Bancada para teste do sistema PA

Wilson Souza

Agosto de 2019

1 Introdução

O Sistema Navio2 é composto por uma placa que é acoplada diretamente ao sistema Raspberry PI3 disponibilizando os seguintes sensores e interfaces:

- **Unidade de medição inercial MPU9250**
- **Unidade de medição inercial LSM9DS1**
- **Barômetro MS5611**
- **Entrada PPM**
- **14 Saídas PWM**
- **GPS**
- **6 entradfas analógicas**
- **Interface UART**
- **Interface I2C**

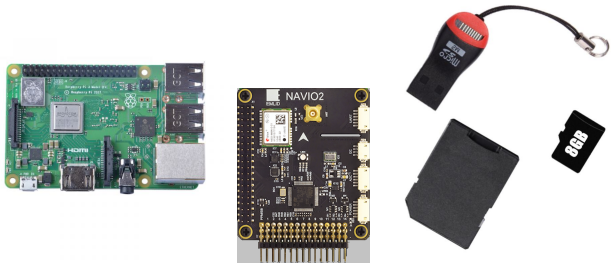
A interface UART é utilizada para comunicar-se com o Display Nextion com *touchscreen* e a interface I2C será utilizada para acessar outras 4 portas analógicas do conversor analógico-digital ADS1115. Vale ressaltar a existência de 4portas USB no sistema Raspberry PI3 que podem ser utilizadas para conectarrem-se com outros periféricos tais como interfaces UART, teclado, mouse, etc.

Para preparar o sistema Raspberry para utilizar os recursos da placa Navio2 e do Display Nextion é necessária a execução das seguintes etapas:

1. Baixar o arquivo de imagem do cartão SD com o sistema operacional disponibilizado pelo fabricante da placa Navio2.
2. Baixar o software para gravar o arquivo de imagem no cartão SD.
3. Gravar o cartão SD com arquivo de imagem adquirido anteriormente.
4. Baixar o script de instalação dos arquivos auxiliares no desenvolvimento do sistema PA.
5. Primeira inicialização do sistema e configurações iniciais.
6. Baixar softwares de acesso remoto e primeiro acesso.

Para se obter um sistema Raspberry apto para o desenvolvimento do controlador são necessários os seguintes itens:

- Placa Raspberry, apresentada na figura 1a
- Placa Navio2, apresentada na figura 1b.
- Cartão microSD de pelo menos 4GB e adaptador microSD para USB, apresentado na figura 1c.



(a) (b) (c)

Figura 1: Hardware necessário para configurar o sistema Raspberry.

Segue a descrição de cada uma destas etapas.

1.1 Baixar a imagem do Sistema Operacional

Acesse o site `http://files.emlid.com/images/emlid-raspbian-20190227.img.xz` e salve o arquivo em uma pasta em seu computador.

1.2 Baixar software para gravar imagem no cartão SD

Acesse o site `https://www.balena.io/etcher/` e selecione o arquivo a ser baixado conforme o sistema operacional que esteja utilizando.

1.3 Gravar a imagem do Sistema Operacional no cartão SD

Execute o software baixado anteriormente, a tela inicial é apresentada na figura 2. Nela já é possível fazer as duas configurações necessárias. Clique no botão "Select image"e carregue o arquivo da imagem baixado. Feito isso selecione o disco alvo onde será gravado o conteúdo do arquivo imagem.E por fim inicie a gravação.

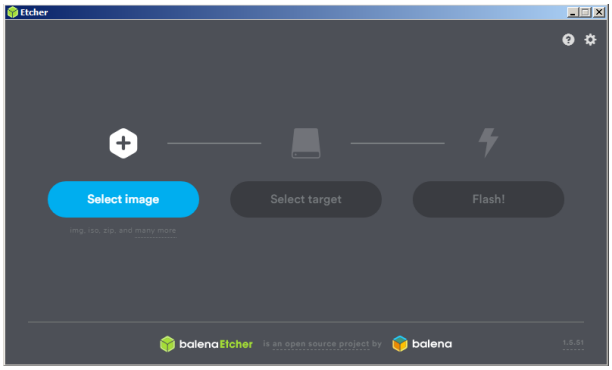


Figura 2: Tela inicial Balena Etcher.

1.4 Baixar script de instalação

Uma vez concluído o processo a partição boot estará acessível pelo sistema operacional do seu pc, Caso isso não ocorra remova o adaptador de SD card da USB e insira novamente para que a partição boot esteja acessível. Acesse o site `https://github.com/wilsons1978/alpha/blob/master/Makefile`. A página será conforme a figura 3, clique no botão **Raw**. Em seguida clique com o botão direito do mouse (caso o seu mouse esteja configurado para canhoto, será o botão esquerdo) e selecione a opção "Salvar página como".Procure no computador que está usando um disco de nome **boot** e salve o arquivo Makefile neste local.

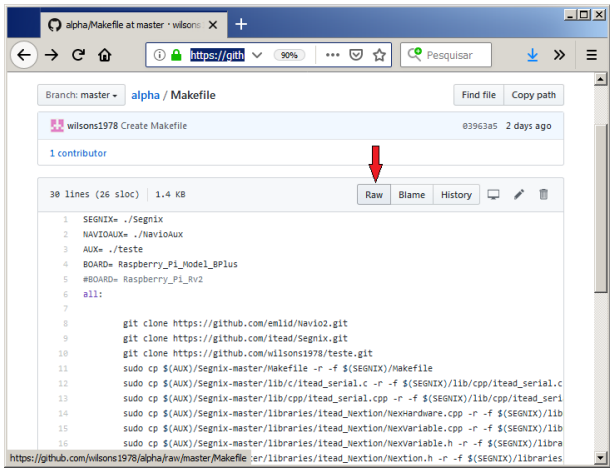


Figura 3: Página para download do script de instalação.

1.5 Inicializar Raspberry

Foram finalizadas as etapas de aquisição dos software e arquivos em seu computador. Será necessário colocar o cartão microSD na placa Raspberry. Conecte o monitor na placa Raspberry e conecte a fonte de alimentação na placa para colocar o Raspberry em execução.

1.5.1 Configuração com o comando raspi-config

Faça o login do sistema raspberry com os seguintes dados:

Usuário: **pi** Senha: **raspberry**

No prompt digite o comando:

sudo raspi-config

Ao ser executado este comando aparecerá a tela apresentada na figura 4. A primeira configuração a ser ajustada é a configuração da rede *wifi*.

- Selecione a opção 2 Network options, como na figura 5 e pressione **Enter**.
- Selecione a opção N2 Wi-fi, como na figura 6 e pressione **Enter**.
- Será aberta a tela de inseção do nome da rede Wi-fi, como na figura 7a e pressione **Enter**.
- Será aberta a tela de inserção da senha da rede Wi-fi, como na figura 7b, e pressione **Enter**.

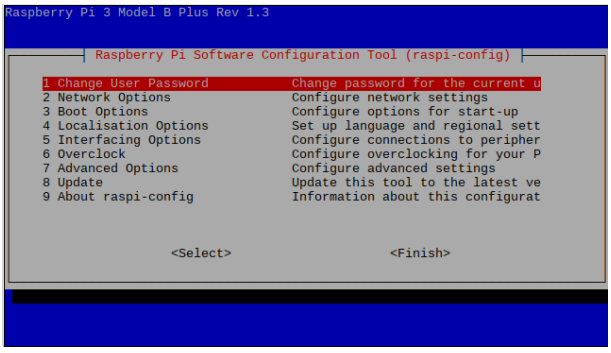


Figura 4: Tela inicial comando raspi-config.

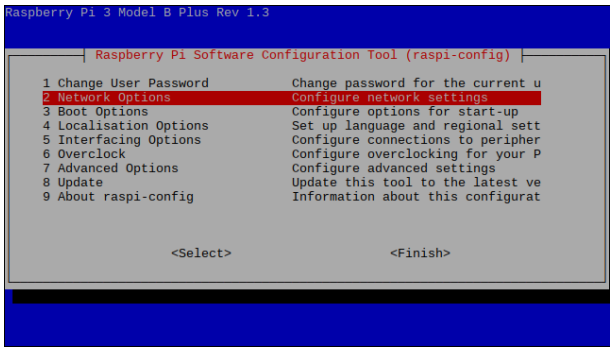


Figura 5: Seleção das opções de rede.

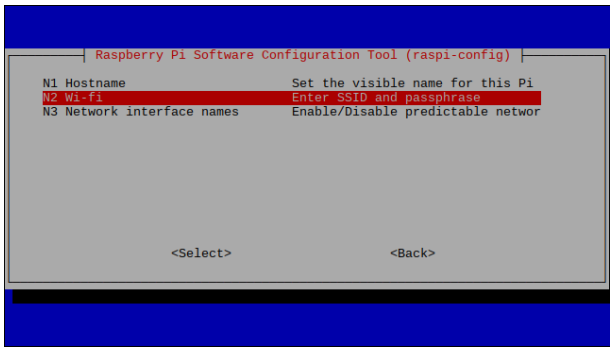
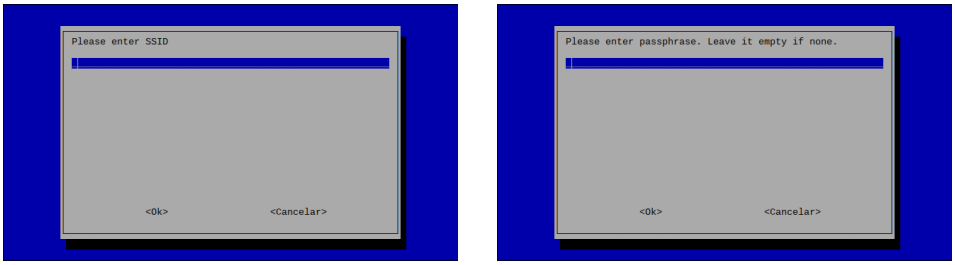


Figura 6: Seleção das configurações da rede Wi-fi.



(a) (b)

Figura 7: Inserção do nome e da senha da rede Wi-fi.

Segue assim a configuração da interface **SSH**. De volta a tela inicial:

- Selecione a opção 5 Interfacing options, como na figura 8 e pressione **Enter**.
- Selecione a opção P2 SSH, como na figura 9a. Trata -se de um protocolo para acessar serviços de rede, que será usado para o acesso remoto ao sistema Raspberry. Pressione **Enter**.
- Será aberta a tela para habilitar a execução do protocolo SSH, como na figura 9b, e pressione **Enter**.

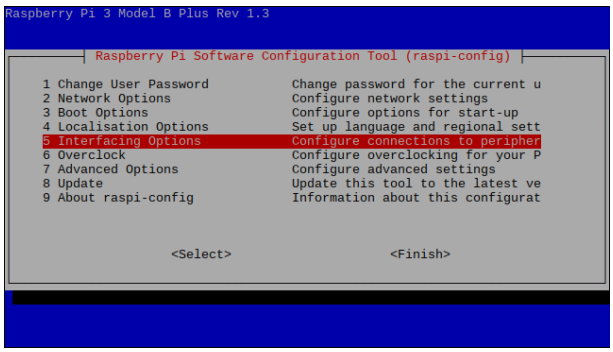
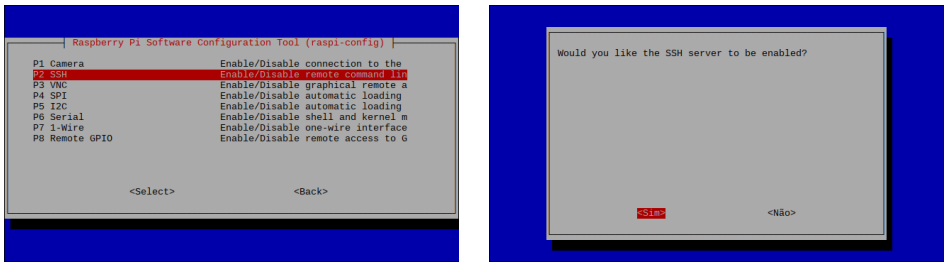


Figura 8: Seleção das opções de interface.

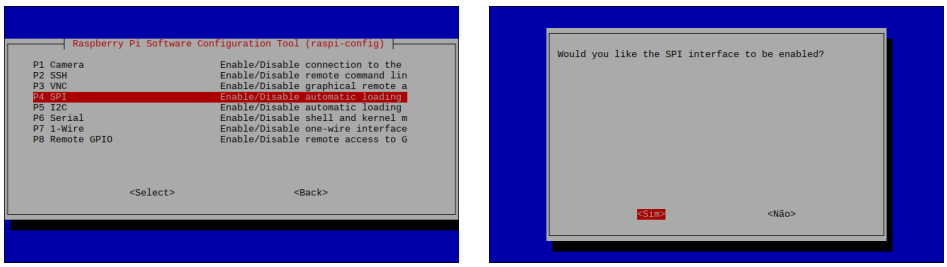


(a) (b)

Figura 9: Seleção e habilitação do protocolo SSH.

Segue assim a configuração da interface **SPI**. De volta a tela inicial:

- Selecione a opção 5 Interfacing options, como na figura 8 e pressione **Enter**.
- Selecione a opção P4 SPI, como na figura 10a, e pressione **Enter**.
- Será aberta a tela para habilitar a execução do protocolo SPI, como na figura 10b e pressione **Enter**.

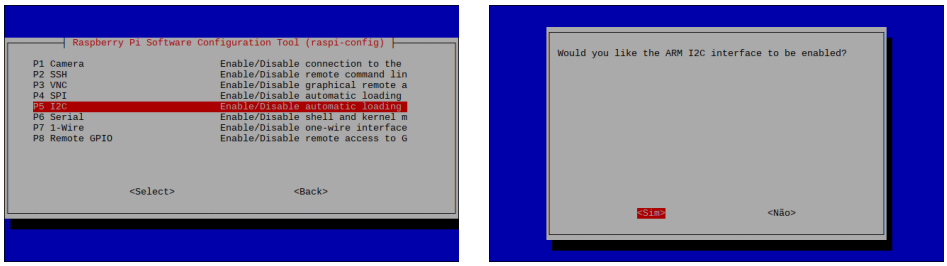


(a) (b)

Figura 10: Seleção e habilitação do protocolo SPI.

Segue assim a configuração da interface **I2C**. De volta a tela inicial:

- Selecione a opção 5 Interfacing options, como na figura 8, e pressione **Enter**.
- Selecione a opção P5 I2C, como na figura 11a, e pressione **Enter**.
- Será aberta a tela para habilitar a execução do protocolo I2C, como na figura 11b, e pressione **Enter**.

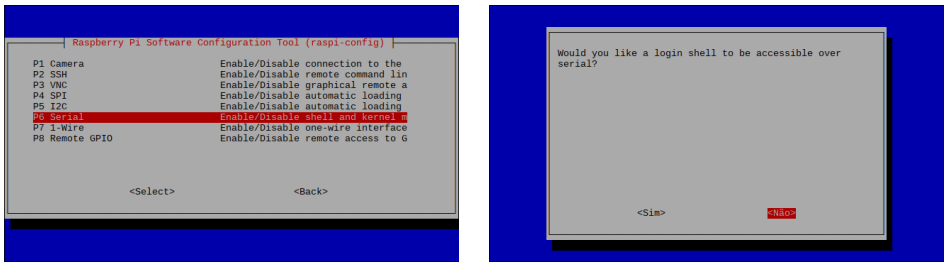


(a) (b)

Figura 11: Seleção e habilitação do protocolo I2C.

Segue assim a configuração da interface **UART**. De volta a tela inicial:

- Selecione a opção 5 Interfacing options, como na figura 8, e pressione **Enter**.
- Selecione a opção P6 Serial, como na figura 12a, e pressione **Enter**.
- Será aberta a tela para habilitar a execução da porta serial como console. Desabilite esta opção, como na figura 12b, e pressione **Enter**.
- Será aberta a tela para habilitar a execução da porta serial, como na figura 13, habilite esta opção e pressione **Enter**.



(a) (b)

Figura 12: Seleção e desabilitação da porta serial no modo console.

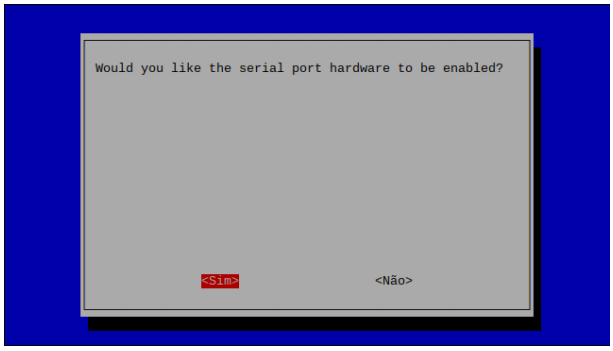


Figura 13: Seleção e habilitação da porta serial.

Segue assim a configuração da interface **I2C**. De volta a tela inicial:

- Selecione a opção 7 Advanced Options, como na figura 14, e pressione **Enter**.
- Selecione a opção A1 Expand filesystem, como na figura 15a, e pressione **Enter**.
- Será aberta a tela apresentando o progresso na expansão da partição, como na figura 15b, quando estiver concluído pressione **Enter**.

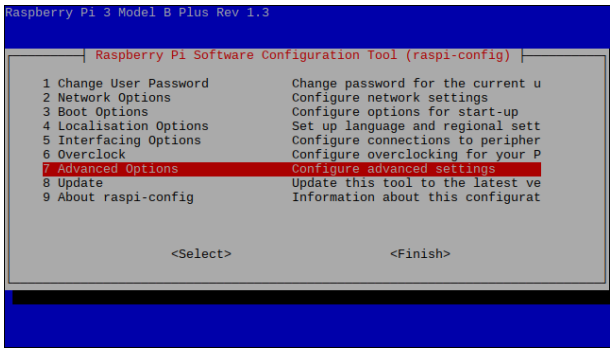
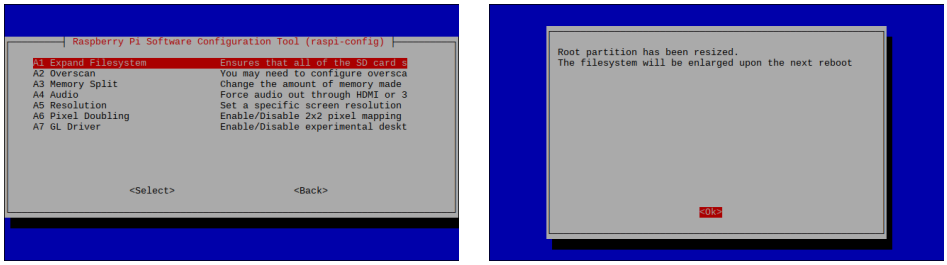


Figura 14: Seleção de opções avançadas.



(a) (b)

Figura 15: Seleção de Expansão da partição Linux e expansão finalizada.

Finalizadas as configurações, selecione a opção **Finish** com as setas laterais e pressione **Enter**. A tela perguntará se deseja reiniciar o sistema, selecione não, conforme a figura 16. Caso isso não ocorra, ignore este passo. Desligue o sistema usando o comando:

```
sudo shutdown -h now
```

Aguarde por volta de 20 segundos, desligue a alimentação da placa Raspberry. Aguarde novamente por 20 segundos e religue a alimentação para que o sistema reinicialize.

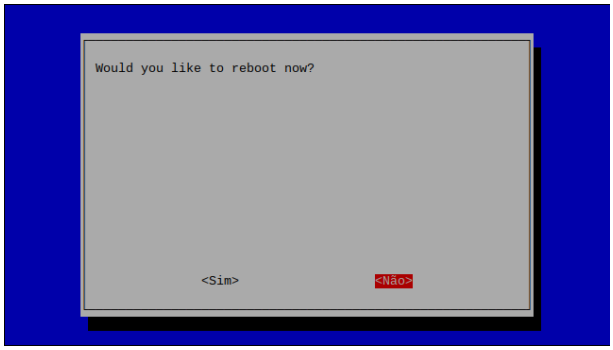


Figura 16: Finalizando e salvando as configurações.

1.6 Preparar acesso remoto

Executados os passos anteriores, faça o login do sistema sub-seção 1.5.1. Execute o comando:

```
ifconfig
```

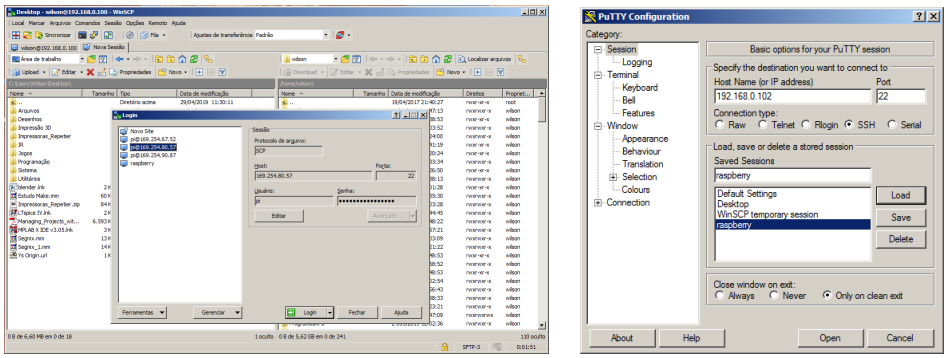
A resposta do comando poderá ser conforme apresentada na figura 17. Como se pode ver o sistema Raspberry possui três tipos de conexões:

- **eth0** : conexão de rede a cabo que neste caso está conectada.
- **lo** : conexão de loopback usada para testes.
- **wlan** : conexão de Wi-fi(na figura wlan0) com o endereço 192.168.0.104.

```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether b8:27:eb:39:80:52  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Loopback Local)
        RX packets 9  bytes 524 (524.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9  bytes 524 (524.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.104  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::7945:e1b8:5032:f184  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:6c:d5:07  txqueuelen 1000  (Ethernet)
        RX packets 12348  bytes 8709345 (8.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8820  bytes 1534373 (1.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~ $
```

Figura 17: Resposta ao comando ifconfig.

Instale em seu computador os softwares **Winscp** e **Putty** para acessar remotamente os sistema Raspberry. Utilize o endereço obtido ao executar o comando ifconfig. O software Winscp será utilizado para acessar os arquivos do sistema Raspberry. A tela inicial é apresentada na figura 18a. O software **Putty** permite a execução de comandos em uma interface por linha de comando. A tela inicial é apresentada na figura 18b.



(a) (b)

Figura 18: Softwares usados para acesso remoto.

Para ambos os softwares o login e senha são exatamente os apontados na subseção 1.5.1. Com o login executado use o comando para criar a pasta **C1** (caso desse, utilize outro nome para o arquivo substitua o C1 pelo nome escolhido):

```
mkdir C1
```

Com a pasta criada execute os seguintes comandos em sequência:

```
sudo mv /boot/Makefile ./C1
cd C1
sudo make
```

Com isso se inicia a instalação do software necessário para o desenvolvimento do PA. Quando for requisitado o login para o site github.com use os seguintes dados:

Usuário: **wilsons1978**
Senha: **pl@n@d0r**

2 Estrutura de arquivos

A instalação resulta na seguinte estrutura de arquivos na pasta de trabalhos.

- Pasta **Navio2**: possui arquivos para acessar os recursos da placa Navio2 e alguns exemplos.
- Pasta **NavioAux**: possui arquivos auxiliares para facilitar o acesso aos recursos da placa Navio2 e da tela Nextion-Segnix.
- Pasta **Segnix**: possui os arquivos para compilar diversos drivers para dispositivos da Segnix, entre eles a tela Nextion.
- Arquivo **main.cpp**: Arquivo fonte colocado como exemplo de umaimplementação simples de um controlador PID com interface com a tela Nextion.
- Arquivo **Makefile**: script de automatização da compilação de todos os arquivos-fonte usados para gerar um executável com acesso aos recursos da palca Navio2 e da ela Nextion.

2.1 Pasta Navio2

A pasta Navio2 possui quatro pastas e quatro arquivos que são:

- Pasta **.git**: possui configurações e registros do pacote git.
- Pasta **C++**: possui arquivos-fonte em linguagem C++, usados para acessar o hardware da placa Navio2 e exemplos de uso.
- Pasta **Python**: possui arquivos-fonte em linguagem Python, usados para acessar o hardware da placa Navio2 e exemplos de uso.
- Pasta **Utilities**: possui arquivos-fonte em linguagem C e Python para o sistema Raspeberry+placa Navio2 comunique com o pc do usuário.
- Arquivos **.gitignore**,**CONTRIBUTING.md**,**LICCENSE** e **README.md**: arquivos do pacote git e licenças de uso.

A pasta mais relevante é a pasta **C++** que possui duas pastas que são:

- **Navio**: possui três pastas e um arquivo que são:
  - Pasta **Common** para arquivos comuns aos modelos Navio+ e Navio2.
  - Pasta **Navio+** para arquivos do modelo Navio+.
  - Pasta **Navio2** para arquivos do modelos Navio2.
  - Arquivo **Makefile** script de compilação dos arquivos fonte para gerar a biblioteca que acessa os recursos das placas Navio+ ou Navio2.
- **Examples**: Possui diversos exemplos de utilização dos recursos da placa Navio+ ou navio2 pelo sistema Raspberry.Possui as seguintes pastas:
  - Pasta **AccelGyroMag**: exemplo de código para leitura do acelerômetro, giroscópio e magnetômetro da unidade inercial (IMU) desejada[1].
  - Pasta **ADC**: exemplo de código para a leitura dos canais do conversor analógico-digital, os valores são apresentados em volts.
  - Pasta **AHRS**: exemplo utilizando o algorítmo Mahony AHRS tendo como fonte de dados as unidades inerciais MPU9250 ou LSM9DS1. Tem como saída os valores de *roll*, *pitch* e *yaw* no console e envia pela rede os dados do quaternio.[2]
  - Pasta **Barometer**: exemplo de código para a leitura do barômetro.
  - Pasta **FRAM**: exemplo de código para a leitura e escrita da memória RAM não volátil no modelo Navio2.
  - Pasta **GPS**: exemplo de código para a leitura dos dados do GPS.
  - Pasta **LED**: exemplo de código para acionar o LED RGB presente na placa Navio+ e Navio2.
  - Pasta **Multithread**: exemplo de código para a leitura do barômetro utilizando multi-thread para fazer a leitura em loop de execução independente do loop principal.
  - Pasta **RCInput**: exemplo de código para a leitura do canal PPM usado para receber comandos PWM do receptor.
  - Pasta **Servo**: exemplo de código para acionar as saídas PWM da placas Navio+ ou Navio2.
  - Arquivo **Makefile**: Script que auxilia a compilação dos arquivos de exemplo.

2.2 Pasta NavioAux

A pasta NavioAux possui os seguintes arquivos:

- **tela_var.cpp** e **tela_var.hpp**: implementa o acesso aos parâmetros apresentados e recebidos pela tela *touchscreen*.As variáveis e funções implementadas são:
  - Parâmetro de configuração **setpoint_lat** : Setpoint da latitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **setpoint_lon** : Setpoint da longitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **latp** : Setpoint da ganho proporcional do controlador da latitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **lati** : Setpoint da ganho integrativo do controlador da latitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **latd** : Setpoint da ganho derivativo do controlador da latitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **lonp** : Setpoint da ganho proporcional do controlador da longitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **loni** : Setpoint da ganho integrativo do controlador da longitude ajustado na tela *touch-screen*.
  - Parâmetro de configuração **lond** : Setpoint da ganho derivativo do controlador da longitude ajustado na tela *touch-screen*.
  - Valor de leitura da Navio2 **errorlat** : erro da latitude.
  - Valor de leitura da Navio2 **errorlon** : erro da longitude.

[1] use o comnado sudo ./AccelGyroMag **mpu** para MPU9250 ou **lsm** pra LSM9DS1

[2] Isto pode ser usado com o visualizador 3D para IMU localizado na pasta Navio2/Utilities/3DIMU

  - Valor de leitura da Navio2 **flag0** : flag auxiliar.
  - Valor de leitura da Navio2 **flagpiloto** : flag de controle do piloto automático (0:stand by, 1:automatico e 2:manual).
  - Valor de leitura da Navio2 **latnavio2** : valor de latitude medida.
  - Valor de leitura da Navio2 **lonnavio2** : valor de longitude medida.
  - Valor de leitura da Navio2 **leituraadc1** : Valor de leitura canal 1 do conversor analógico-digital.
  - Valor de leitura da Navio2 **leituraadc2** : Valor de leitura canal 2 do conversor analógico-digital.
  - Função de inicialização da tela *touchscreen* **init_tela** : possui como parâmetro o endereço de uma variável **tela_var** (exemplo: init_tela(&tela)).

- **timer.cpp** e **timer.h**: implementa classe auxiliar **timer** para tratar com intervalos de tempo.
  - Função **get_period**: retorna ao tempo decorrido desde a aquisição da referência (tempo dados em segundos).
  - Função **forced_loop_interval**: mantém o fluxo de execução em loop até que o tenha transcorrido o intervalo de tempo dado como parâmetro da função (intervalo dado em microssegundos) .
  - Função **get_reference**: retorna o valor da referência utilizada para a comparação do tempo transcorrido.
  - Função **set_reference**: define o valor de parâmetro de entrada como o valor da refêrencia utilizada para a comparação do tempo transcorrido.

- **imu.cpp** e **imu.h**: implementa classe auxiliar **imu** para acessar as unidades inerciais e gerar dados de velocidade e posição tanto linear quanto na rotação (**não implementado**).

2.3 Pasta Segnix

Possui os arquivos utilizados pela tela *touch-screen*. Necessária durante a instalação da tela e acesso aos arquivos de *include* *.h presentes na subpasta **include**.

2.4 Arquivo Makefile

Segue abaixo a listagem do arquivo Makefile.

```
CXX ?= g++
NAVIO = ./Navio2/C++/Navio
NAVIOAUX = ./NavioAux
PIGPIO_PATH := $(PIGPIO_PATH)
#LIB = -L$(PIGPIO_PATH)
SEGNIX= ./Segnix
INCLUDES := -I ./Navio2/C++/Navio
INCLUDES += -I ./NavioAux
#INCLUDES += -I$(PIGPIO_PATH)
INCLUDES += -I ./Segnix/libraries/itead_Nextion
INCLUDES += -I/usr/include/python2.7 -I/usr/include/arm-linux-gnueabihf/python2.7
LINKERDATA= -L/usr/lib/python2.7/config-arm-linux-gnueabihf -L/usr/lib
-lpython2.7 -ldl -lutil -lm -Xlinker -export-dynamic -Wl,-O1 -Wl,-Bsymbolic-functions
LIB_NEXTION= -liteadc -liteadcpp -liteadmodule
PIGPIO_DO =-lpigpio # || $(MAKE) pigpio
all:
        $(MAKE) -C ./Navio2/C++/Navio all
        $(CXX) -std=c++11 $(INCLUDES) $(LIB) main.cpp $(NAVIOAUX)/timer.cpp
        $(NAVIOAUX)/tela_var.cpp $(NAVIOAUX)/imu.cpp -L$(NAVIO) -lnavio
        $(LIB_NEXTION) -lrt -lpthread -o Pendulum $(LINKERDATA) $(PIGPIO_DO)
clean:
        rm -f Pendulum
```

Abaixo temos uma descrição linha a linha:

1. `CXX ?= g++` : esta linha define a variável **CXX** com a string **g++** que é a definição do compilador a ser utilizado, ou seja, um compilador para C++.

2. NAVIO = ./Navio2/C++/Navio : define a pasta onde está o código fonte dos drivers para acessar os recursos da placa Navio2.
3. NAVIOAUX = ./NavioAux : .
4. PIGPIO_PATH := $(PIGPIO_PATH) :.
5. #LIB = -L$(PIGPIO_PATH) : comando de compilação necessário para placa Navio+.
6. SEGNIX= ./Segnix :.
7. INCLUDES := -I ./Navio2/C++/Navio :.
8. INCLUDES += -I ./NavioAux :.
9. #INCLUDES += -I$(PIGPIO_PATH) : comando de compilação necessário para placa Navio+.
10. INCLUDES += -I ./Segnix/libraries/itead_Nextion :.
11. INCLUDES += -I/usr/include/python2.7-I/usr/include/arm-linux-gnueabihf/python2.7 :.
12. LINKERDATA= -L/usr/lib/python2.7/config-arm-linux-gnueabihf -L/usr/lib :.
13. -lpython2.7 -ldl -lutil -lm -Xlinker -export-dynamic -Wl,-O1 :.
14. -Wl,-Bsymbolic-functions :.
15. LIB_NEXTION= -liteadc -liteadcpp -liteadmodule :.
16. PIGPIO_DO =-lpigpio :.
17. # || $(MAKE) pigpio : comando de compilação necessário para placa Navio+.
18. all: :.
19. $(MAKE) -C ./Navio2/C++/Navio all :.
20. $(CXX) -std=c++11 $(INCLUDES) $(LIB) main.cpp $(NAVIOAUX)/timer.cpp :.
21. $(NAVIOAUX)/tela_var.cpp $(NAVIOAUX)/imu.cpp -L$(NAVIO) -lnavio :.
22. $(LIB_NEXTION) -lrt -lpthread -o Pendulum $(LINKERDATA) $(PIGPIO_DO) :.
23. clean: :.
24. rm -f Pendulum :.