Esse artigo foi escrito pelo engenheiro Ismael Lopes da Silva, exclusivamente para o site "www.embarcados.com.br". O link para o artigo é "https://www.embarcados.com.br/blue-pill-stm32f103c8t6-api-gpio/".

No sétimo artigo da série "A Blue Pill", daremos continuidade no estudo. A dica é acompanhar a série desde o primeiro artigo porque é um caminho estruturado, onde a sequência traz benefícios no aprendizado (linha de raciocínio). Nesse artigo veremos as macros de programação e detalhes de configuração das Entradas e Saídas de Propósito Geral (GPIO).

API GPIO (Interface de programação de aplicativos)

Veremos as funções da biblioteca libopencm3 que estão disponíveis para trabalhar com as GPIOs. Primeiramente precisamos incluir os arquivos de cabeçalho, como mostrado a seguir:

```
#include <libopencm3/stm32/rcc.h>
#include <libopencm3/stm32/gpio.h>
```

O arquivo de cabeçalho rcc.h é necessário para definir e habilitar o clock para as GPIOs. O arquivo de cabeçalho gpio.h é necessário para trabalhar com as funções das GPIOs:

```
void gpio_set(uint32_t gpioport, uint16_t gpios)
void gpio_clear(uint32_t gpioport, uint16_t gpios)
uint16_t gpio_get(uint32_t gpioport, uint16_t gpios)
void gpio_toggle(uint32_t gpioport, uint16_t gpios)
uint16_t gpio_port_read(uint32_t gpioport)
void gpio_port_write(uint32_t gpioport, uint16_t data)
void gpio_port_config_lock(uint32_t gpioport, uint16_t gpios)
```

Para todas as funções anteriores, o argumento gpioport pode ser uma das macros da tabela 1.1. Outros MCU STM32, pode haver portas adicionais. Apenas uma porta pode ser especificado de cada vez.

Macro	**Descrição**
GPIOA	GPIO porta A
GPIOB	GPIO porta B
GPIOC	GPIO porta C

Tabela 1.1 – Macros referente as portas

Nas funções GPIO da biblioteca libopencm3, um ou mais bits GPIO podem ser resetados ou setados de uma única vez. A tabela 1.2 lista os nomes de macros suportadas. Observe também que tem a macro denominada GPIO_ALL.

Macro	**Definição**	**Descrição**
GPIO0	(1<<0)	Bit 0
GPIO1	(1<<1)	Bit 1
GPIO2	(1<<2)	Bit 2

GPIO3	(1<<3)	Bit 3
GPIO4	(1<<4)	Bit 4
GPIO5	(1<<5)	Bit 5
GPIO6	(1<<6)	Bit 6
GPIO7	(1<<7)	Bit 7
GPIO8	(1<<8)	Bit 8
GPIO9	(1<<9)	Bit 9
GPIO10	(1<<10)	Bit 10
GPIO11	(1<<11)	Bit 11
GPIO12	(1<<12)	Bit 12
GPIO13	(1<<13)	Bit 13
GPIO14	(1<<14)	Bit 14
GPIO15	(1<<15)	Bit 15
GPIO_ALL	0xffff	Todos os bit de o à 15

Tabela 1.2 – Macros referente aos pinos

Um exemplo do uso da macro GPIO_ALL pode ser o seguinte:

```
gpio_clear (PORTB, GPIO_ALL);          // limpa todos os pinos PORTB
```

Um recurso especial da série STM32, suportado pela biblioteca libopencm3, é a capacidade de bloquear uma configuração de GPIO, da seguinte maneira:

```
void gpio_port_config_lock (uint32_t gpioport, uint16_t gpios)
```

Depois de chamar a função gpio_port_config_lock(), para os pinos GPIO selecionados de uma determinada porta, a configuração dessas I/O é congelada até o próximo reset do MCU. Isso pode ser útil em sistemas que a segurança é crítica, para que não haja modificação acidental ou até mesmo intencional, portanto, quando uma GPIO for configurada como uma entrada ou uma saída, é garantido que permanecerá assim.

Configuração da GPIO

Veremos que a série STM32 é muito configurável. No artigo anterior, no programa miniblink, tem a seguinte chamada, que é curiosa:

```
rcc_periph_clock_enable (RCC_GPIOC)
```

Isso inclui os clocks necessários para as várias portas GPIO e periféricos. A função rcc_periph_clock_enable (RCC_GPIOC), da biblioetaca libopencm3, foi usada para habilitar o clock do sistema para a porta C. Se este clock não estiver habilitado, a GPIO porta C não funcionaria. O motivo pelo qual os clocks são desabilitados é para economizar no consumo de energia. Isto é importante para a conservação da bateria.

Continuando com o programa miniblink, a próxima função chamada é a gpio_set_mode():

```
gpio_set_mode(
        GPIOC,                          //macro da tabela 1.1
        GPIO_MODE_OUTPUT_2_MHZ,         //macro da tabela 1.3
        GPIO_CNF_OUTPUT_PUSHPULL,       //macro da tabela 1.4
        GPIO13                          //macro da tabela 1.2
    );
```

Esta função requer quatro argumentos. O primeiro argumento especifica a porta GPIO afetada, tabela 1.1. O segundo argumento define o modo do pino da porta GPIO, tabela 1.3, se o pino será entrada ou saída. O terceiro argumento é também relacionado com a configuração do pino da porta GPIO, tabela 1.4, se o pino é entrada ou saída analógica, digital ou função alternativa. O quarto argumento especifica o pino da porta GPIO afetada, tabela 1.2.

Macro	**Valor**	**Descrição**
GPIO_MODE_INPUT	0x00	Modo entrada
GPIO_MODE_OUTPUT_2_MHZ	0x02	Modo saída, até 2 MHz
GPIO_MODE_OUTPUT_10_MHZ	0x01	Modo saída, até 10 MHz
GPIO_MODE_OUTPUT_50_MHZ	0x03	Modo saída, até 50 MHz

Tabela 1.3 – Macros referente a configuração de I/O

A macro GPIO_MODE_INPUT define o pino GPIO como uma entrada, como seria de esperar. Mas há três macros do modo de saída. Cada seleção de saída afeta a velocidade com que cada pino de saída responde a uma transição de sinal. Em nosso programa de exemplo, a opção 2 MHz foi selecionada, porque é a menor frequência de trabalho, porém, é mais do que suficiente. Nesse caso o LED pisca lentamente, para ser perceptível aos olhos humanos. Também escolhendo 2 MHz, energia é economizada e a Interferência Eletromagnética (EMI) é reduzida.

Macro	**Valor**	**Descrição**
GPIO_CNF_INPUT_ANALOG	0x00	Modo entrada analógica
GPIO_CNF_INPUT_FLOAT	0x01	Modo entrada digital, flutuando (padrão)
GPIO_CNF_INPUT_PULL_UPDOWN	0x02	Modo entrada digital, pull-up e pull-down
GPIO_CNF_OUTPUT_PUSHPULL	0x00	Modo saída, push/pull
GPIO_CNF_OUTPUT_OPENDRAIN	0x01	Modo saída, open drain
GPIO_CNF_OUTPUT_ALTFN_PUSHPULL	0x02	Modo saída função alternativa, push/pull
GPIO_CNF_OUTPUT_ALTFN_OPENDRAIN	0x03	Modo saída função alternativa, open drain

Tabela 1.4 – Macros referente a configuração de I/O

Porta GPIO configurada como entrada

Se o segundo argumento da função gpio_set_mode() for GPIO_MODE_INPUT significa que estamos configurando o pino como uma entrada que pode ser :

- ✔ Entrada Analógica;
- ✔ Entrada digital flutuante;
- ✔ Entrada digital, pull-up e pull-down.

Para entender melhor a entrada GPIO e sua configuração, vamos ver a figura a seguir.

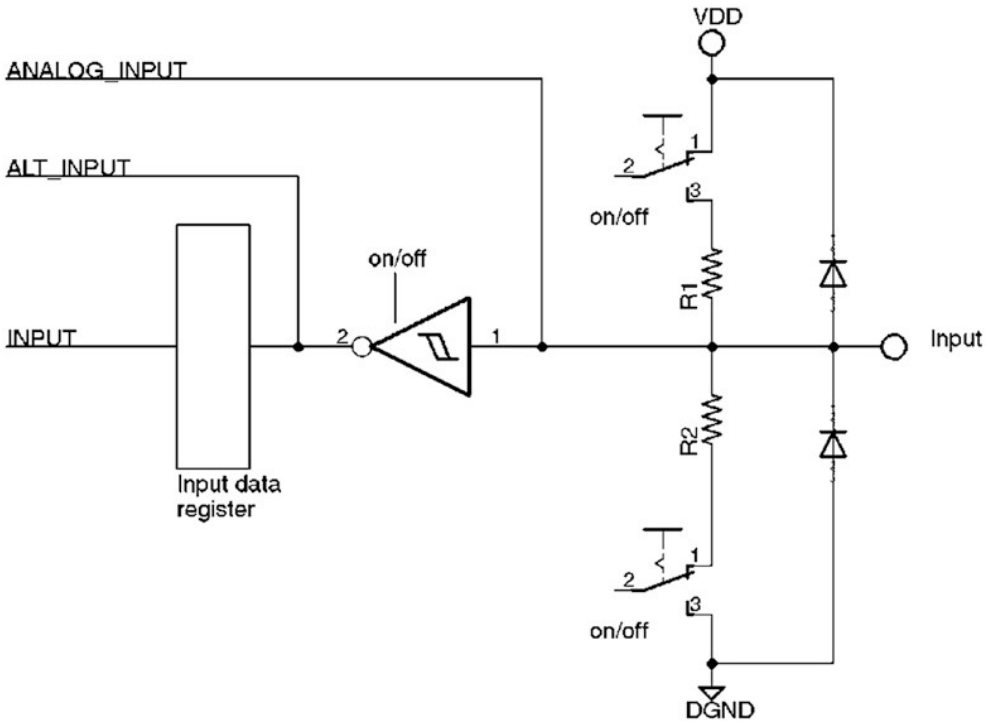

Figura 1.2 – Esquema elétrico de uma porta de entrada

Obviamente que o lado do MCU é a saída do Registrador de Dado de Entrada, e o lado da entrada que interliga com o meio externo, pino do MCU, é onde temos os dois diodos de proteção contra tensão estática ou excesso de potencial na entrada.

Quando o pino é configurado como uma entrada analógica, (GPIO_CNF_INPUT_ANALOG) as duas chaves, que conectam os resistores R1 e R2, são desligadas para não interferirem no sinal analógico que vem do pino externo. Na verdade não existem essas chaves, mas, dispositivos com esse comportamento. A porta inversora Schmitt Trigger também é desabilitada para reduzir o consumo de energia. O sinal dessa entrada analógica é encaminhado para a linha rotulada como "Entrada Analógica" que vai para um periférico Conversor de Sinal Analógico para Digital (ADC).

Quando o pino é configurado como uma entrada digital flutuante (GPIO_CNF_INPUT_FLOAT), as duas chaves, que conectam os resistores R1 e R2, são desligadas para não interferirem no sinal digital que vem do pino externo. Se o pino é configurado como uma entrada digital pull-up e pull-down (GPIO_CNF_INPUT_PULL_UPDOWN), as duas chaves, que conectam os resistores R1 e R2, são ligadas, ativando os resistores R1 (pull-up) e R2 (pull-down). Para ambos modos de entradas digitais, a porta inversora Schmitt Trigger é habilitada, para fornecer um sinal digital limpo pela histerese da inversora Schmitt Trigger. O sinal digital na saída da porta inversor Schmitt Trigger vai para a linha rotulada como "Entrada Alternativa" e também para a entrada do Registrador de Dado de Entrada. Depois veremos mais detalhes sobre funções alternativas, mas, para o momento podemos dizer que uma entrada pode atuar como uma entrada GPIO ou como uma entrada de um periférico.

Algumas entradas do MCU toleram +5 Volts. O diagrama elétrico dessas entradas são idênticos ao da ilustração da figura 1.2, exceto que os diodos de proteção permitem que a tensão suba acima de +3,3 Volts até +5 Volts.

Porta GPIO configurada como saída

Quando um pino de uma porta GPIO for configurado como saída, temos quatro opções para configurá-la, que são:

- ✔ Modo saída, push/pull;
- ✔ Modo saída, open drain;
- ✔ Modo saída função alternativa, push/pull;
- ✔ Modo saída função alternativa, open drain.

Para trabalhar como saída GPIO padrão, você não deve escolher os modos de função alternativas. Para usar periféricos, como a USART, você deve escolher entre os modos de função alternativas. Um erro comum é configurar uma saída como pino GPIO padrão, como por exemplo, GPIO_CNF_OUTPUT_PUSHPULL, sendo que seria uma saída TX da USART. A macro correta para essa situação seria GPIO_CNF_OUTPUT_ALTFN_PUSHPULL para esse periférico.

A figura 1.3 ilustra o diagrama de blocos para saídas GPIO. Para saídas tolerantes à +5 Volts, como vimos nas entradas, a única alteração no circuito é que os diodos de proteção são capazes de suportar tensões até +5 Volts. Para portas não tolerantes a +5 Volts, os diodos de proteção suportam até +3,3 Volts.

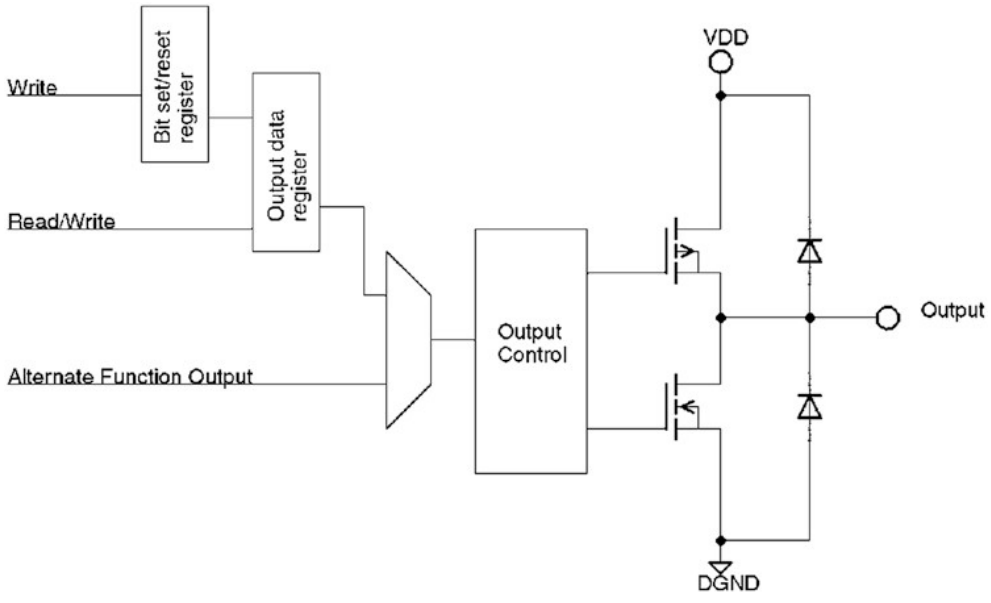

Figura 1.3 – Esquema elétrico de uma porta de saída

O circuito de "controle de saída", bloco "Output Control", no modo push/pull, determina se está acionando o Transistor P-MOS ou o N-MOS. No modo open drain, apenas o Transistor N-MOS trabalha, e o Transistor P-MOS é desabilitado.

Quando você escreve um zero na saída o Transistor N-MOS conduzirá e chaveará o pino de saída para baixo (potencial GND). No modo open drain quando escrevemos um nível alto na saída o Transistor N-MOS também não conduzirá, então, os dois Transistores são colocados no estado "desligado". Os resistores de entrada, com valores de resistências altos, ilustrados na figura 1.2, são desativados no modo de saída, portanto, foram omitidos na figura 1.3.

Os bits de dados de saída são selecionados no Registrador de Dados de Saída da GPIO ou na fonte de função alternativa. As saídas GPIO vão para o Registro de Dados de Saída, que pode ser escrito como uma palavra inteira ou como bits individuais.

O Registrador Set/Reset Bits permite que bits individuais da GPIO possam ser alterados como se fossem uma operação atômica. Em outras palavras, uma interrupção não pode ocorrer no meio de uma operação and/or com um bit. Observe que os dados de saída de uma GPIO podem ser capturados no Registrador de Dados de Saída, então, é possível ler novamente quais são as configurações de saída atuais. Isso não funciona para as funções alternativas.

Quando a saída é configurada para um periférico, por exemplo, a USART, os dados são provenientes desse periférico através da linha de saída da função alternativa. Verificando como os dados são direcionados, através da ilustração na figura 1.3, enfatizo o fato de que você deve configurar a porta para GPIO ou funções alternativas, evitando perdas de tempo devido problemas de configuração.

Alinhamento de configuração para evitar problemas

Os periféricos na plataforma STM32 são altamente configuráveis, portanto, aumenta a chance de cometer erros. Para reduzir a chance de errar, vamos criar uma sequência alinhada para termos sucesso na configuração do MCU.

Normalmente, o problema é uma omissão ou o uso de uma macro incorreta que gera um erro no processo de compilação. A sequência de configuração é importante, então, se tivermos um passo a passo nos ajudará bastante, aumentando as chances de sucesso.

Entradas GPIO

Ao configurar os pinos como entrada GPIO, use o procedimento a seguir para configurá-lo. Isso se aplica apenas às entradas GPIO, e não para uma entrada periférica, como uma entrada da USART.

- ✔ Habilite o clock da porta GPIO. Por exemplo, se o pino GPIO for da porta C, habilite o clock com uma chamada para a função rcc_periph_clock_enable(RCC_GPIOC). Você deve habilitar cada porta usada individualmente, usando as macros RCC_GPIOx;

- ✔ Configure o modo do pino de entrada com a função gpio_set_mode(), especificando a porta no primeiro argumento, e a macro GPIO_MODE_INPUT no segundo argumento;

- ✔ Na chamada da função gpio_set_mode(), escolha a macro apropriada GPIO_CNF_INPUT_ANALOG, GPIO_CNF_INPUT_FLOAT ou GPIO_INPUT_PULL_UPDOWN conforme sua aplicação requer;

- ✔ Finalmente, especifique no último argumento na chamada da função gpio_set_mode() todos os números de pinos que se aplicam essa configuração. Por exemplo, eles podem ser agrupados, como GPIO12 | GPIO15.

Saída GPIO Digital, Push/Pull

Normalmente, as saídas digitais são configuradas para o modo push/pull, então, para esse caso:

- ✔ Habilite o clock da porta GPIO. Por exemplo, se o pino GPIO for da porta B, habilite o clock com uma chamada para a função rcc_periph_clock_enable(RCC_GPIOB). Você deve habilitar cada porta usada individualmente, usando as macros RCC_GPIOx;

- ✔ Configure o modo do pino de saída com a função gpio_set_mode(), especificando a porta no primeiro argumento, e uma das macros GPIO_MODE_OUTPUT _ * _ MHZ no segundo argumento. Para taxa de sinal não crítica, sempre escolha o valor mais baixo, GPIO_MODE_OUTPUT_2_MHZ, para economizar energia e reduzir a EMI;

- ✔ Na chamada para a função gpio_set_mode(), defina a macro GPIO_CNF_OUTPUT_PUSHPULL no terceiro argumento. Não use nenhuma das macros ALTFN para uso como GPIO, elas são apenas para uso com periférico;

- ✔ Finalmente, especifique no último argumento na chamada da função gpio_set_mode() todos os números de pinos que se aplicam essa configuração. Por exemplo, eles podem ser agrupados, como GPIO12 | GPIO15.

Saída GPIO Digital, Open Drain

Ao trabalhar com um barramento, onde mais de que um Transistor pode ser usado para pull-down uma tensão, uma saída open drain é requerida. Exemplos são encontrados nas comunicações de barramento I2C ou CAN. O procedimento a seguir é recomendado apenas para saídas GPIO open drain, e não use este procedimento para saída de periféricos:

- ✔ Habilite o clock da porta GPIO. Por exemplo, se o pino GPIO for da porta B, habilite o clock com uma chamada para a função rcc_periph_clock_enable(RCC_GPIOB). Você deve habilitar cada porta usada individualmente, usando as macros RCC_GPIOx;

- ✔ Configure o modo do pino de saída com a função gpio_set_mode(), especificando a porta no primeiro argumento, e uma das macros GPIO_MODE_OUTPUT _ * _ MHZ no segundo argumento. Para taxa de sinal não crítica, sempre escolha o valor mais baixo, GPIO_MODE_OUTPUT_2_MHZ, para economizar energia e reduzir a EMI;

- ✔ Na chamada para a função gpio_set_mode(), defina a macro GPIO_CNF_OUTPUT_OPENDRAIN no terceiro argumento. Não use nenhuma das macros ALTFN para uso como GPIO, elas são apenas para uso com periférico;

- ✔ Finalmente, especifique no último argumento na chamada da função gpio_set_mode() todos os números de pinos que se aplicam essa configuração. Por exemplo, eles podem ser agrupados, como GPIO12 | GPIO15.

Características da GPIO

Resumiremos os recursos dos pinos do GPIO STM32. Muitos são tolerantes a +5 Volts como entradas, enquanto, outros têm corrente limitada na saída. Utilizando a convenção de documentação do STM32, as portas são frequentemente referenciadas como PB5, para se referir à porta GPIO B do pino GPIO5, então, usaremos esta convenção.

A tabela 1.5 resume essas importantes características da GPIO, conforme elas se aplicam ao dispositivo Blue Pill. Todos os pinos das GPIOs suportam uma corrente de 25 mA, exceto onde destacado outro valor.

	GPIO_PORTA			GPIO_PORTB			GPIO_PORTC		
Pino	**Tensão**	**Reset**	**Altern.**	**Tensão**	**Reset**	**Altern.**	**Tensão**	**Reset**	**Altern.**
GPIO0	3V	PA0	Sim	3V	PB0	Sim			
GPIO1	3V	PA1	Sim	3V	PB1	Sim			
GPIO2	3V	PA2	Sim	5V	PB2/ BOOT1	Não			
GPIO3	3V	PA3	Sim	5V	JTDO	Sim			
GPIO4	3V	PA4	Sim	5V	JNTRST	Sim			
GPIO5	3V	PA5	Sim	3V	PB5	Sim			
GPIO6	3V	PA6	Sim	5V	PB6	Sim			
GPIO7	3V	PA7	Sim	5V	PB7	Sim			
GPIO8	5V	PA8	Não	5V	PB8	Sim			
GPIO9	5V	PA9	Não	5V	PB9	Sim			
GPIO10	5V	PA10	Não	5V	PB10	Sim			
GPIO11	5V	PA11	Não	5V	PB11	Sim			
GPIO12	5V	PA12	Não	5V	PB12	Não			
GPIO13	5V	JTMS/ SWDIO	Sim	5V	PB13	Não	3V	3mA @2MHz	Sim
GPIO14	5V	JTCK/ SWCLK	Sim	5V	PB14	Não	3V	3mA @2MHz	Sim
GPIO15	5V	JTDI	Sim	5V	PB15	Não	3V	3mA @2MHz	Sim

Tabela 1.5 – Características das GPIOs

A coluna "Altern." na tabela 1.5 indica onde funções alternativas podem ser aplicadas. As GPIOs de entrada marcadas com "5V" podem tolerar com segurança um sinal de +5 Volts, enquanto os outros marcados com "3V" podem aceitar apenas sinais de até +3,3 Volts. A coluna "Reset" indica o estado da configuração do GPIO após um reset e reinicialização do MCU.

Os pinos GPIO PC13, PC14 e PC15 têm corrente limitada em máximo 3 mA, e nunca devem ser usadas como fonte de corrente. Além disso, a documentação indica que esses pinos nunca devem ser configurados para operações superiores a 2 MHz, quando configurados como saídas.

Limites de tensão de entrada

Como o STM32F103C8T6 pode operar em uma faixa de tensões, então, a tabela 1.6 documenta o que você pode esperar do dispositivo Blue Pill, operando a +3,3 volts.

Símbolo	**Descrição**	**Faixa**
V_{IL}	Tensão entrada nível baixo padrão	0 à 1,164 Volts
V_{IL}	Entradas que toleram +5V	0 à 1,166 Volts
V_{IH}	Tensão entrada nível alto	1,155 à 3,3/5,0 Volts

Tabela 1.6 – Limites de tensão baseado numa alimentação padrão de +3,3 Volts

Você deve ter notado que há uma pequena sobreposição entre os limites superior do V_{IL} e o limite inferior do intervalo V_{IH}. A documentação do STM32 indica que há cerca de 200 mV de histerese entre esses estados de entrada.

Limites de tensão de saída

Os limites do GPIO de saída estão documentados na tabela 1.7, com base no dispositivo Blue Pill operando a +3,3 Volts.

Símbolo	**Descrição**	**Faixa**
V_{OL}	Tensão saída nível baixo	0,4 à 1,3 Volts
V_{OH}	Tensão saída nível alto	2,0 à 3,3 Volts

Tabela 1.7 – Limites de tensão baseado com corrente <= 20 mA

Conforme a proposta inicial estamos evoluindo no aprendizado, portanto, aqui concluo o sétimo artigo da série.