

Obs.: Material em edição constante para realizar pequenas melhorias.

Alô mundo Com a Greenpill

Nesta Prática teremos o primeiro contato com um processador de arquitetura ARMV6-M, mais precisamente um core Cortex M0, que é construído para permitir que os desenvolvedores construam plataformas de baixo custo de alto desempenho para uma ampla gama de dispositivos, incluindo controladoras, sistemas automotivos, sistemas de controle industrial e redes de sensores sem fio.

A placa Greenpill utilizada se baseia no microcontrolador STM32F070F696 de 32 bits da ST. Para realizar a gravação de debug deste microcontrolador se faz necessário a aquisição de um outro dispositivo, o STLink (Programador) ou um conversor USB-UART. Nesta prática e durante toda a disciplina, utilizaremos o programador STLink v2.

Para começar as atividades precisamos tomar uma primeira e muito importante decisão, qual IDE/Compilador utilizar, e podemos citar os mais profissionais, que são o Keil uVision e IAR SystemWorkbench for STM32. Existem diversos packs para utilizar IDE's conhecidas como o Eclipse ou NetBeans. Nós utilizaremos a ferramenta STMCubelide, gratuita e mantida pela própria STM. O STM32CUBEIDE é baseado no Eclipse, que já é nosso conhecido de práticas anteriores.

Em todo microcontrolador há pelo menos uma porta de IO de uso geral e com a stm32 não é diferente. A Greenpill tem 3 portas de entrada/saída de uso geral (GPIO), chamadas de Port A, B e F, na figura 1 você pode perceber a disposição

delas. Os pinos de cada porta têm vários modos de operação e isso é o que os torna robustos e complexos no início. Nas placas de desenvolvimento, a nomeação do pino da porta IO é truncada e, portanto, encontraremos PA0, PA12, etc. em vez de GPIOA0, GPIOA12, etc. Mesmo nos manuais de referência, essa nomeação curta é amplamente utilizada.

Este primeiro programa é bem simples, nós utilizaremos a porta PA0 da placa para comandar o acionamento de um LED.

Montando o Hardware

Configure o lado SLAVE da fonte de alimentação da sua bancada para fornecer uma tensão de 3.3V@100mA .

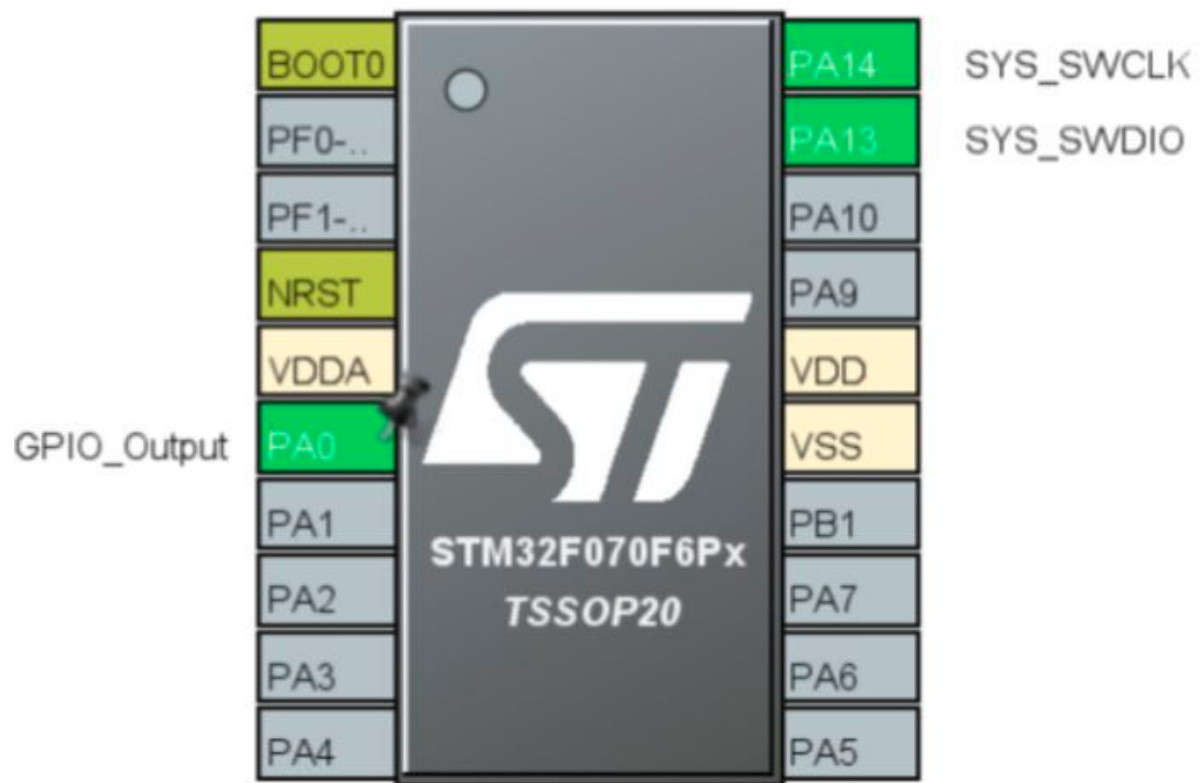


Figura 1: Detalhamento dos pinos da Greenpill



Figura 1: STLink V2

VCC	1	<input type="checkbox"/>	<input type="checkbox"/>	2	VCC (optional)
N/U	3	<input type="checkbox"/>	<input type="checkbox"/>	4	GND
N/U	5	<input type="checkbox"/>	<input type="checkbox"/>	6	GND
SWDIO	7	<input type="checkbox"/>	<input type="checkbox"/>	8	GND
SWCLK	9	<input type="checkbox"/>	<input type="checkbox"/>	10	GND
N/U	11	<input type="checkbox"/>	<input type="checkbox"/>	12	GND
SWO	13	<input type="checkbox"/>	<input type="checkbox"/>	14	GND
RESET	15	<input type="checkbox"/>	<input type="checkbox"/>	16	GND
N/C	17	<input type="checkbox"/>	<input type="checkbox"/>	18	GND
N/C	19	<input type="checkbox"/>	<input type="checkbox"/>	20	GND

SMD

Figura 2: Detalhamento dos pinos STM32 da STLink V2

Tabela 1: Conexões entre Stlink v2 e Greenpill

STLINKV2	GREENPILL
Pino 1 (VCC)	Pino 5 ou 16 (VDD)
Pino 4 (GND)	Pino 15 (VSS)
Pino 7 (SWDIO)	Pino 19 (PA13 ou SYS_SWDIO)
Pino 9 (SWCLK)	Pino 20 (PA14 ou SYS_SWCLK)

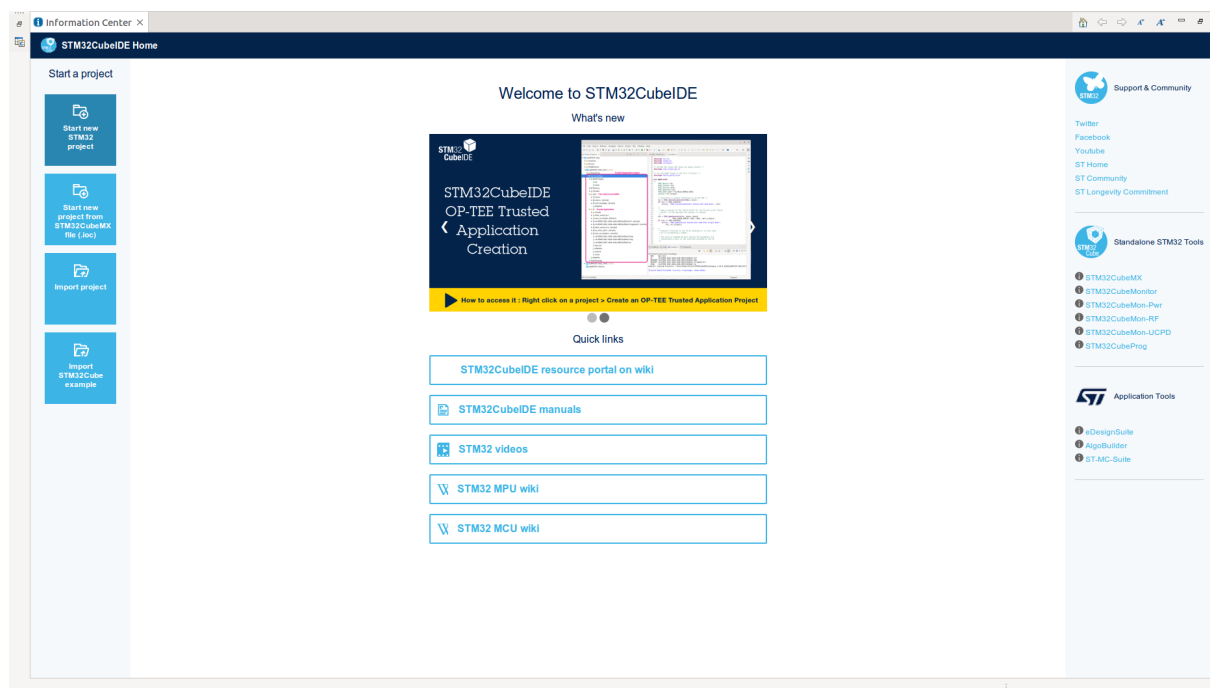
Tabela 2: Conexões entre Greenpill e Fonte de alimentação

Fonte de alimentação	GREENPILL
3.3V	Pino 5 (VDD)
GND	Pino 15 (VSS)
3.3V	Pino 16 (VDD)
GND	Pino 1 (BOOT)

Monte também um circuito que conecte um led ao pino PA0 por meio de um resistor.

Compilando e Debugando

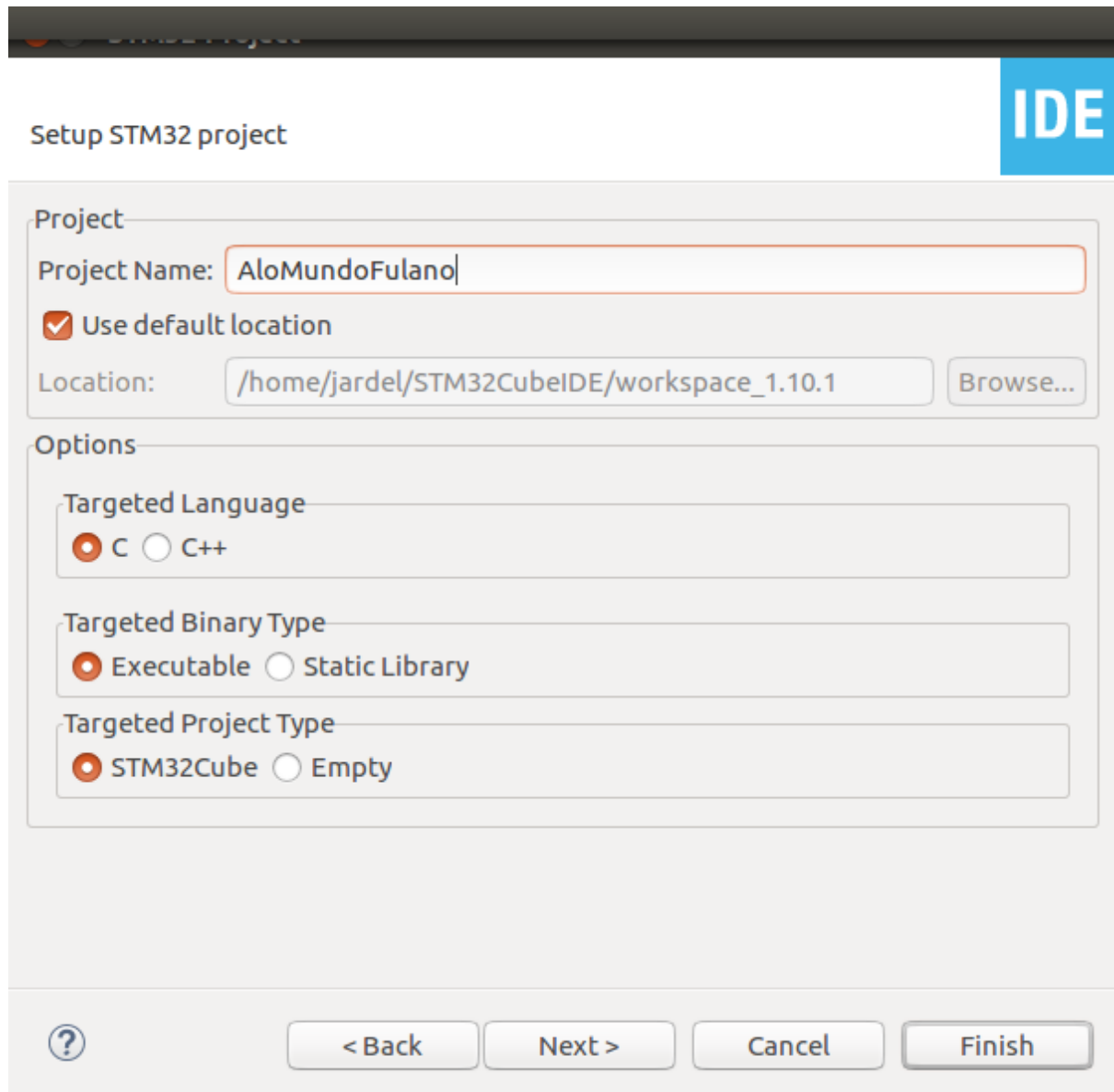
Abra um terminal e execute o cube : /opt/st/stm32cubeide_1.10.1/stm32cubeide &



Selecione Start New STM32 Project

preencha com o microcontrolador STM32F070F6 o campo commercial part number

Preencha o campo project name e clique em next



The image shows the 'Setup STM32 project' dialog box in STM32CubeIDE. The dialog has a title bar with the STM32CubeIDE logo and a blue 'IDE' button. The main content is divided into two sections: 'Project' and 'Options'. In the 'Project' section, the 'Project Name' field is filled with 'AloMundoFulano'. The 'Use default location' checkbox is checked. The 'Location' field shows the path '/home/jardel/STM32CubeIDE/workspace_1.10.1' with a 'Browse...' button next to it. The 'Options' section contains three groups of radio buttons: 'Targeted Language' with 'C' selected, 'Targeted Binary Type' with 'Executable' selected, and 'Targeted Project Type' with 'STM32Cube' selected. At the bottom, there is a help icon (question mark) and four buttons: '< Back', 'Next >', 'Cancel', and 'Finish'.

Setup STM32 project

IDE

Project

Project Name: AloMundoFulano

☒ Use default location

Location: /home/jardel/STM32CubeIDE/workspace_1.10.1 Browse...

Options

Targeted Language

☒ C ☐ C++

Targeted Binary Type

☒ Executable ☐ Static Library

Targeted Project Type

☒ STM32Cube ☐ Empty

? < Back Next > Cancel Finish

E então em Finish na tela abaixo

STM32 Project

Firmware Library Package Setup

Setup STM32 target's firmware

IDE

Target and Firmware Package

Target Reference: STM32F070F6Px

Firmware Package Name and Version: STM32Cube FW_F0 V1.11.3

Firmware and Software Package Repository

Location: /home/jardel/STM32Cube/Repository

See 'Firmware Updater' for settings related to package installation

Code Generator Options

- ☐ Add necessary library files as reference in the toolchain project configuration file
- ☐ Copy all used libraries into the project folder
- ☒ Copy only the necessary library files

? < Back Next > Cancel Finish

Selecione yes na tela abaixo

Open Associated Perspective?

Device Configuration Tool editor is associated with Device Configuration Tool perspective. Do you want to open this perspective now?

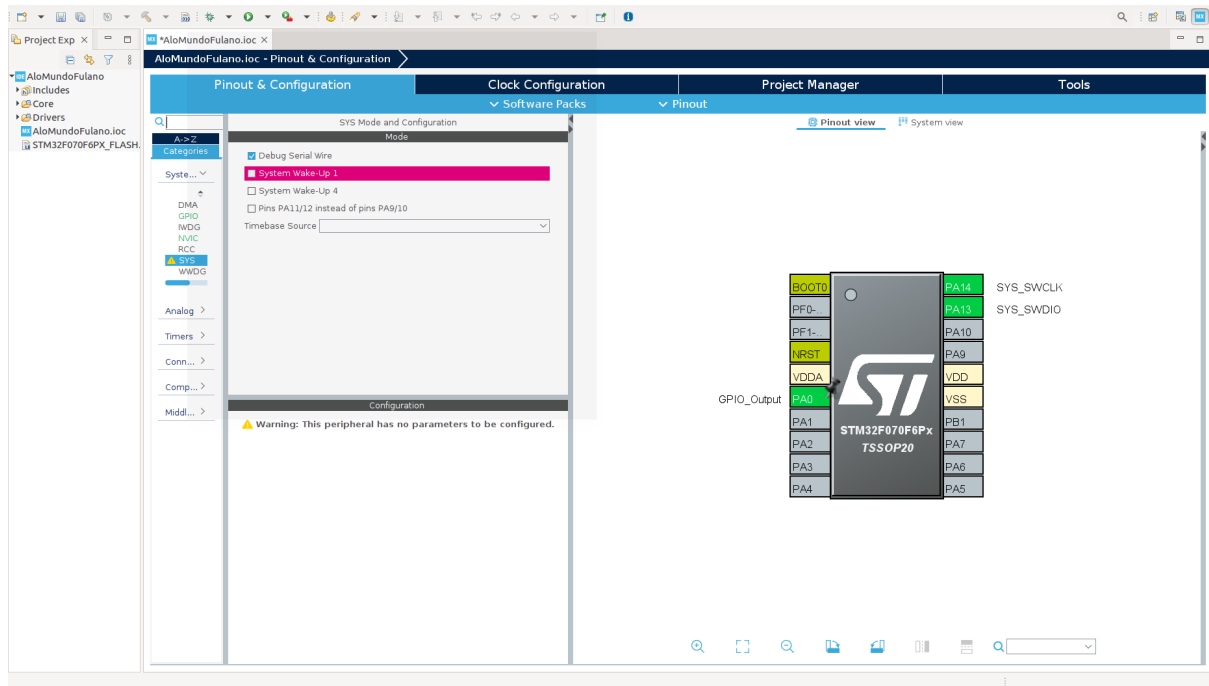
☒ Remember my decision

No Yes

Aguarde o download, caso necessário.

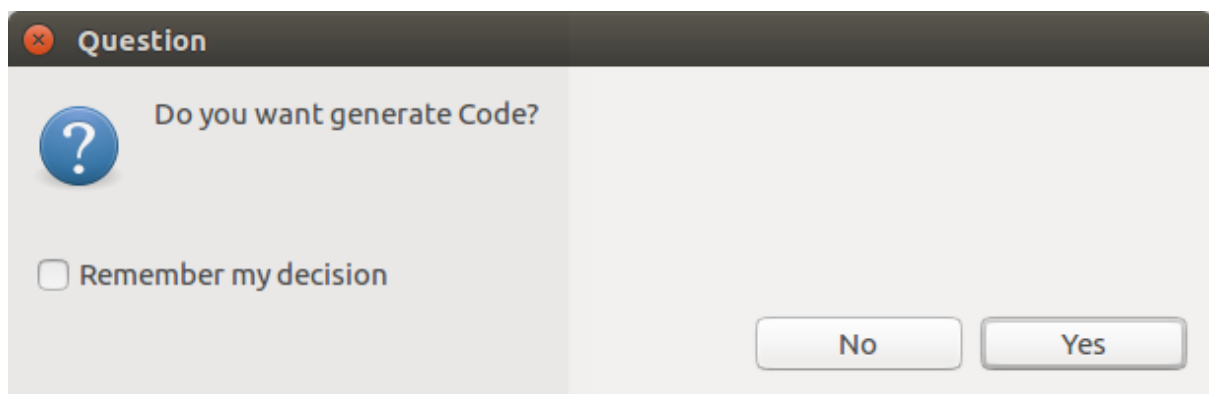
Clique em PA0 e configure-o como GPIO_Ouput

Na aba da esquerda, em Sys, marque Debug Serial Wire. Se você se esquecer de fazer isso, ao executar seu código no microcontrolador, o mesmo ficará em um estado travado, e somente poderá ser destravado no modo 'connect under reset' no software stlink utility para windows. Dica: NÃO ESQUEÇA !

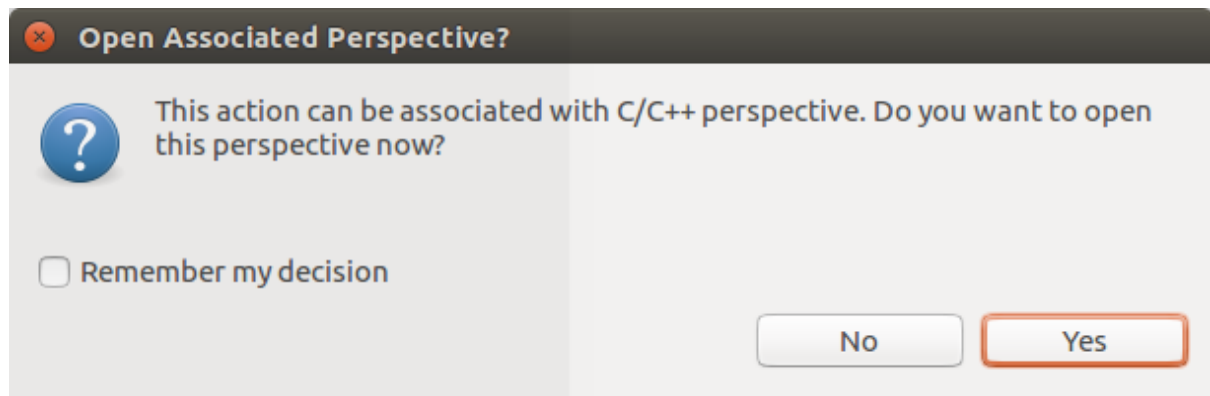


Clique em File -> Save ou digite ctrl+s

Clique em yes na tela abaixo:



E mais uma vez em yes na tela abaixo



É necessário desabilitar as otimizações feitas pelo compilador:

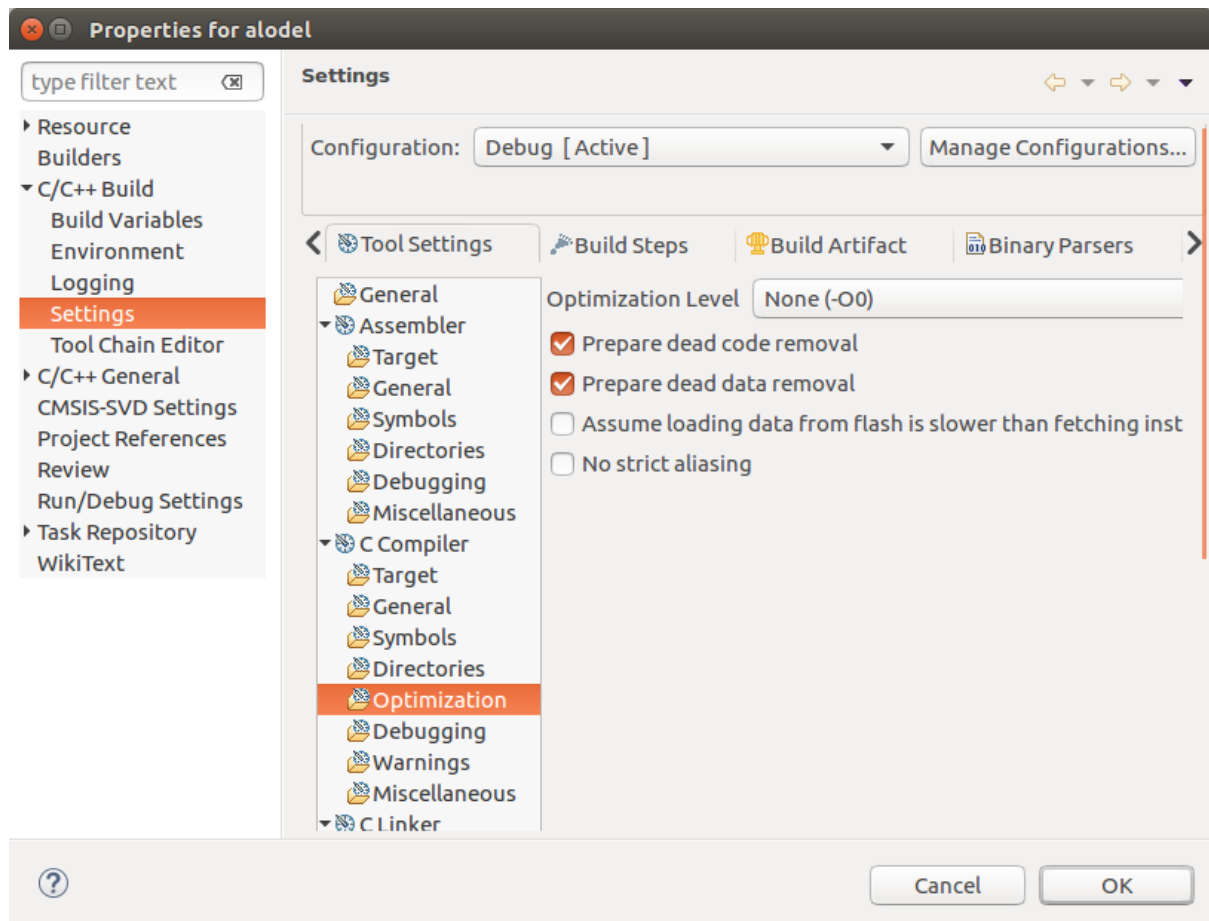
Project -> properties

item Settings em c/c++ build

aba tools settings

c compiler -> optimization

Selecione optimization level 'None (-O0)'. Se você esquecer de fazer isso, a execução passo a passo (com F6) ficará inconsistente, fazendo parecer que você tem um bug.



Agora vamos alterar o código gerado pelo Cube para fazer o LED piscar:

Em Src, abra a main.c e insira essas duas linhas dentro do loop while(1), acima da linha de comentário "/* USER CODE END WHILE */":

```
HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_0);
HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_0);
```

HAL_GPIO_TogglePin

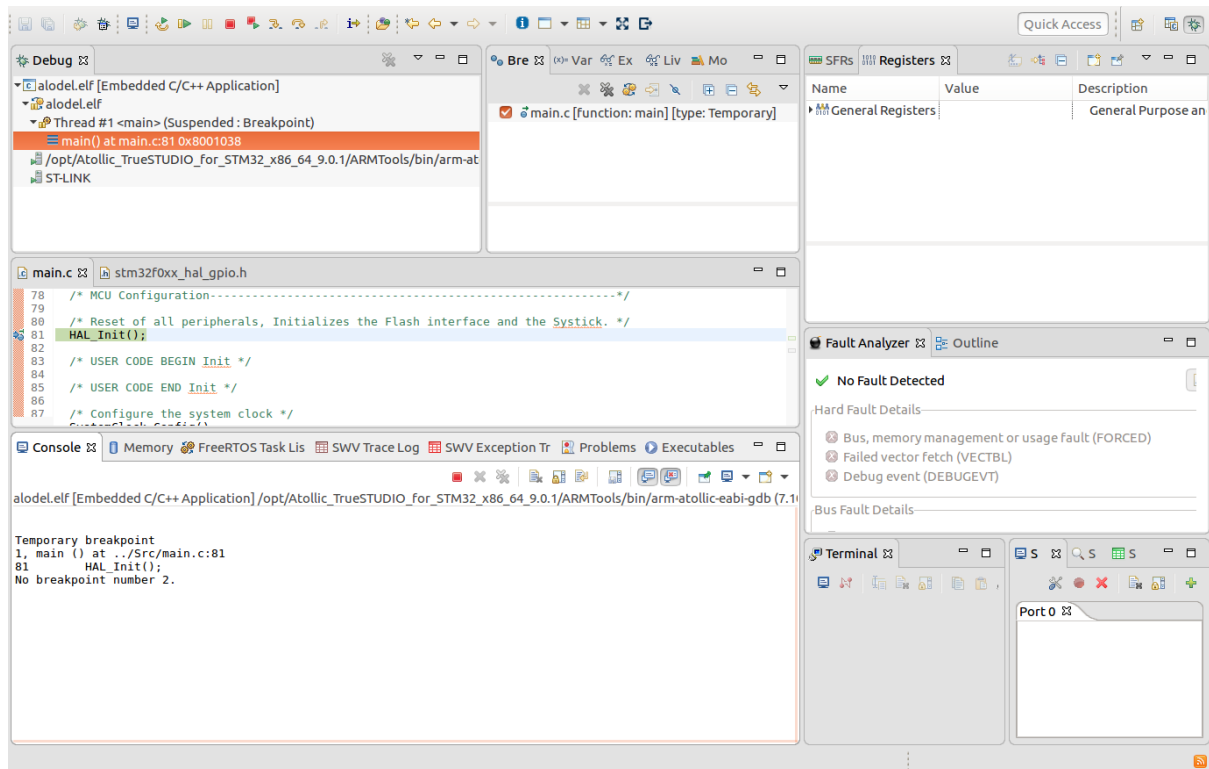
Function name	void HAL_GPIO_TogglePin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)
Function description	Toggle the specified GPIO pin.
Parameters	<ul style="list-style-type: none"> GPIOx: where x can be (A..F) to select the GPIO peripheral for STM32F0 family GPIO_Pin: specifies the pin to be toggled.
Return values	<ul style="list-style-type: none"> None

* *Snapshot* parcial da página 213 do documento UM1785 User manual [3].

Use CTRL+B para compilar e linkar

Em seguida menu run->debug para iniciar carregar o programa no microcontrolador.

Clique no código main.c



Use F6 para executar passo a passo até ver o LED piscando.

Ou Insira um breakpoint (botão direito do mouse) e Use F8 para executar até o breakpoint ser atingido

Atividade inicial: Faça o LED piscar a uma frequência aproximada de 1Hz (Execução contínua com F8) . Dica: Veja o documento “Description of STM32F0 HAL and low-layer drivers” postado no Sigaa e procure por um função que funcione como delay.

Atividade: Agora desenvolva seu primeiro projeto.

Baseado no programa anterior, desenvolva um programa que utilize o LED PA0 para emitir a uma chamado de SOS em código morse. Em um laço o led deverá piscar 3 vezes mais rápido, e em outro laço deverá piscar em intervalos maiores. A emissão do código morse deverá ter um intervalo de exibição entre os ciclos para que o fim da mensagem não fique colada ao início da retransmissão.

Desafio: Código genérico para Código Morse.

Crie um código genérico que consiga transcrever qualquer mensagem informada para o formato morse abaixo:

“.” = ponto= emissão rápida | “-” = linha = emissão lenta

A	.-	J	.-.-.-	S	...	2	..-.-.-
B	-...	K	-.-	T	-	3	...--
C	-.-.-	L	.-.-.	U	..-	4-
D	-..	M	--	V	...-	5
E	.	N	-.	W	.-.-	6	-.....
F	...-	O	---	X	-.-.-	7	-.....
G	---	P	.-.-.	Y	-.-.-	8	-----
H	Q	---.-	Z	---.	9	-----
I	..	R	.-.	1	.-.-.-.-	0	-----

Dica: Faça um porting da implementação do Professor David Money Harris
<http://pages.hmc.edu/harris/class/e85/morse.c>

Agora, pois, permanecem a fé, a esperança e o amor. Porém o maior desses é o amor.
(1 Cor. 13:13)

Referências

[1] <https://www.st.com/resource/en/datasheet/dm00088500.pdf>

[2] <http://pages.hmc.edu/harris/class/e85/morse.c>

[3] UM1785 User manual, Description of STM32F0 HAL and low-layer drivers,
https://www.st.com/content/ccc/resource/technical/document/user_manual/2f/77/25/0f/5c/38/48/80/DM00122015.pdf/files/DM00122015.pdf/jcr:content/translations/en.DM00122015.pdf