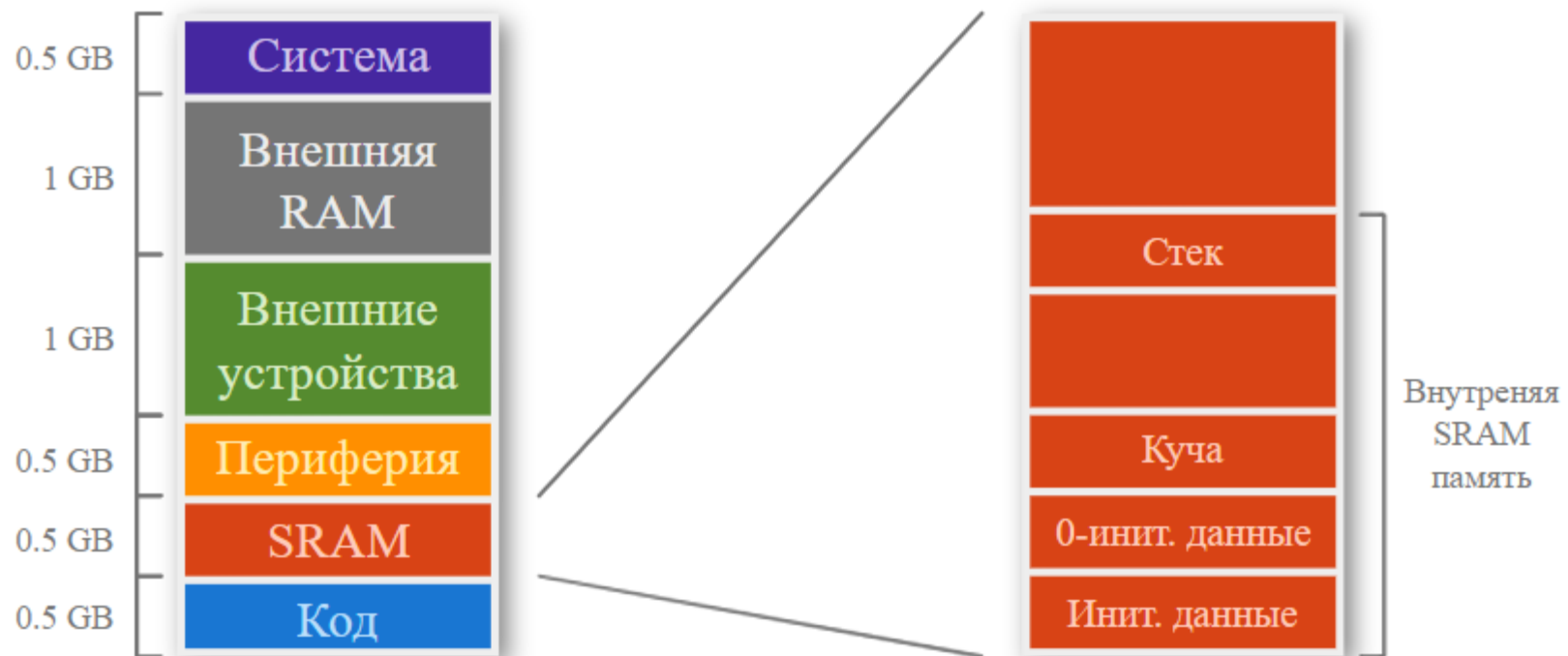
**PLL** (сокр. Phase-Locked Loop) — система фазовой автоподстройки частоты (ФАПЧ), позволяет умножать частоту HSI или HSE на множитель от 2 до 16. Частота поступающая на ФАПЧ делится пополам.



# Архитектура ядра



# Карта памяти





## Несколько слов о библиотеках

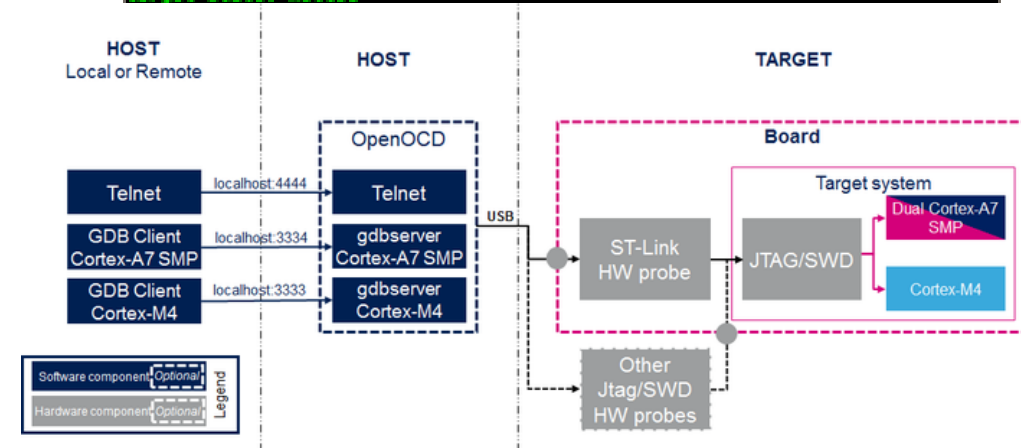


# Инструменты для разработки: два пути



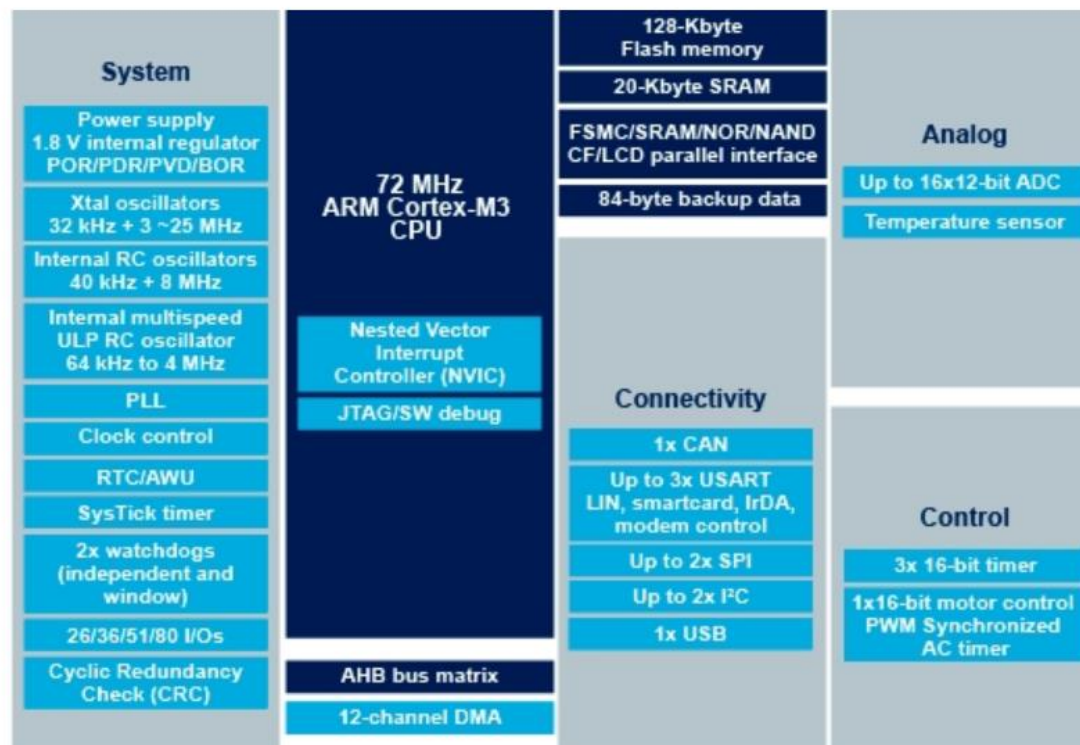
VS

```
Terminal
~ $ telnet localhost 4444
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^['.
Open On-Chip Debugger
> halt
target state: halted
target halted due to debug-request at 0xc1009437 in protected mode
> reg EAX
EAX (/32): 0x00000000
> reg EIP
EIP (/32): 0xc1009437
> step
step done from EIP 0xc1009437 to 0xc1009430
target state: halted
```



# Используемый МК stm32f103rbt6

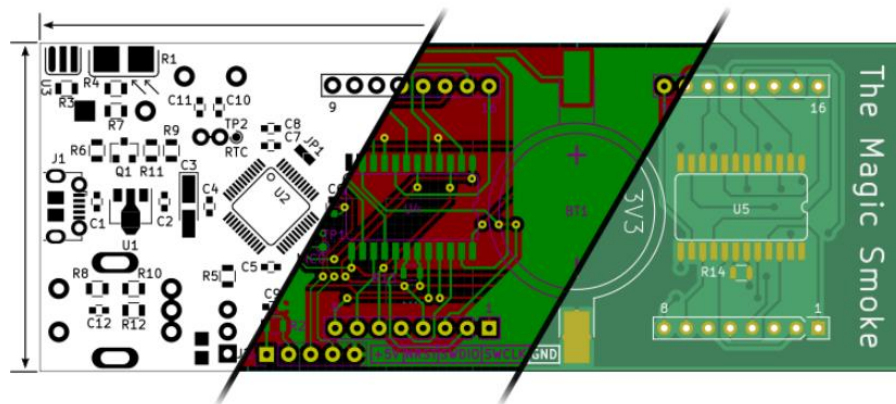
## Circuit Diagram



<https://www.st.com/en/microcontrollers-microprocessors/stm32f103rb.html>

Некоторый материал первой части данного курса был взят  
отсюда: <https://themagicsmoke.ru/courses/stm32/index.html>

## Курс «Штурмуем STM32»

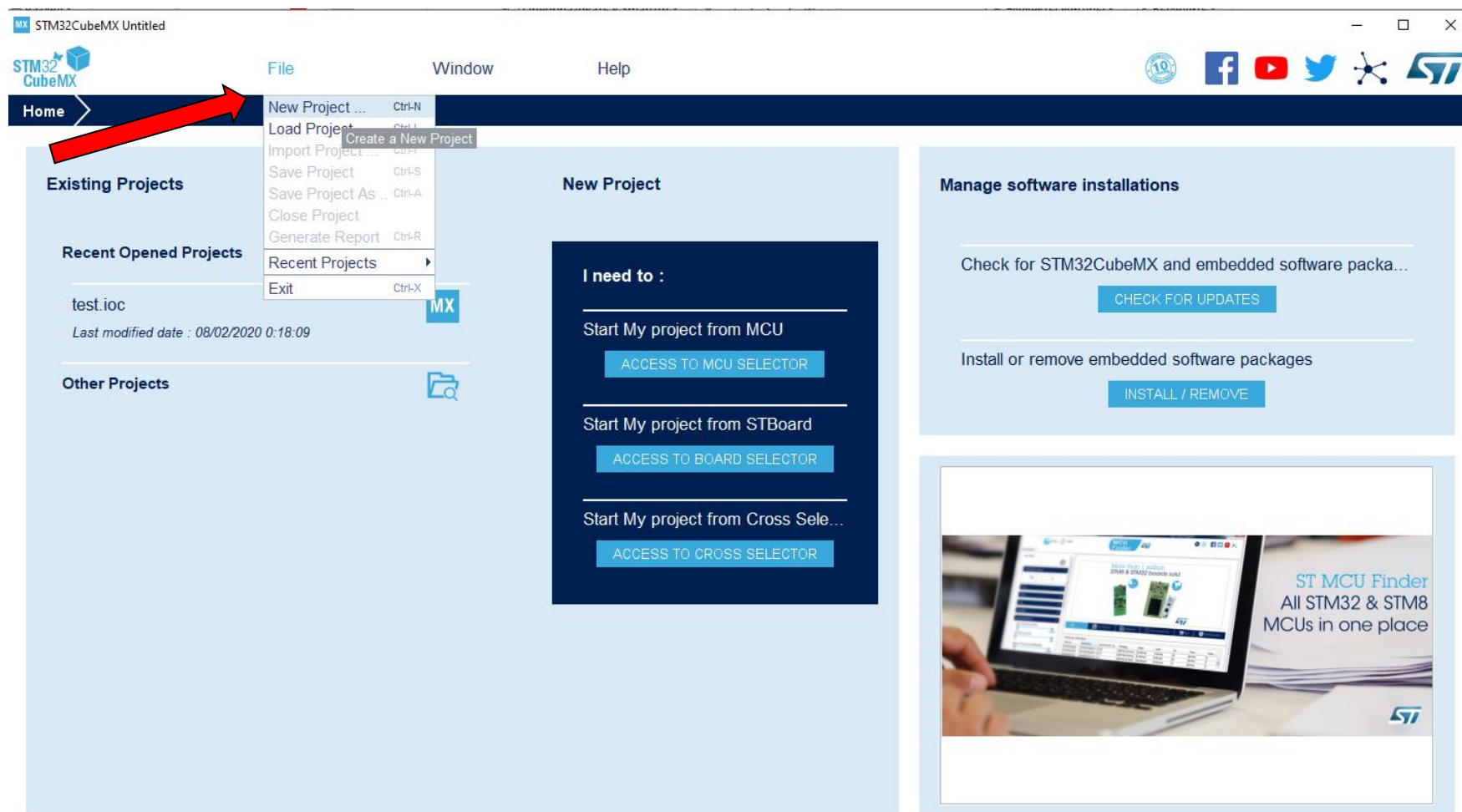


Данный курс является рефлексией и логическим завершением двухлетнего руководства кружком электроники на ИРИТ-Ртф, УрФУ (НЕС – Hardware Engineering Club, позже IMEN – I Am an Engineer). Курс в был прочитан студентам 3-го курса Радиотехнического Факультета УрФУ в весеннем семестре 2016 года ([отчёт](#)), а также в осеннем семестре 2016 года школьникам СУНЦ УрФУ ([отчёт](#)). После курс был значительно переделан, добавлены новые темы, видеоизменено устройство и опробован на десятиклассниках СУНЦ УрФУ осенью-зимой 2018 года. Заказать набор (3000 ₽ + доставка) можно связавшись с автором по [электронной почте](#). В дальнейшем появится магазин.

# Занятие №1

## ПРАКТИКА

# Создаем первый проект





# Выбираем наш МК: STM32F103RB

MX New Project

MCU/MPU Selector | Board Selector | Cross Selector

MCU/MPU Filters

Part Number Search

stm32f103rb

Core

Series

Line

Package

Other

Price From 2.515 to 2.515

2.515

IO From 50 to 51

50 51

Eeprom = 0 (Bytes)

Flash = 128 (kBytes)

Ram = 20 (kBytes)

Freq. = 72 (MHz)

STM32F103RB

Mainstream Performance line, ARM Cortex-M3 MCU with 128 Kbytes Flash, 72 MHz CPU, motor control, USB and CAN

ACTIVE Active  
Product is in mass production

Unit Price for 10kU (US\$) : 2.515

TFBGA64

The STM32F103xx medium-density performance line family incorporates the high-performance ARM Cortex-M3 32-bit RISC core operating at a 72 MHz frequency, high-speed embedded memories (Flash memory up to 128 Kbytes and SRAM up to 20 Kbytes), and an extensive range of enhanced I/Os and peripherals connected to two APB buses. All devices offer two 12-bit ADCs, three general purpose 16-bit timers plus one PWM timer, as well as standard and advanced communication interfaces: up to two I2Cs and SPIs, three USARTs, an USB and a CAN.

The devices operate from a 2.0 to 3.6 V power supply. They are available in both the 40 to +85 C temperature range and the 40 to +105 C extended temperature range. A comprehensive set of power-saving mode allows the design of low-power applications.

The STM32F103xx medium-density performance line family includes devices in six different package types: from 36 pins to 100 pins. Depending on the device chosen, different sets of peripherals are included, the description below gives an overview of the complete range of peripherals proposed in this family.

These features make the STM32F103xx medium-density performance line microcontroller family suitable for a wide range of applications such as

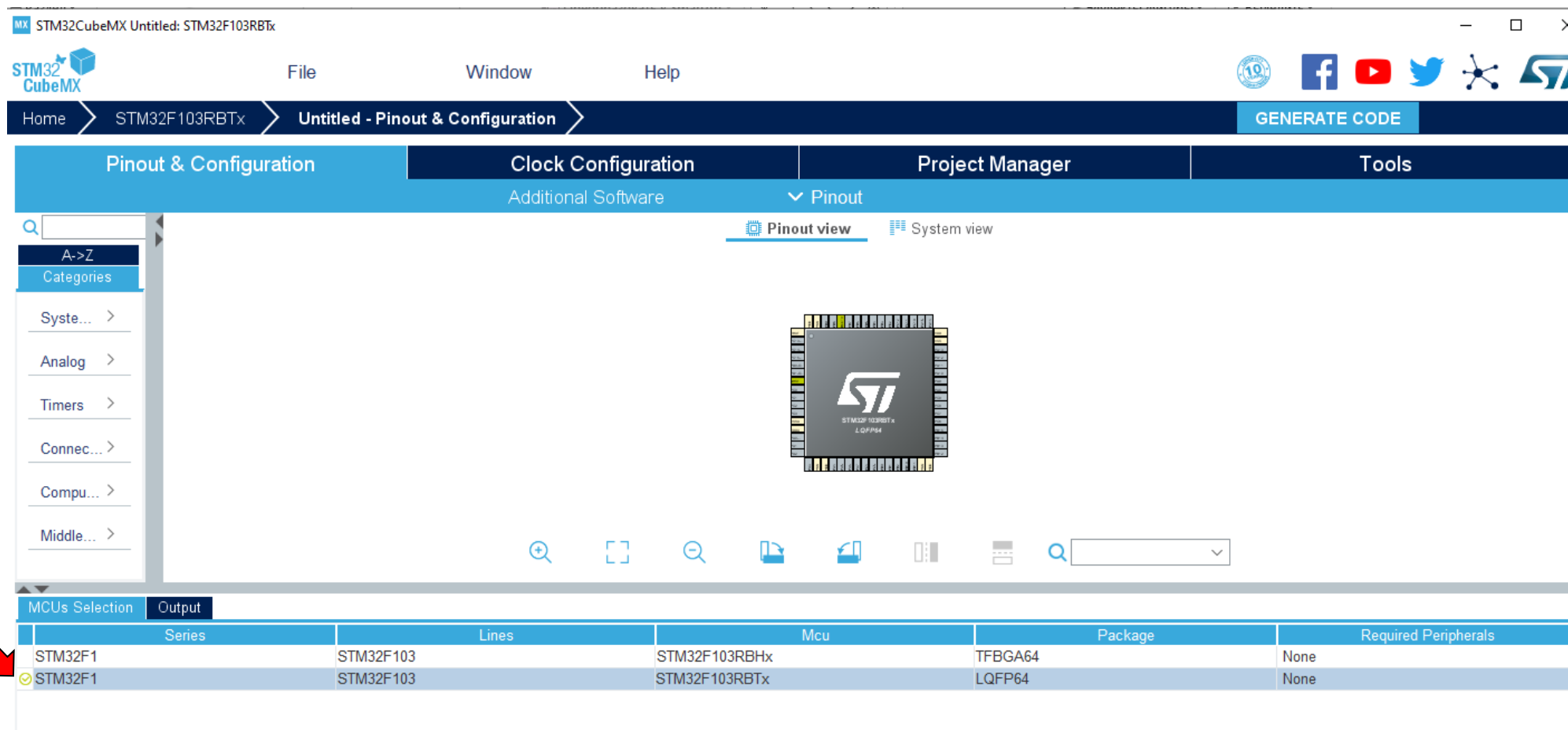
MCU/MPUs List: 2 items

Display similar items

	Reference	Marketing St.	Unit Price for 10kU	Board	Package	Flash	RAM	IO	Freq.	GFX Sc.	DES/TDES
★	STM32F103RB	Active	2.515		TFBGA64	128 kByte...	20 kBytes	50	72 MHz	0.0	0
★	STM32F103RB	Active	2.515		LQFP64	128 kByte...	20 kBytes	51	72 MHz	0.0	0

Export

# Выбираем тип корпуса

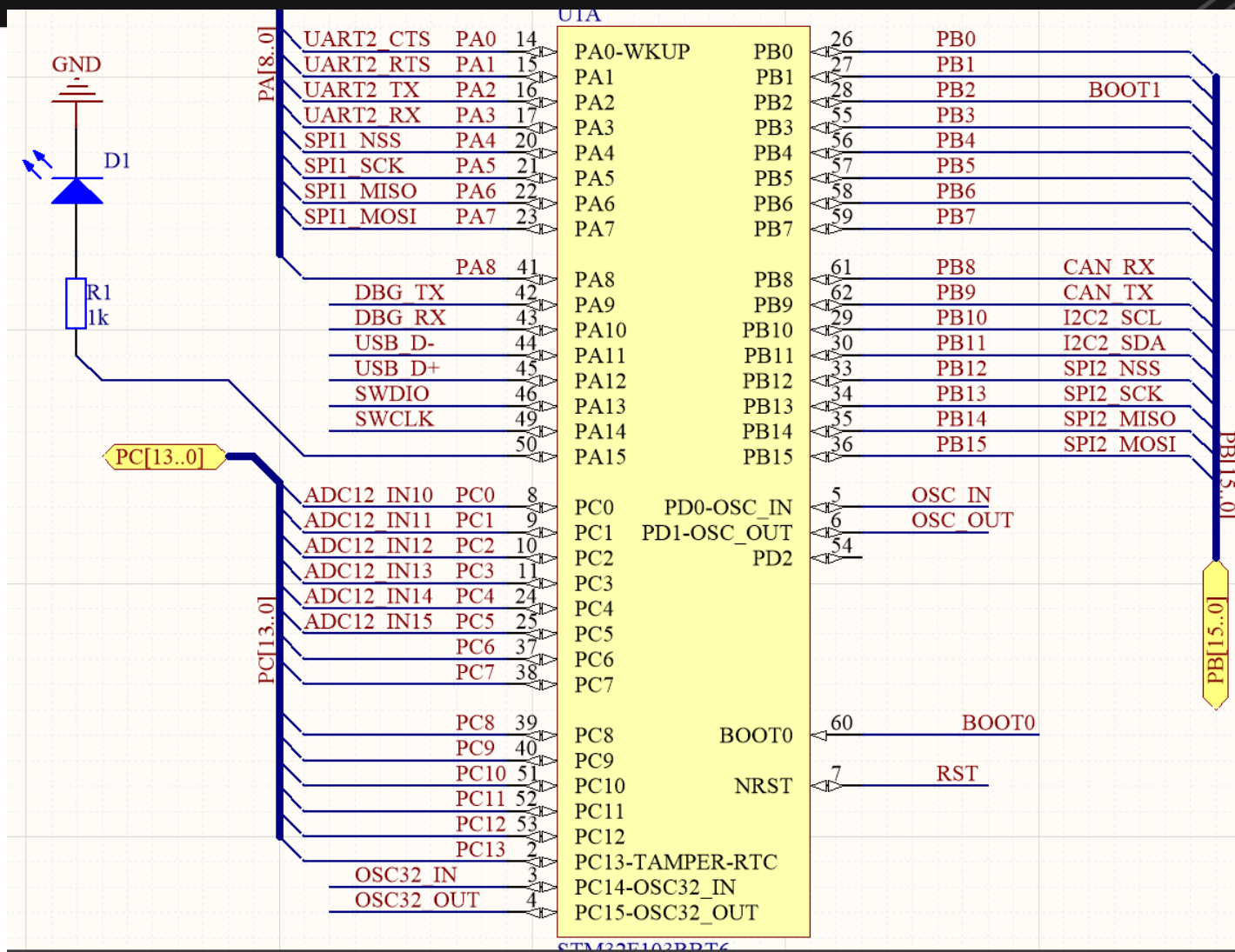


The screenshot shows the STM32CubeMX software interface. The top menu bar includes File, Window, and Help. The main window is titled 'STM32CubeMX Untitled: STM32F103RBTx'. The 'Pinout & Configuration' tab is active, showing a 3D model of the STM32F103RBTx microcontroller. A red arrow points to the 'MCUs Selection' table at the bottom of the interface.

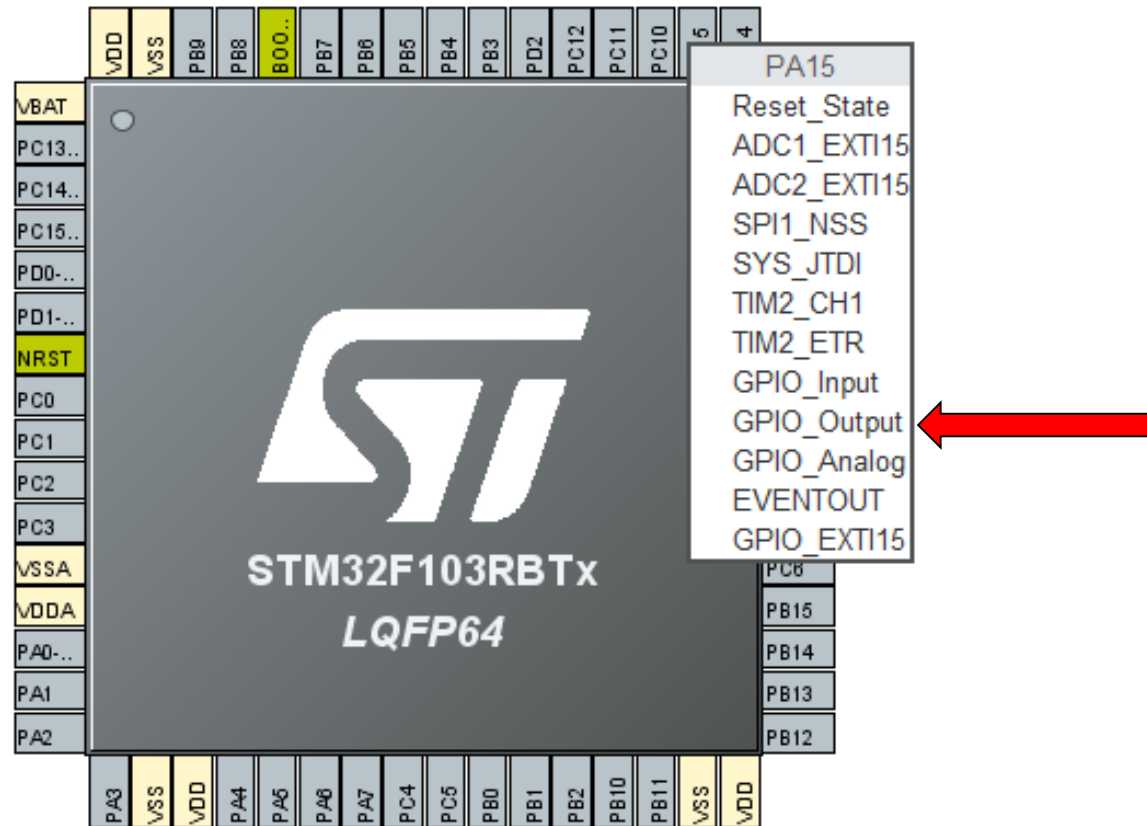
	Series	Lines	Mcu	Package	Required Peripherals
	STM32F1	STM32F103	STM32F103RBHx	TFBGA64	None
<input checked="" type="radio"/>	STM32F1	STM32F103	STM32F103RBTx	LQFP64	None



# Настраиваем выводы микроконтроллера опираясь на схему платы



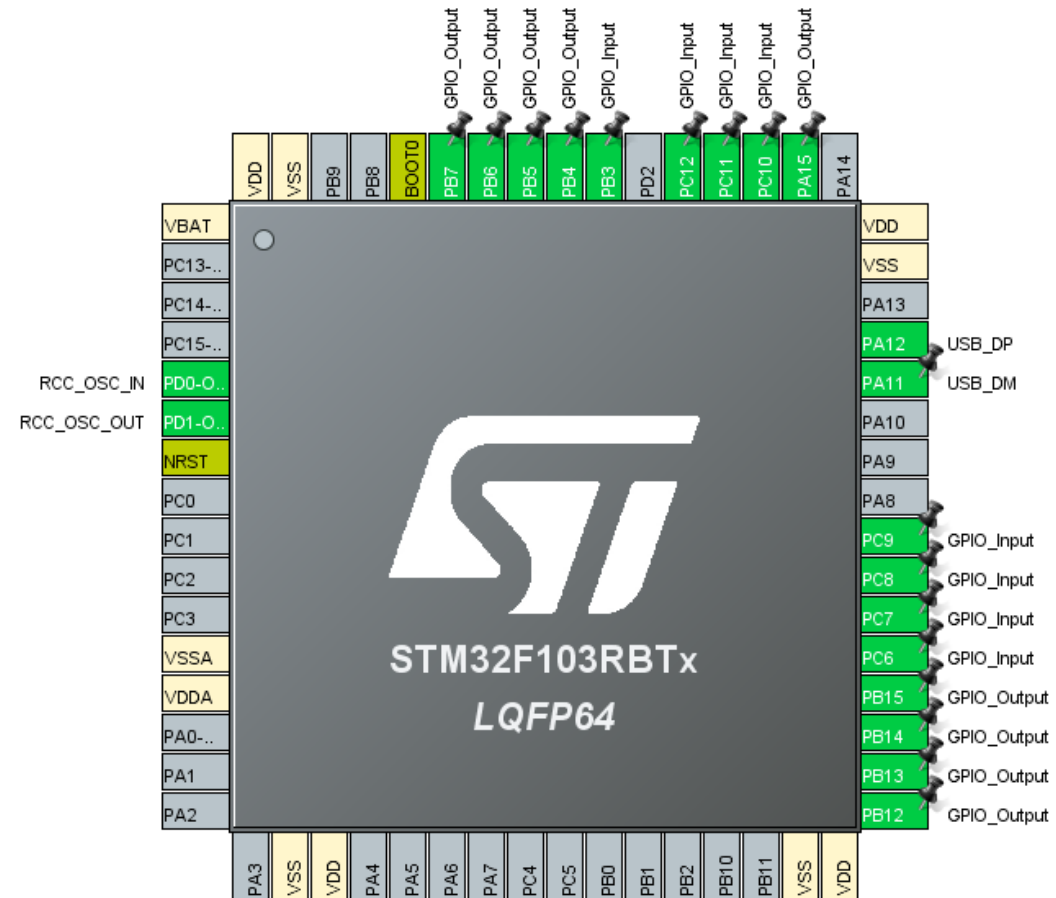
# Настроим ножку PA15 на выход (к ней подключен светодиод Status)



По аналогии можем настроить выводы, к которым подключены остальные светодиоды и кнопки

LED0	PB13
LED1	PB12
LED2	PB14
LED3	PB15
LED4	PB4
LED5	PB5
LED6	PB6
LED7	PB7

BTN1	PC11
BTN2	PC12
BTN3	PB3
BTN4	PC10
BTN5	PC8
BTN6	PC7
BTN7	PC6
BTN8	PC9



# Настраиваем модуль GPIO

The screenshot displays the STM32CubeMX software interface for configuring the STM32F103RBTx microcontroller. The 'System Core' list on the left shows 'GPIO' selected. The 'GPIO Mode and Configuration' table lists various pins, with 'PC8' highlighted. The 'PC8 Configuration' section at the bottom shows 'Input mode' and 'No pull-up and no pull-down' selected.

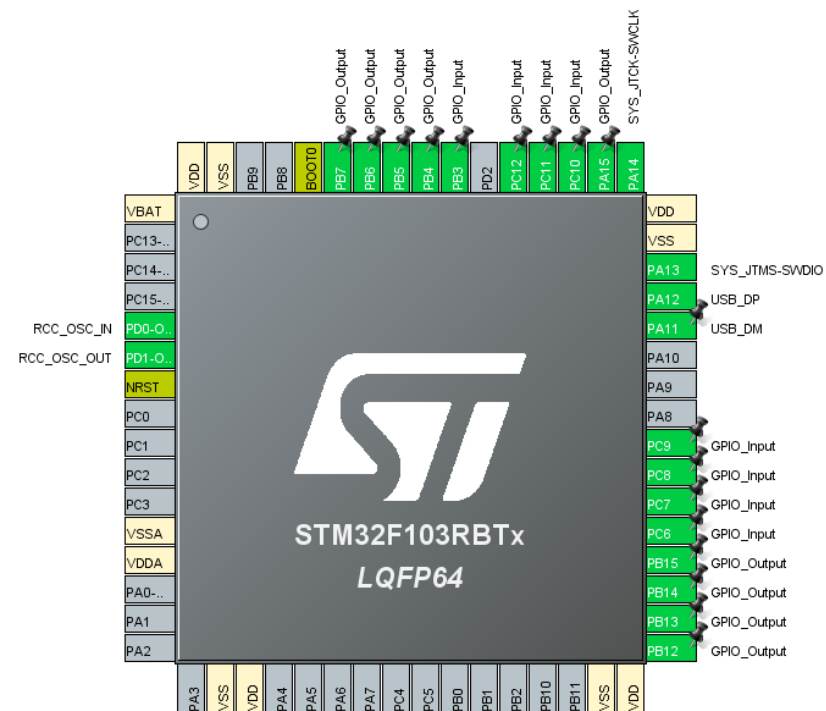
Pin Name	Signal on Pin	GPIO output	GPIO mode	GPIO Pull-up	Maximum ou...	User Label	Modified
PA15	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB3	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PB4	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB5	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB6	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB7	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB12	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB13	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB14	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB15	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PC6	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC7	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC8	n/a	n/a	Input mode	No pull-up an...	n/a		<input checked="" type="checkbox"/>
PC9	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC10	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC11	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC12	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>

PC8 Configuration :

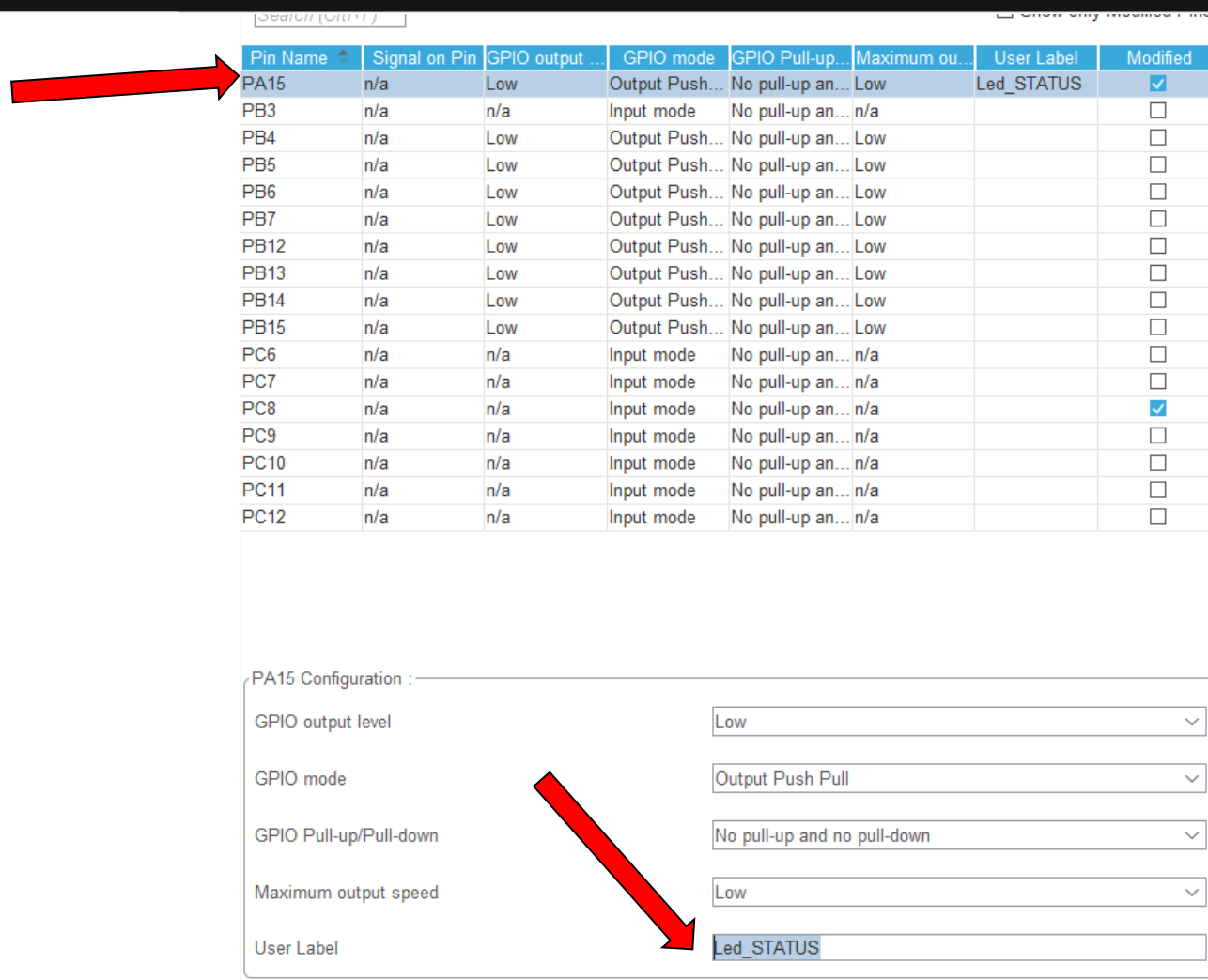
GPIO mode: Input mode

GPIO Pull-up/Pull-down: No pull-up and no pull-down

User Label:



## Для удобства выводам МК можно задать человеческие названия



Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA15	n/a	Low	Output Push...	No pull-up an...	Low	Led_STATUS	<input checked="" type="checkbox"/>
PB3	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PB4	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB5	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB6	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB7	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB12	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB13	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB14	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PB15	n/a	Low	Output Push...	No pull-up an...	Low		<input type="checkbox"/>
PC6	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC7	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC8	n/a	n/a	Input mode	No pull-up an...	n/a		<input checked="" type="checkbox"/>
PC9	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC10	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC11	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>
PC12	n/a	n/a	Input mode	No pull-up an...	n/a		<input type="checkbox"/>

PA15 Configuration :

GPIO output level: Low

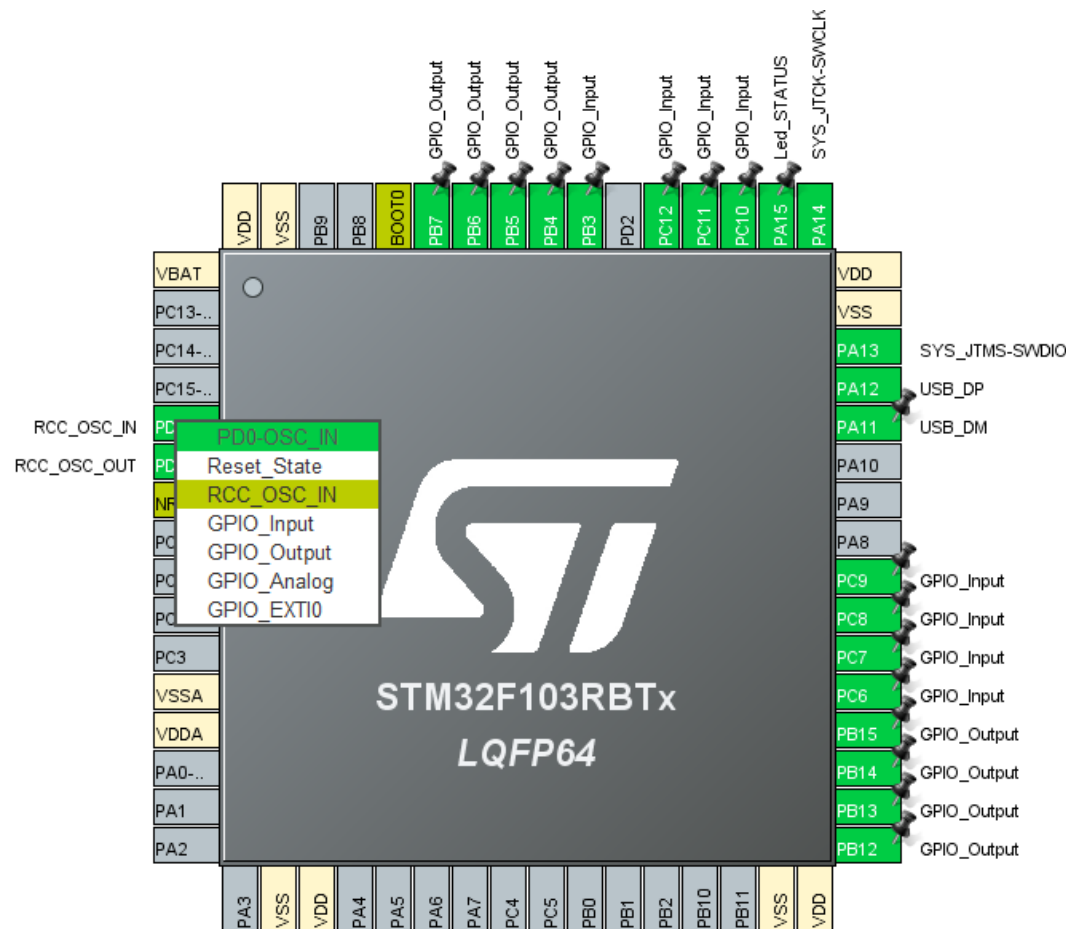
GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

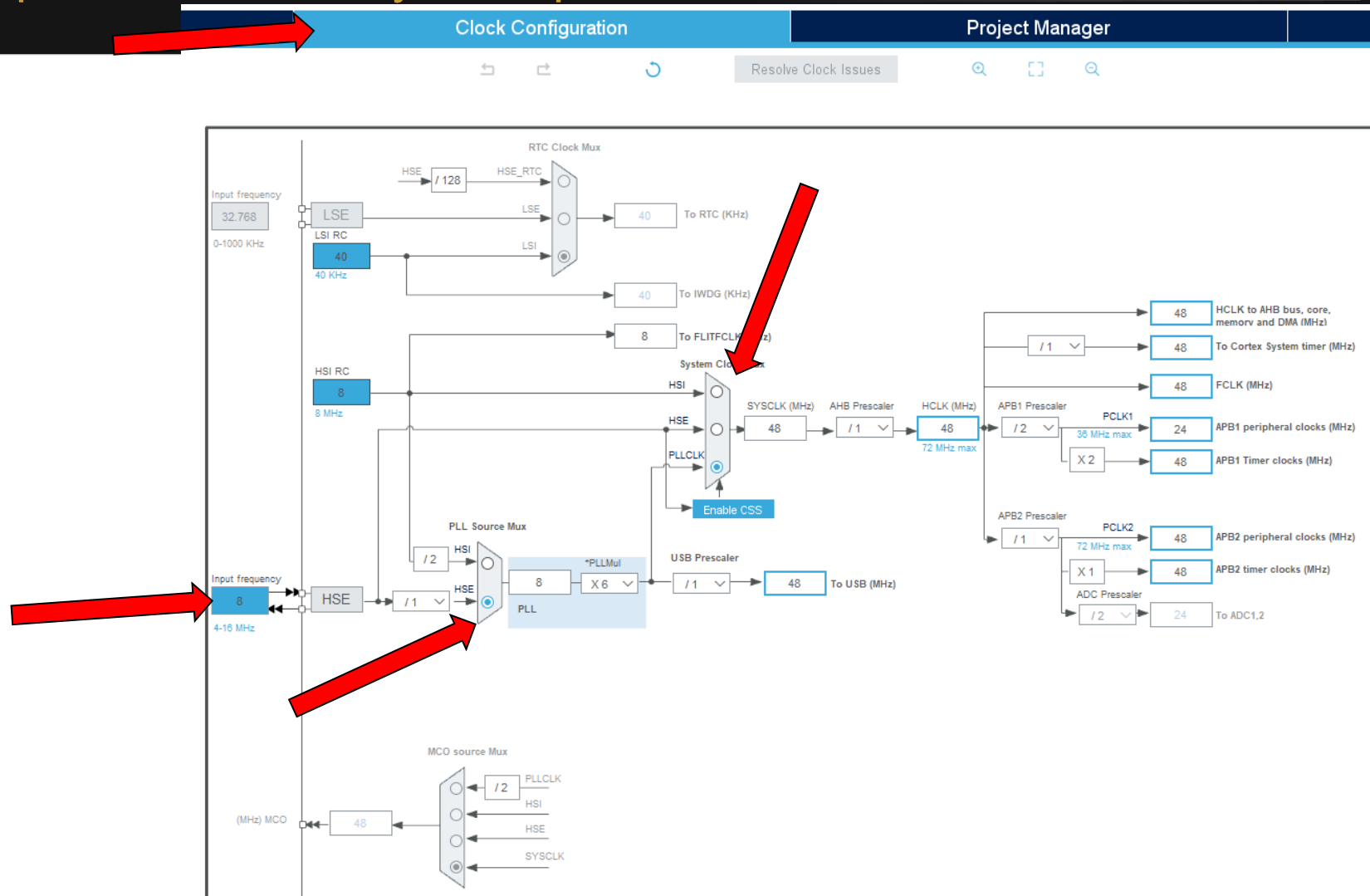
Maximum output speed: Low

User Label: Led\_STATUS

# Не забываем подключить кварцевый резонатор



# Настраиваем систему тактирования



# Заканчиваем с настройками проекта и генерируем заготовку

home > STM32F103RBTx > test.ioc - Project Manager

Pinout & Configuration | Clock Configuration | **Project Manager**

**Project**

Project Name  
test

Project Location  
D:\files\work\soft\stm32

Application Structure  
Basic ☐ Do not generate the main()

Toolchain Folder Location  
D:\files\work\soft\stm32\test\

Toolchain / IDE  
TrueSTUDIO ☒ Generate Under Root

**Code Generator**

**Advanced Settings**

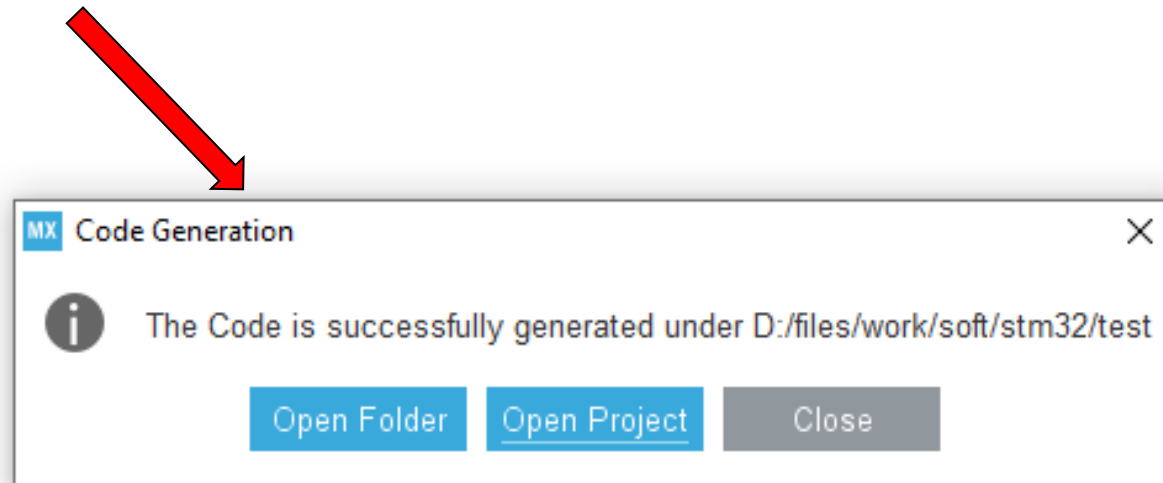
Linker Settings  
Minimum Heap Size 0x200  
Minimum Stack Size 0x400

Mcu and Firmware Package  
Mcu Reference  
STM32F103RBTx

Firmware Package Name and Version  
STM32Cube\_FW\_F1 V1.8.0

☒ Use Default Firmware Location  
C:/Users/home\_station/STM32Cube/Repository/STM32Cube\_FW\_F1\_V1.8.0 Browse





# CubeMX сгенерировал для нас проект

The screenshot displays the Atollic TrueSTUDIO for STM32 IDE. The main window shows a C file named `main.c` with the following content:

```
1  /* USER CODE BEGIN Header */
2  /**
3   * @file
4   * @brief : Main program body
5   * @attention
6   *
7   * <h2><center>copy; Copyright (c) 2020 STMicroelectronics.
8   * All rights reserved.</center></h2>
9   *
10   * This software component is licensed by ST under Ultimate Liberty license
11   * SLA0044, the "License"; You may not use this file except in compliance with
12   * the License. You may obtain a copy of the license at:
13   * www.st.com/SLA0044
14   *
15   */
16  /* USER CODE END Header */
17
18  /* Includes -----*/
19  #include "main.h"
20  #include "usb_device.h"
21
22  /* Private includes -----*/
23  /* USER CODE BEGIN Includes */
24
25  /* USER CODE END Includes */
26
27  /* Private typedef -----*/
28  /* USER CODE BEGIN PTD */
29
30  /* USER CODE END PTD */
31
32  /* Private define -----*/
33  /* USER CODE BEGIN PD */
34
35  /* USER CODE END PD */
36
37  /* Private macro -----*/
38  /* USER CODE BEGIN PM */
39
40  /* USER CODE END PM */
41
42  /* Private variables -----*/
43
44  /* USER CODE BEGIN Private variables */
45
```

The Project Explorer on the left shows the project structure, including `test`, `Binaries`, `Includes`, `Drivers`, `Middlewares`, `Src`, `startup`, `Debug`, `Release`, `inc`, `STM32F103RB_FLASH.ld`, `test.elf.launch`, and `test.ioc`.

The Outline view on the right shows the following functions:

- `main.h`
- `usb_device.h`
- `SystemClock_Config(void) : void`
- `MX_GPIO_Init(void) : void`
- `main(void) : int`
- `SystemClock_Config(void) : void`
- `MX_GPIO_Init(void) : void`
- `Error_Handler(void) : void`
- `assert_failed(uint8_t*, uint32_t) : void`

The Build Analyzer at the bottom right shows the memory usage for the `test.elf` file, dated 22.02.20 12:12.

Memory Regions	Memory Details	Start address	End address	Size	Free	Used	Usage (%)
RAM		0x20000000	0x20005000	20 KB	13,7 KB	6,3 KB	31,50%
FLASH		0x08000000	0x08020000	128 KB	112,66 KB	15,34 KB	11,98%

# Первая программа

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    /* USER CODE END WHILE */

    HAL_GPIO_TogglePin(Led_STATUS_GPIO_Port, Led_STATUS_Pin);
    HAL_Delay(1000);

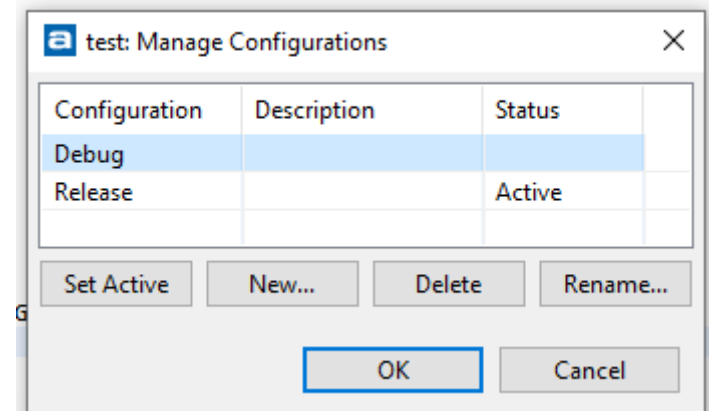
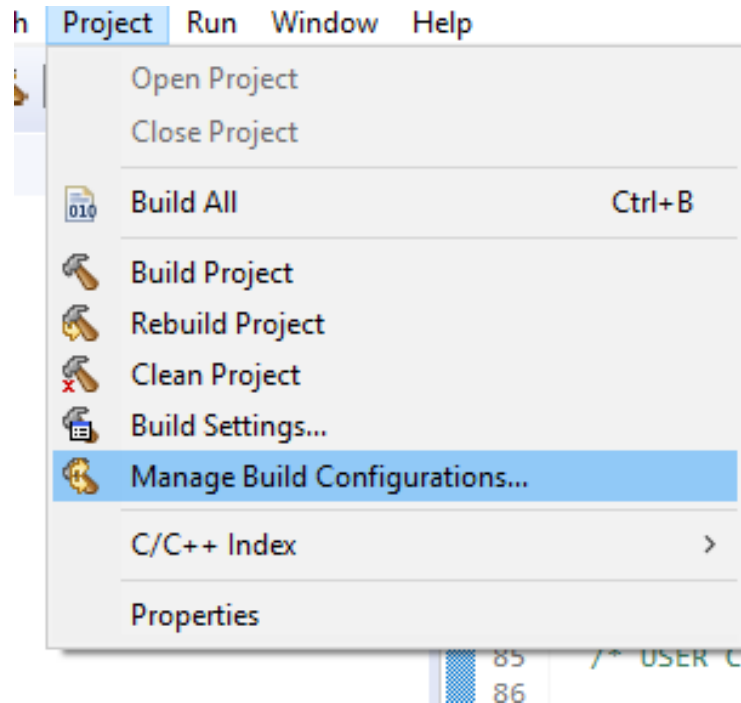
    /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```

## Как работать с кодом?

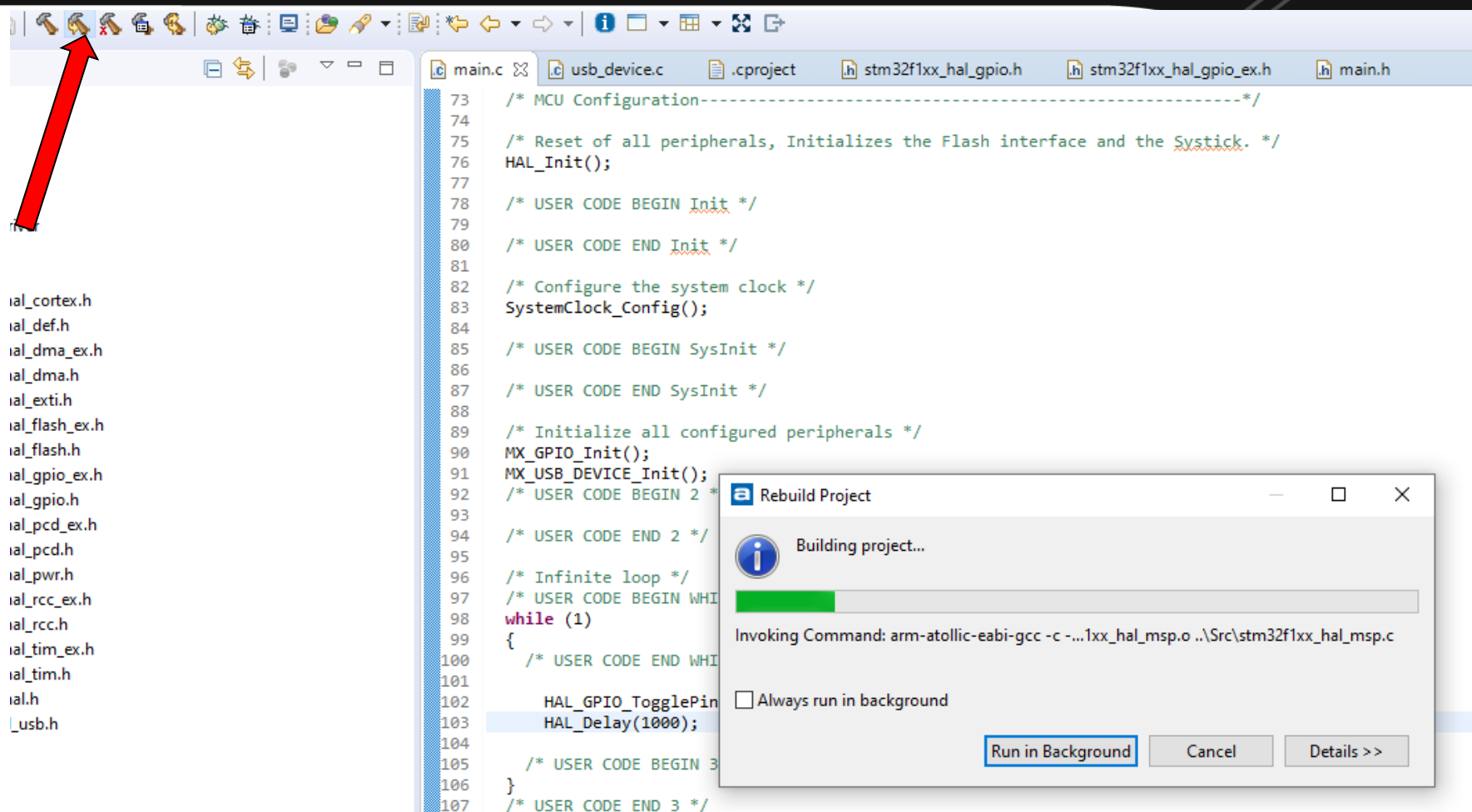
**Не удаляйте комментарии, созданные CubeMX и пишите ваш код между ними (между begin и end). В случае, если Вам придется регенерировать проект, CubeMX оставит без изменений то, что находится между комментариями, остальное будет удалено!**

**Не бойтесь заглядывать в файлы. В них можно найти ответы почти на все вопросы. Для того, чтобы узнать какие функции есть для работы с конкретным модулем и какие структуры или дефайны описаны, просто откройте соответствующий \*.h файл.**

# Режимы компиляции Debug и Release



# Компилируем



The screenshot shows an IDE window with a project named 'main.c'. The toolbar at the top contains various icons, with a red arrow pointing to the 'Build' icon (a hammer). The main editor displays the following C code:

```
73 /* MCU Configuration-----*/
74
75 /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
76 HAL_Init();
77
78 /* USER CODE BEGIN Init */
79
80 /* USER CODE END Init */
81
82 /* Configure the system clock */
83 SystemClock_Config();
84
85 /* USER CODE BEGIN SysInit */
86
87 /* USER CODE END SysInit */
88
89 /* Initialize all configured peripherals */
90 MX_GPIO_Init();
91 MX_USB_DEVICE_Init();
92 /* USER CODE BEGIN 2 */
93
94 /* USER CODE END 2 */
95
96 /* Infinite loop */
97 /* USER CODE BEGIN WHILE */
98 while (1)
99 {
100 /* USER CODE END WHILE */
101
102 HAL_GPIO_TogglePin
103 HAL_Delay(1000);
104
105 /* USER CODE BEGIN 3 */
106 }
107 /* USER CODE END 3 */
```

A 'Rebuild Project' dialog box is open, showing the command: 'arm-atollic-eabi-gcc -c -...1xx\_hal\_msp.o ..\Src\stm32f1xx\_hal\_msp.c'. The dialog also includes a checkbox for 'Always run in background' and buttons for 'Run in Background', 'Cancel', and 'Details >>'.

# Находим hex файл прошивки в соответствующей папке

Этот компьютер > Новый том (D:) > files > work > soft > stm32 > test > Release >				
Имя	Дата изменения	Тип	Размер	
Drivers	02.01.2020 21:53	Папка с файлами		
Middlewares	02.01.2020 21:53	Папка с файлами		
Src	22.02.2020 12:31	Папка с файлами		
startup	22.02.2020 12:31	Папка с файлами		
stm32flash.exe	10.02.2016 19:14	Приложение	247 КБ	
test.elf	22.02.2020 12:31	Файл "ELF"	148 КБ	
test.hex	22.02.2020 12:31	Файл "HEX"	44 КБ	
test.list	22.02.2020 12:31	Файл "LIST"	251 КБ	
test.map	22.02.2020 12:31	Файл "MAP"	144 КБ	

# Прошиваем

The screenshot shows the STM32CubeProgrammer application window. The 'Memory & File edition' tab is active. A file named 'test.hex' is loaded into the 'Device memory' section. The address is set to 0x8000000, size to 0x3DA4, and data width to 32-bit. A red arrow points to the 'test.hex' file name, and another red arrow points to the 'Download' button. The 'ST-LINK' configuration panel on the right shows 'Port' set to SWD and 'Connect' button. The 'Log' panel at the bottom shows a series of error messages related to ST-LINK connection failures.

STM32CubeProgrammer

Memory & File edition

Device memory: test.hex

Address: 0x8000000 Size: 0x3DA4 Data width: 32-bit Download

Address	0	4	8	C	ASCII
0x08000000	20005000	08002B25	08002655	08002657	.P. %+. .U&. .W&. .
0x08000010	08002659	0800265B	0800265D	00000000	Y&. . [&. .] &. . . . .
0x08000020	00000000	00000000	00000000	0800265F	. . . . . _&. .
0x08000030	08002661	00000000	08002663	08002665	a&. . . . . c&. . e&. .
0x08000040	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x08000050	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x08000060	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x08000070	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x08000080	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x08000090	08002669	08002B6D	08002B6D	08002B6D	i&. . m+ . . m+ . . m+ . .
0x080000A0	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x080000B0	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .
0x080000C0	08002B6D	08002B6D	08002B6D	08002B6D	m+ . . m+ . . m+ . . m+ . .

Log

12:33:45 : STM32CubeProgrammer API v2.2.0  
12:34:53 : Read File: D:\files\work\soft\stm32\test\Release\test.hex  
12:34:53 : Number of segments: 1  
12:34:53 : segment[0]: address= 0x8000000, size= 0x3DA4  
12:34:56 : ST-LINK error (DEV\_NO\_STLINK)  
12:34:56 : Error: Problem occurred while trying to connect  
12:35:03 : ST-LINK error (DEV\_NO\_STLINK)  
12:35:03 : ST-LINK error (DEV\_NO\_STLINK)  
12:35:03 : ST-LINK error (DEV\_NO\_STLINK)  
12:35:04 : ST-LINK error (DEV\_NO\_STLINK)  
12:35:06 : ST-LINK error (DEV\_NO\_STLINK)

ST-LINK configuration

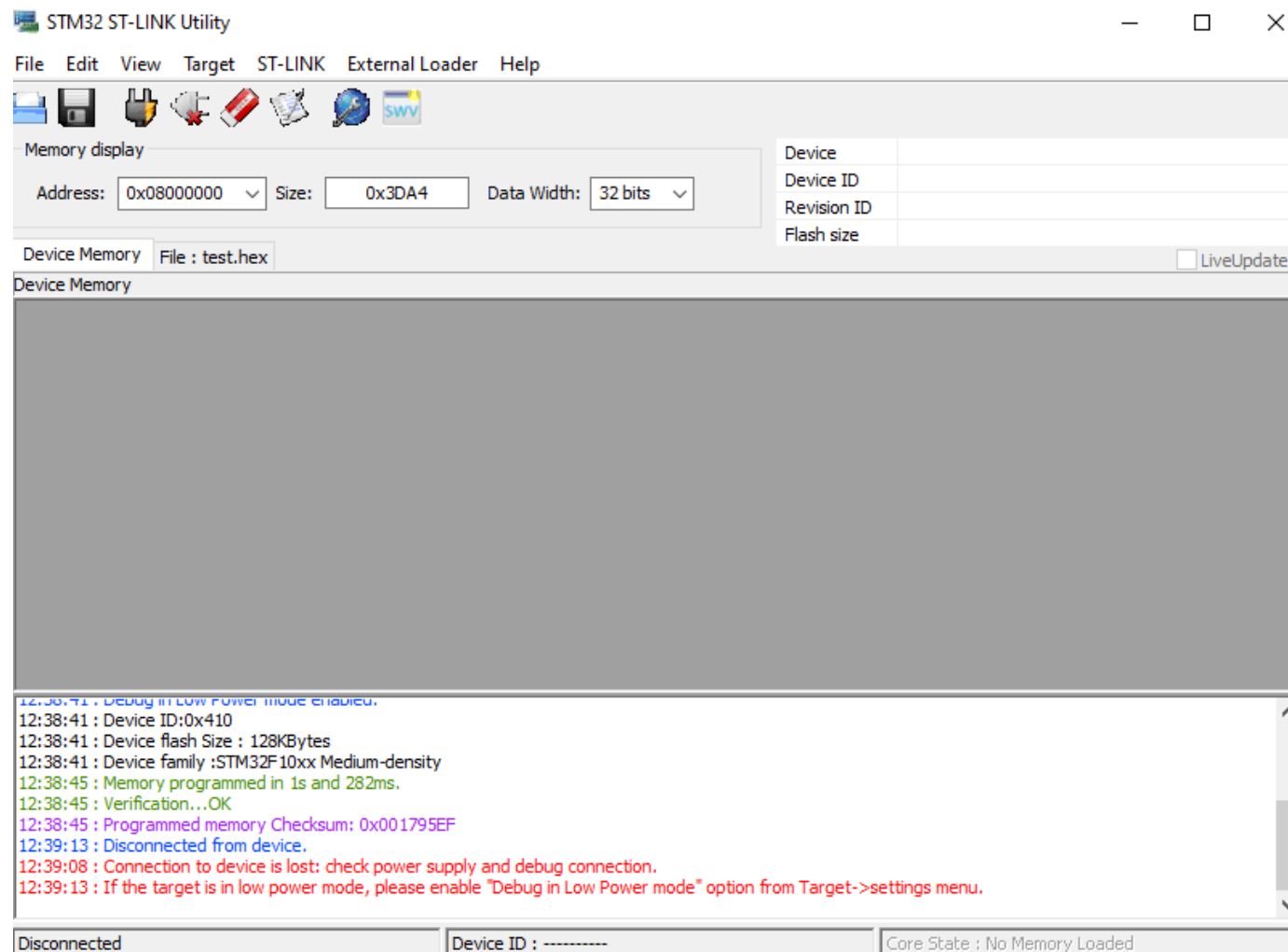
Port: SWD  
Frequency (kHz):  
Mode: Normal  
Access port: 0  
Reset mode: Software reset  
Shared: Disabled  
External loader:  
Target voltage:  
Firmware version:  
Firmware upgrade

Device information

Device:  
Type:  
Device ID:  
Flash size:  
CPU:



# Альтернативная программа для прошивки





STM & XILINX.

**Спасибо за внимание,  
спасибо за старания!**

**GitHub**

[https://github.com/v-crys/course\\_stm32\\_fpga](https://github.com/v-crys/course_stm32_fpga)