

Curso de Programação de Microcontroladores STM32

Felipe C. Gehrke
Ricardo Elias D. Castilhos

AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

AGENDA

- **Revisão do curso anterior;**
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

Revisão do Curso Anterior

- Quem fez ?
- Qual foi o conteúdo ?

Formato das Aulas

Explicação

Dúvidas

Prática

AGENDA

- Revisão do curso anterior;
- **GIT (Github);**
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

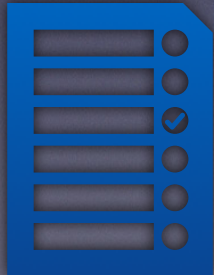
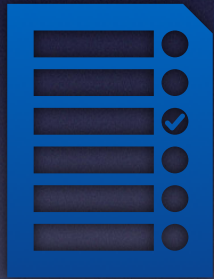
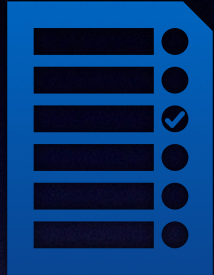
GIT

- Sistema de versionamento de código distribuido;
- Desenvolvido por Linus Torvalds;
- Software Open Source;



Funcionamento

Código



Versões




Repositório



Versões

Commits on Apr 3, 2018

Merge pull request #126 from mbrickn/patch-1 ...
 ascheink committed on 3 Apr

Verified



51779e9



update index



dmil committed on 3 Apr



8f4aa07



add to index



dmil committed on 3 Apr



c65956f



update README



dmil committed on 3 Apr



f420011



update README



dmil committed on 3 Apr



a64bad0



Commits on Mar 30, 2018

update README



dmil committed on 30 Mar



6100bee



rename file



dmil committed on 30 Mar



f6098ea



add special-elections data



dmil committed on 30 Mar



313343c



git clone

Clona um repositório do servidor

git add

Adiciona os arquivos atuais ao versionamento

git commit

Criar uma nova versão na máquina local

git push

Envia a versão para o servidor

Download do Git

<https://git-scm.com>

Github

[Features](#)[Business](#)[Explore](#)[Marketplace](#)[Pricing](#)[Sign in](#) or [Sign up](#)

Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside millions of other developers.

Username

Email

Password

Use at least one letter, one numeral, and seven characters.

[Sign up for GitHub](#)

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy statement](#). We'll occasionally send you account related emails.

Github Student Pack

<https://education.github.com/pack>

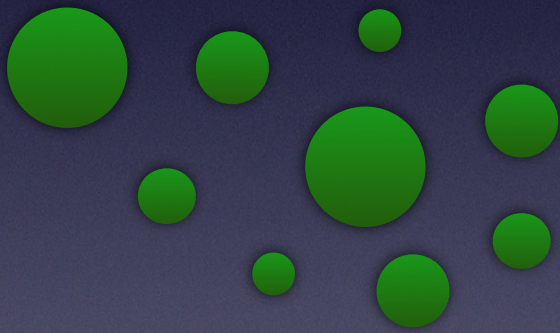
Mãos à obra ?

AGENDA

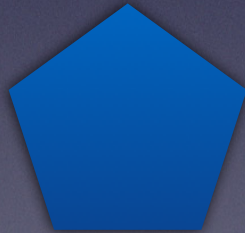
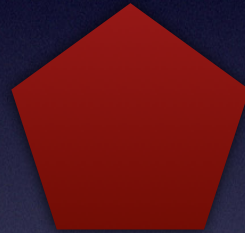
- Revisão do curso anterior;
- GIT (Github);
- **Conceitos de programação;**
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

Fases do Projeto

Ideias



Conceito



O que ?



Forma



Como ?

Tipos de Sistemas Embarcados

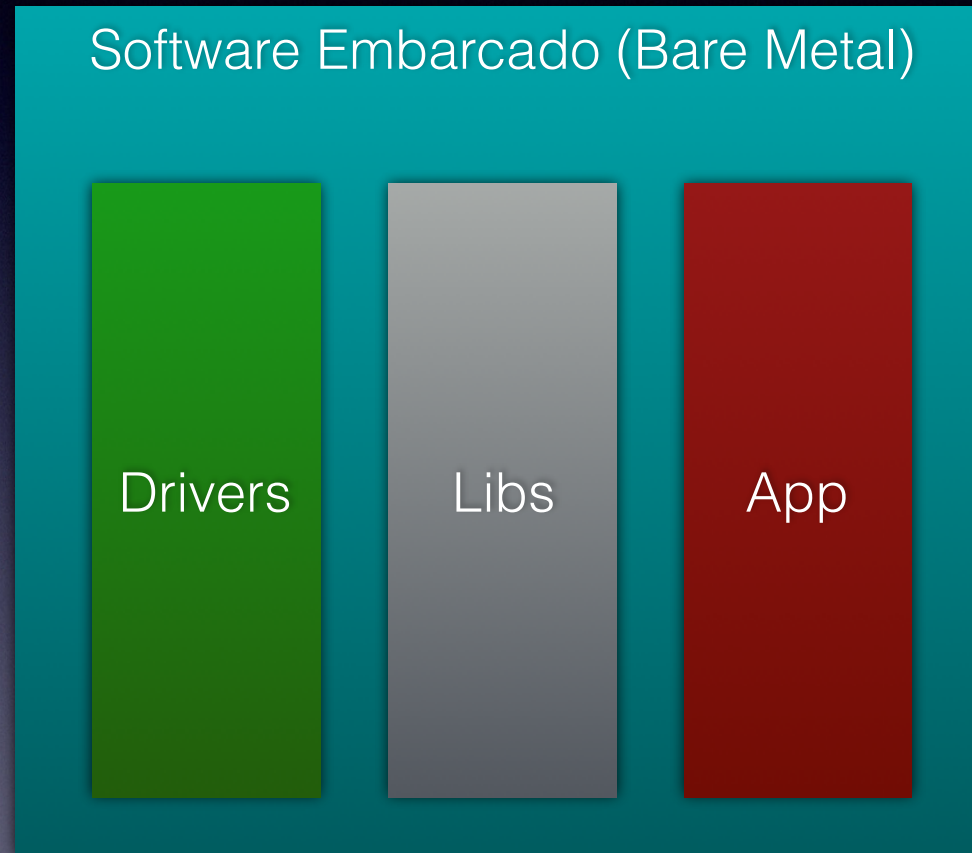


Bare-Metal



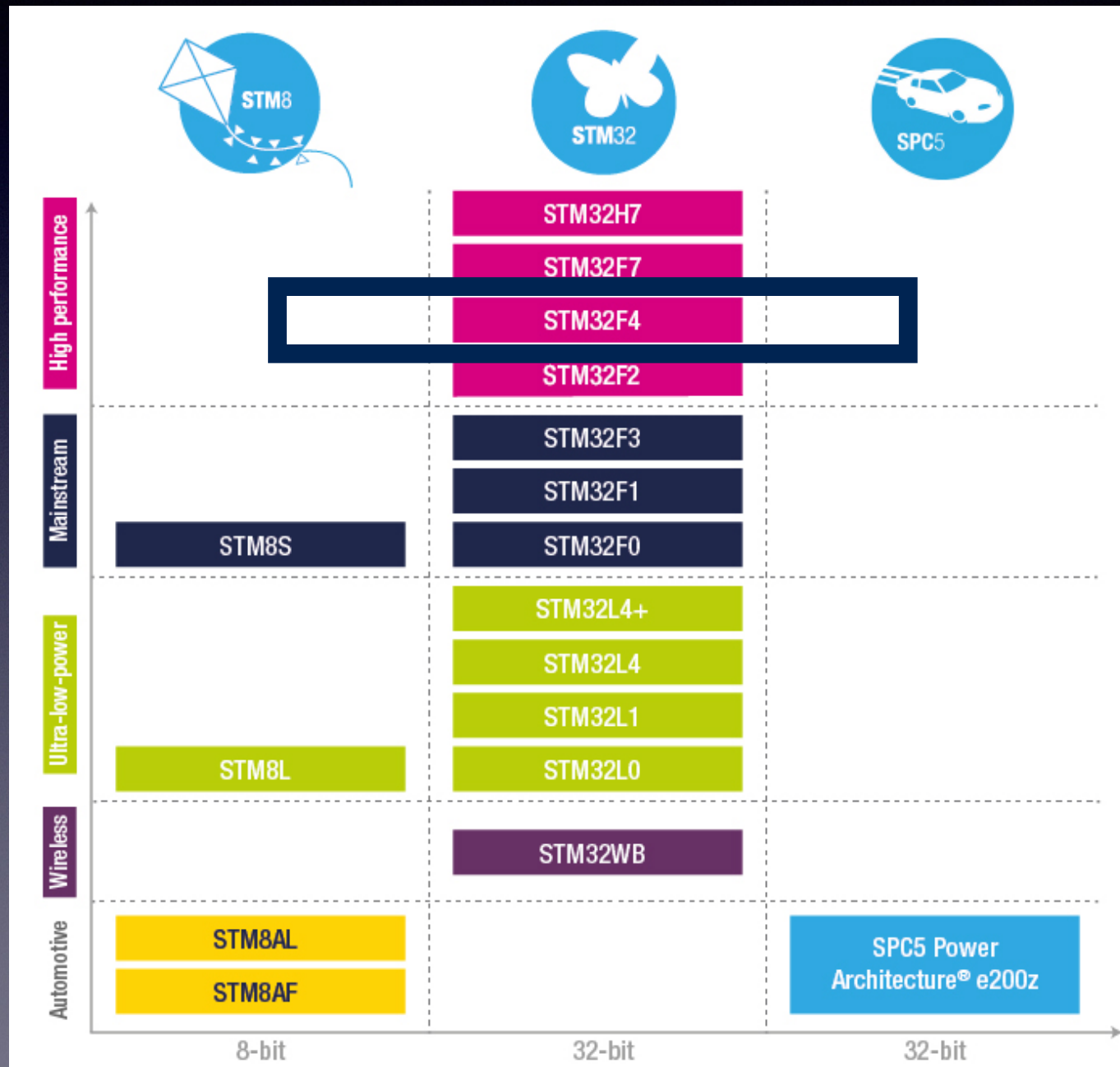
RTOS

Exemplo de Estrutura da Aplicação



Familias de Microcontroladores ST

STM32F407VGT6



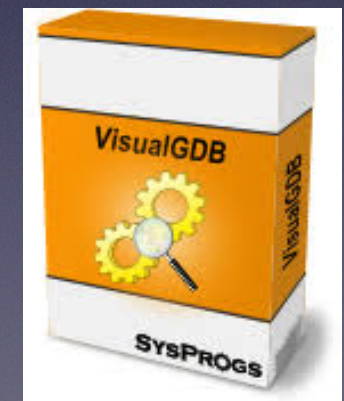
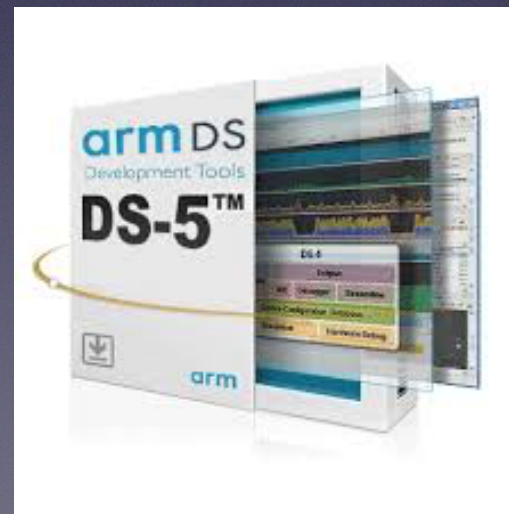
AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- **STM32CubeMX;**
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- **IDE**;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

IDE's



IDE - Atollic

STM32_workspace_9.0 - C/C++ - ad1/Src/main.c - Atollic TrueSTUDIO for STM32

File Edit Source Refactor View Navigate Search Project Run Window Help

Project Explorer

- ad1
 - Binaries
 - Includes
 - Drivers
 - Inc
 - adc.h
 - gpio.h
 - main.h
 - stm32f4xx_hal_conf.h
 - stm32f4xx_it.h
 - Src
 - adc.c
 - gpio.c
 - main.c
 - stm32f4xx_hal_msp.c
 - stm32f4xx_it.c
 - system_stm32f4xx.c
 - startup
 - Debug
 - ad1.elf.launch
 - ad1.ioc
 - mx.scratch
 - STM32F407VG_FLASH.ld
- ad2
- ad3
- Piscaled

main.c

```
84  /* USER CODE BEGIN Init */
85
86  /* USER CODE END Init */
87
88  /* Configure the system clock */
89  SystemClock_Config();
90
91  /* USER CODE BEGIN SysInit */
92
93  /* USER CODE END SysInit */
94
95  /* Initialize all configured peripherals */
96  MX_GPIO_Init();
97  MX_ADC1_Init();
98  /* USER CODE BEGIN 2 */
99
100 /* USER CODE END 2 */
101
102 /* Infinite loop */
103 /* USER CODE BEGIN WHILE */
104 while (1)
105 {
106     HAL_ADC_Start(&hadc1);
107
108     if (HAL_ADC_PollForConversion(&hadc1, 100) == HAL_OK)
109     {
110         value = HAL_ADC_GetValue(&hadc1);
111         printf("Value: %d\n", value);
112
113         if (value > 1024 && value < 2048)
114             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, 1);
115         else
116             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, 0);
117
118         if (value > 2048 && value < 3072)
119         {
120             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, 1);
121             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, 1);
122         }
123         else
124             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, 0);
125
126         if (value > 3072 && value < 4096)
```

Outline

- main.h
- stm32f4xx_hal.h
- adc.h
- gpio.h
- SystemClock_Config(void) : void
- main(void) : int
- SystemClock_Config(void) : void
- _ErrorHandler(char, int) : void
- assert_failed(uint8_t*, uint32_t) : void

Build Analyzer

ad1.elf - /ad1/Debug - 5/4/18 10:51 PM

Memory Regions	Memory Details	Start address	End address	Size	Free	Used	Usage (%)
RAM		0x20000000	0x20020000	128 KB	126.27 KB	1.73 KB	1.35%
CCMRAM		0x10000000	0x10010000	64 KB	64 KB	0 B	0.00%
FLASH		0x08000000	0x08100000	1024 KB	1014.67 KB	9.33 KB	0.91%

Problems

```
<terminated> ad3.elf [Embedded C/C++ Application] ST-LINK
STM32 device: flash programming successful 0x8002018
Programming flash..
STM32 device: flash programming successful 0x8002020
Programming flash..
STM32 device: flash programming successful 0x8002024
Programming flash..
STM32 device: flash programming successful 0x8002028
Debugger connection lost.
Shutting down...
```

1:1

1:33 AM 5/5/2018

AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- **Systick**;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- AD;

System Timer (Systick)

- Geralmente utilizado pelo RTOS (Daí o nome, System Timer);
- Geralmente a interrupção é configurada em 1ms;
- Utilizado em aplicações Bare-Metal para definir bases de tempo;
- No caso da ST a utilização se dá por um callback;

AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- **Programação Bare-Metal (Sem RTOS);**
- GPIO;
- Debouncing;
- AD;

Bare Metal

- Aplicação roda no Big Loop (função main);
- Sincronização por bases de tempo;
- Interrupções devem ser curtas para não gerar 'overshot';
- O acesso ao hardware se dá por drivers;

Exemplo de Main Loop

Flags

F50ms = flag 50ms;
F100ms = flag 100ms;
FSerialInt = flag recepção serial;
FSPIInt = flag recepção SPI;

Big Loop

F50ms

Status_Led();
Keyboard_Read();

F100ms

Accelerometer_Filter();

FSerialInt

SerialDataProcess();

FSPIInt

SPIDataSend();

Estrutura de Modulo em C

Modulo em C

Header File (.h)

Interface da biblioteca

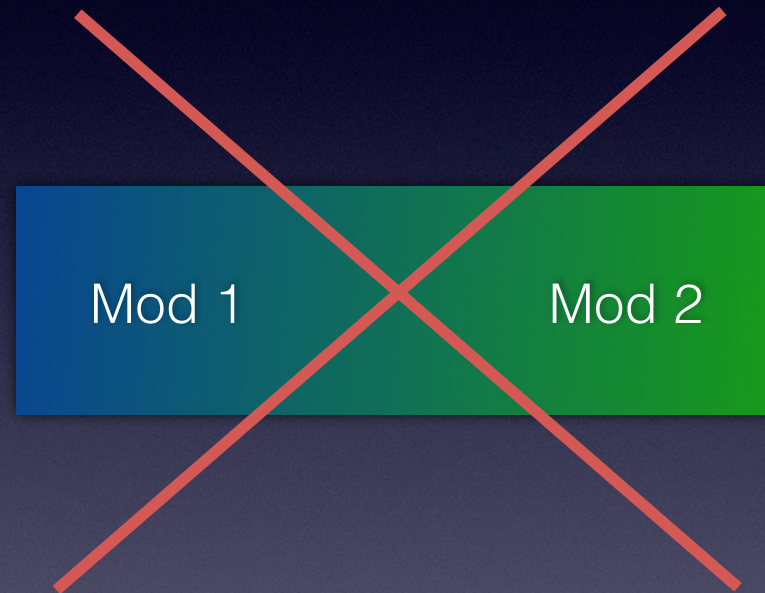
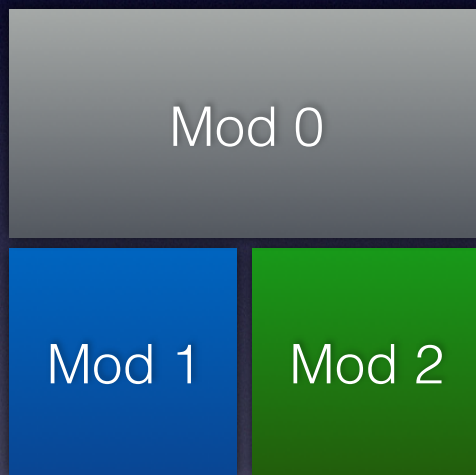
Code File (.c)

Implementação da biblioteca

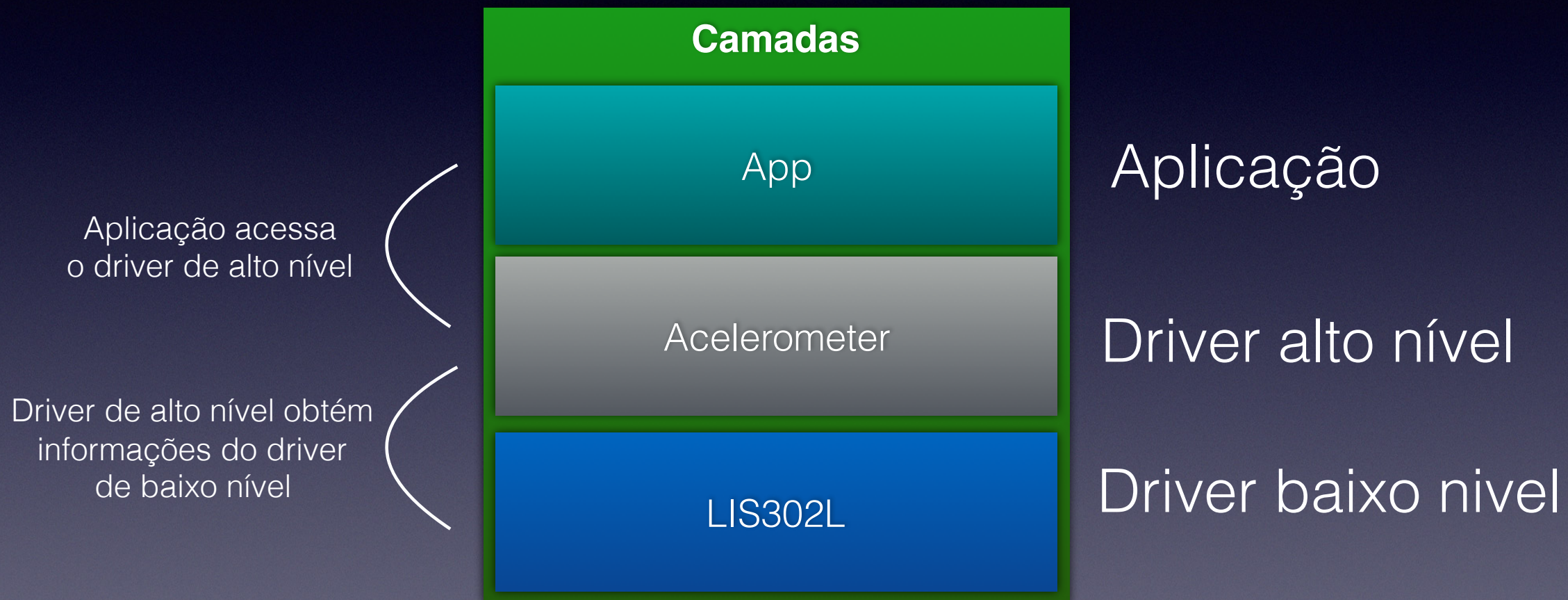
Princípios para Módulos em C

- Devem ser encapsulados;
- Baixo nível de acoplamento;
- Alta coesão;
- Evitar uso de variáveis globais;

Acoplamiento entre Modulos



Exemplo de Estrutura de Driver com Módulos em C



AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- **GPIO;**
- Debouncing;
- AD;

GPIO

- Pinos de Propósito Geral (daí o nome General Purpose I/O);
- Podem ser configurados como Entrada / Saída / Analógico / Interrupção / Pinos de Periféricos;
- Funções para manipulação podem ser vistas no outline do Arquivo `stm32f4xx_hal_gpio.c`;

AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- **Debouncing**;
- AD;

Input Debouncing

- Evita acionamentos acidentais;
- Filtro de ruído para entradas;

Algoritmo

1. Le valor da entrada;
2. Caso valor esteja em '1':
 3. Somar 1 à variável de debounce;
 4. Caso valor da var. de debounce seja maior que '50':
 5. Entrada ativa;
6. Caso valor esteja em '0':
 7. Zerar valor do debounce;
8. Caso 'Entrada ativa':
 9. Executa ação;

Aplicações GPIO

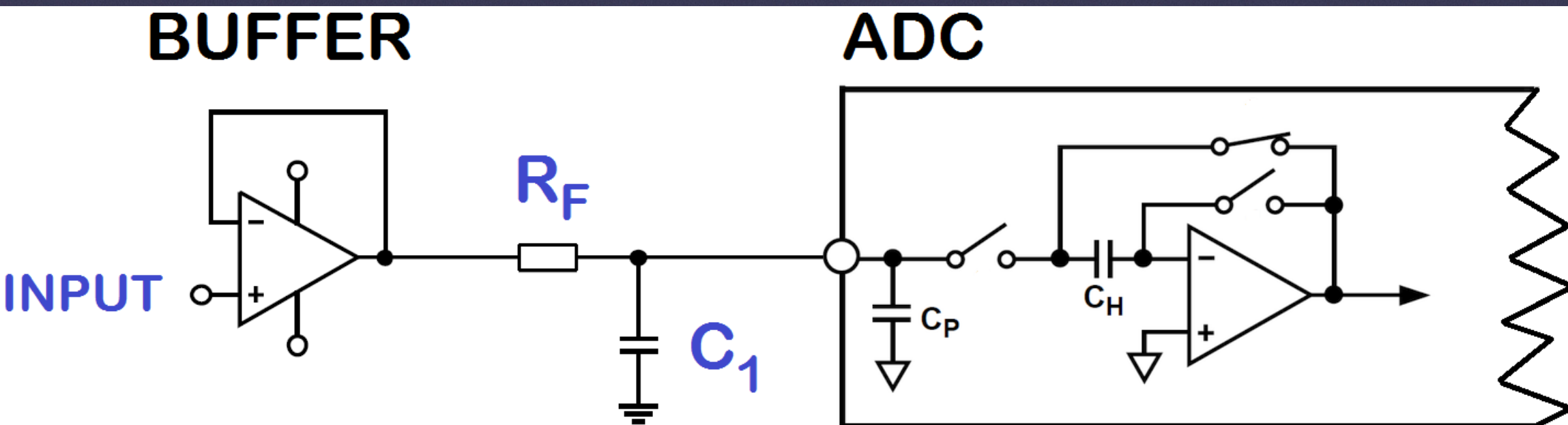
- Ex 1: Pisca *LED* (Sério ?);
- Ex 2: Leitura de 3 entradas (com debounce) + Pisca *LED*;
- Ex 3: Driver Display 7 Segmentos (nosso primeiro módulo) + Entrada + Pisca *LED*;

AGENDA

- Revisão do curso anterior;
- GIT (Github);
- Conceitos de programação;
- STM32CubeMX;
- IDE;
- SysTick;
- Programação Bare-Metal (Sem RTOS);
- GPIO;
- Debouncing;
- **AD**;

ADC (Analog to Digital Converter)

- ADC's de 12 bits (valores de 0 a 4095);
- Podem ser lidos em Polling, Interrupção e DMA;
- É recomendável filtrar os valores lidos;



Filtros para ADC

- Filtro da Media;
- Filtro IIR;
- Filtro FIR;
- etc;

Exercícios

- Leitura do AD em Polling;
- Leitura do AD por Interrupção;
- Leitura do AD por DMA;

Aplicações

- Leitura de um potenciômetro mostrando um valor de '0' de '9' no display de 7 segmentos (Polling, IT, DMA);
- Criação de driver para Teclado AD (Iremos ler 3 teclas usando um só pino);

What's next ?

- GPIO:
 - Teclado matricial;
- ADC:
 - Detectar padrões (Ex. Em corrente alternada detectar posição da fase);
 - Ler áudio;

Obrigado !