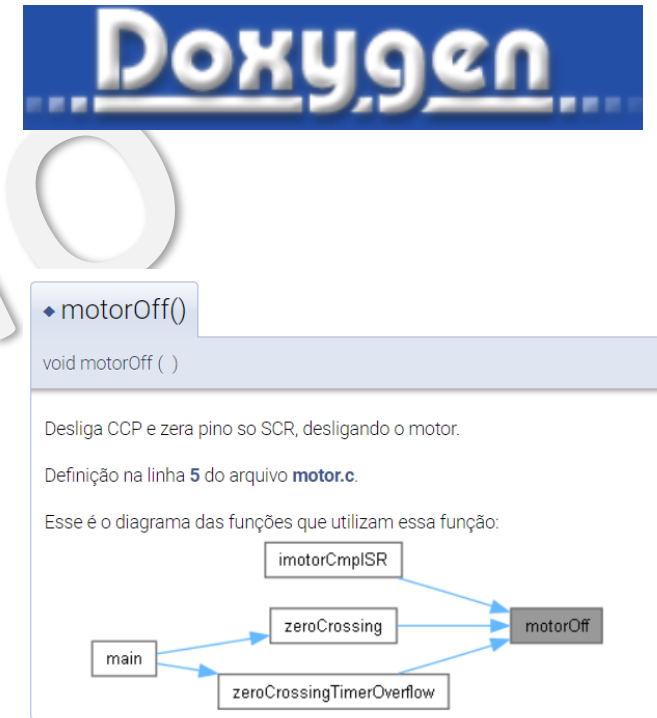


# Documentação do firmware em projetos de sistemas embarcados

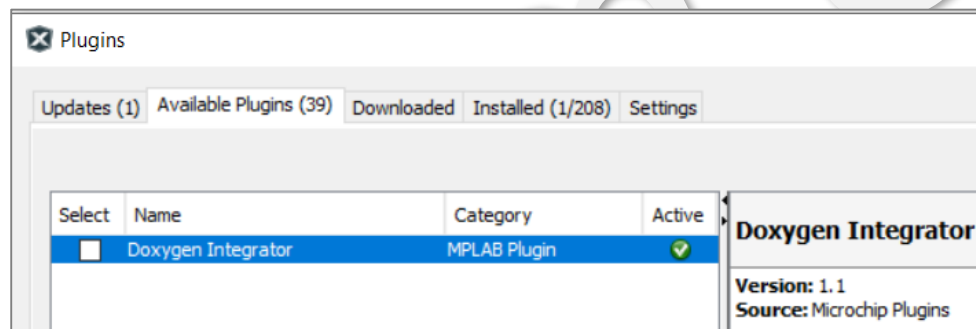
- Existem várias opções para documentar o código desenvolvido para os sistema embarcado. Algumas já estão integradas no IDE, outras são plugin que se instalam e se integram ao IDE e outras são externas ao IDE.
- Um opção muito usada é o Doxygen, baseado em código aberto e disponível no Github com vasta documentação.
- O Doxygen é multiplataforma e em conjunto com outras aplicações consegue gerar textos e diagramas de relações entre funções, variáveis, classes e arquivos, em diferentes codificações de texto e formatos HTML, PDF e RTF.
- A maior parte do texto gerado se baseia nos comentários colocados pelo desenvolvedor no mesmo código, usando as marcas especificadas (não são os mesmos comentários que colocamos nas linhas de código).
- Os diagramas se baseiam nas análises que ele mesmo faz da estrutura do código.



# Documentação do firmware em projetos de sistemas embarcados

- Para usar em nossos projetos precisamos fazer download do Doxygen e instalar em nosso computador, conforme o link <https://www.doxygen.nl/manual/install.html> . Anotem a pasta onde está instalado, precisaremos disso depois.
- Também precisaremos do Graphviz para conseguir criar os diagramas. O instalador pode ser encontrado no link <https://www.graphviz.org/download/>
- Para usar fórmulas e criar documentação em PDF, será necessário instalar o Latex e o Ghostscript. No curso usaremos apenas em HTML.
- Finalmente precisaremos instalar o plug-in Doxygen Integrator no MPLABX, no menu Tools/Plugins, aba Available Plugins.

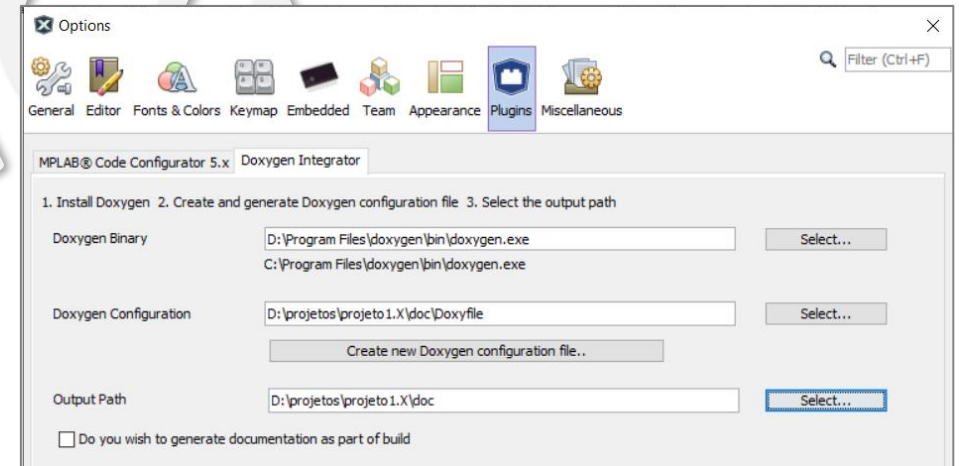
Graphviz



- Outra opção é executar o Doxygen diretamente por comandos. Isso tem algumas vantagens como vamos ver. Nesse caso não precisa instalar o plugin no MPLABX.

# Documentação do firmware ...

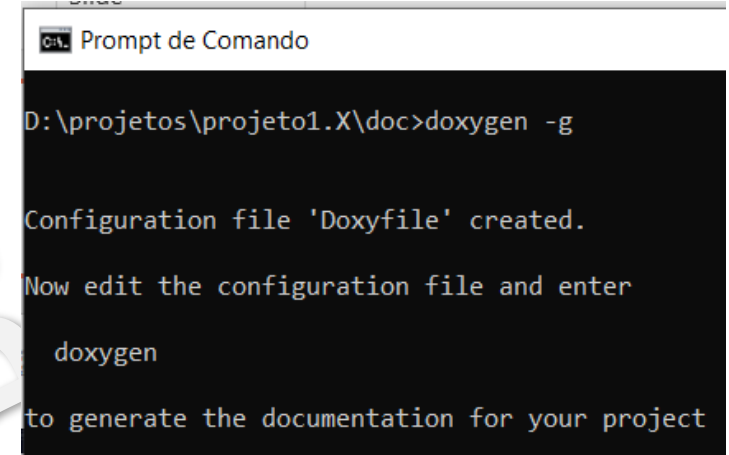
- Em seguida, será necessário configurar o plugin no MPLABX (se não foi instalado, pular para o próximo slide). Para isso abrimos no menu **Tools/Options/Plugins** e clicamos na aba **Doxygen Integrator**.
- Na janela, na opção **Doxygen Binary**, devemos colocar o caminho de instalação do Doxygen, incluindo a pasta bin e o executável (por exemplo em Windows pode ser *C:\Program Files\doxygen\bin\doxygen.exe*)
- Agora devemos definir ou criar a pasta onde será gerada a documentação. Uma recomendação é criar, dentro da pasta do projeto, uma pasta com nome *doc* para isso. O Doxygen criará uma pasta *html* dentro dessa que foi definida e colocará todos os arquivos dentro.
- Para cada projeto devemos criar um novo arquivo de configuração (Doxygen Configuration) que atualizaremos mais tarde, ou copiar um arquivo com a configuração pronta, usado em outro projeto. O arquivo de configuração (Doxyfile) pode ser criado ou copiado na mesma pasta onde será guardada a documentação (doc).
- Depois definimos na opção Output Path o caminho da pasta onde será guardada a documentação.
- Clicamos em OK e procedemos a editar o arquivo de configuração.
- Infelizmente, toda vez que mudemos de projeto, precisaremos atualizar esses dados, embora o arquivo Doxyfile já exista no projeto.



# Documentação do firmware ...

- Para usar Doxygen por linha de comandos, vamos abrir uma janela preta (consola) e nos deslocaremos para a pasta onde será gerada a documentação.
- Caso se deseje usar o Doxygen por linha de comandos em Windows, o caminho do executável deve acrescentado no PATH do sistema operacional. No link a seguir se explica como fazer <https://www.computerhope.com/issues/ch000549.htm>.
- Uma recomendação é criar, dentro da pasta do projeto, uma pasta com nome *doc* para isso. O Doxygen criará uma pasta *html* dentro dessa que foi definida e colocará todos os arquivos dentro.
- Em seguida executaremos o comando ***doxygen -g*** na pasta do projeto, para criar o arquivo de configuração *Doxyfile*, que será editado a continuação.

## Windows



```
C:\> Prompt de Comando

D:\projetos\projeto1.X\doc>doxygen -g

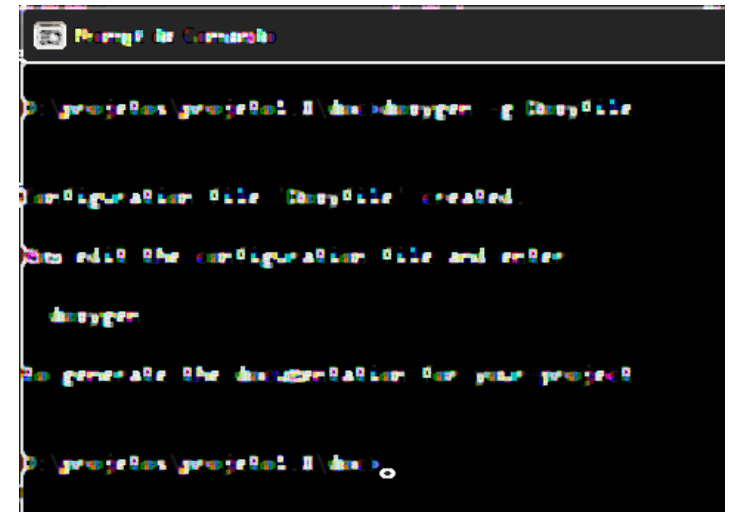
Configuration file 'Doxyfile' created.

Now edit the configuration file and enter

doxygen

to generate the documentation for your project
```

## Linux



```
D:\projetos\projeto1.X\doc>doxygen -g Doxyfile

Configuration file 'Doxyfile' created.

Now edit the configuration file and enter

doxygen

to generate the documentation for your project

D:\projetos\projeto1.X\doc>
```

# Documentação do firmware ...

- No arquivo *Doxyfile* devemos definir vários parâmetros que afetarão a geração de arquivos. A seguir as telas com as modificações necessárias para gerar arquivos HTML e digramas com menus e textos em Português do Brasil.

```
# Doxyfile 1.8.13

#-----
# Project related configuration options
#-----
DOXYFILE_ENCODING      = UTF-8
PROJECT_NAME           = projeto1
PROJECT_NUMBER         =
PROJECT_BRIEF          =
PROJECT_LOGO           =
OUTPUT_DIRECTORY       = D:\projetos\projeto1.X\doc
CREATE_SUBDIRS         = NO
ALLOW_UNICODE_NAMES    = NO
OUTPUT_LANGUAGE        = Brazilian
BRIEF_MEMBER_DESC      = YES
REPEAT_BRIEF           = YES
```

```
#-----
# Configuration options related to the input files
#-----
INPUT                  = D:\projetos\projeto1.X\doc
INPUT_ENCODING         = ISO-8859-1
FILE_PATTERNS          = *.c \
```

A explicação de cada parâmetro pode ser encontrada na documentação d Doxygen e do Graphviz.

```
#-----
# Configuration options related to the HTML output
#-----
GENERATE_HTML          = YES
HTML_OUTPUT            = html
HTML_FILE_EXTENSION    = .html
```

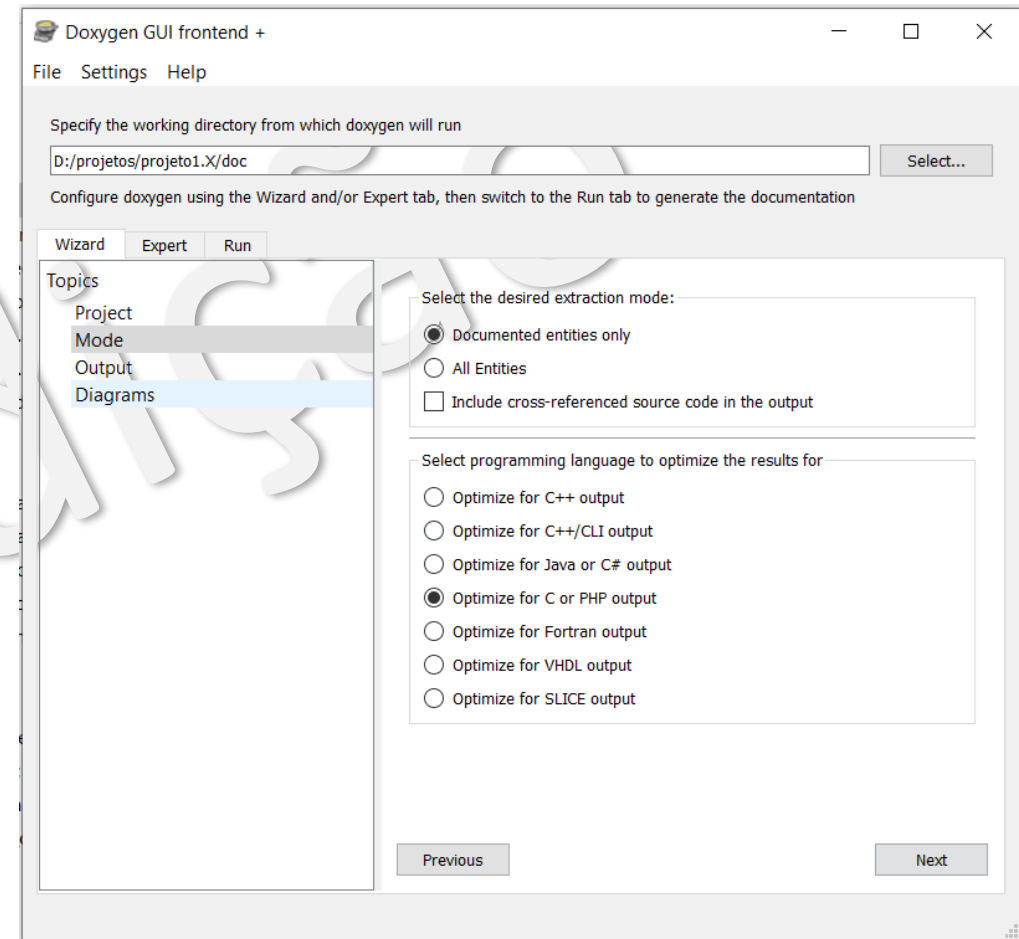
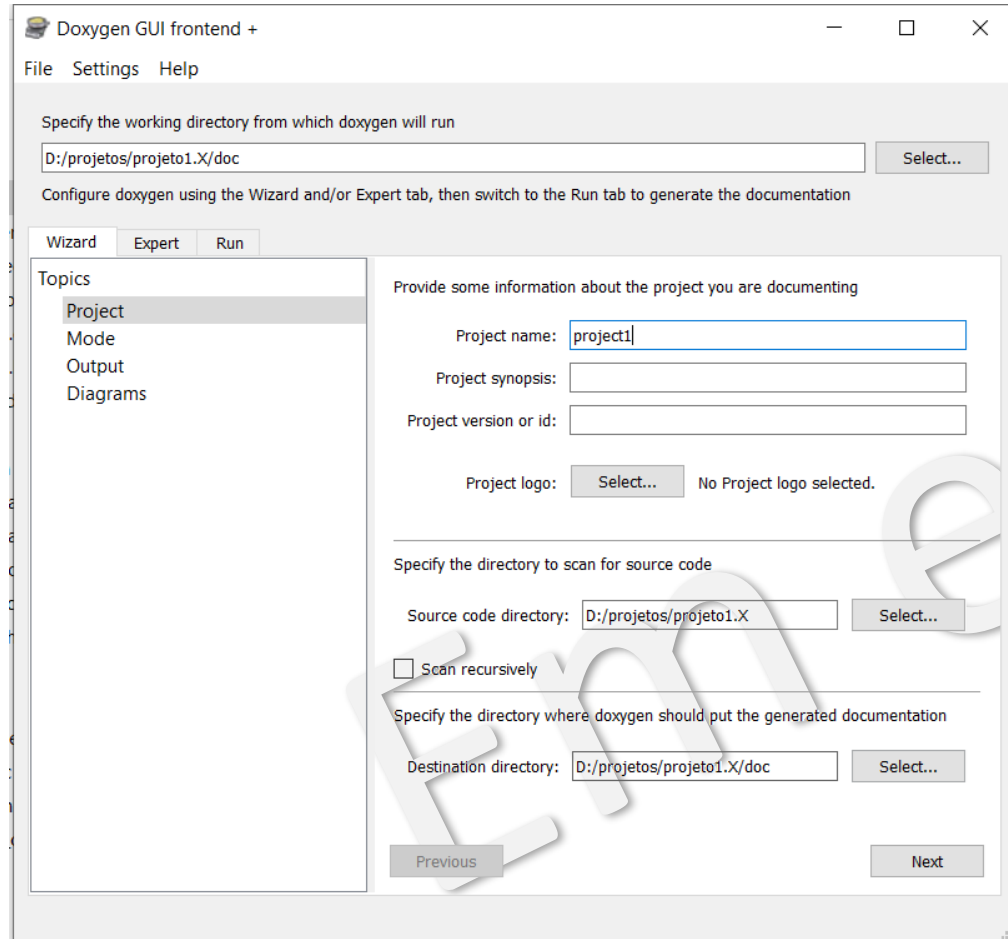
```
#-----
# Configuration options related to the LaTeX output
#-----
GENERATE_LATEX         = NO
LATEX_OUTPUT           = latex
LATEX_CMD_NAME         = latex
```

```
#-----
# Configuration options related to the dot tool
#-----
CLASS_DIAGRAMS         = YES
MSCGEN_PATH            =
DIA_PATH               =
HIDE_UNDOC_RELATIONS   = YES
HAVE_DOT               = YES
DOT_NUM_THREADS        = 0
DOT_FONTNAME           = Helvetica
DOT_FONTSIZE           = 10
DOT_FONTPATH           =
CLASS_GRAPH            = YES
COLLABORATION_GRAPH    = YES
GROUP_GRAPHS          = YES
UML_LOOK               = NO
UML_LIMIT_NUM_FIELDS   = 10
TEMPLATE_RELATIONS     = NO
INCLUDE_GRAPH          = YES
INCLUDED_BY_GRAPH      = YES
CALL_GRAPH             = YES
CALLER_GRAPH           = YES
GRAPHICAL_HIERARCHY    = YES
DIRECTORY_GRAPH        = YES
DOT_IMAGE_FORMAT       = png
INTERACTIVE_SVG        = NO
DOT_PATH               =
DOTFILE_DIRS           =
MSCFILE_DIRS           =
DIAFILE_DIRS           =
PLANTUML_JAR_PATH      =
PLANTUML_CFG_FILE      =
PLANTUML_INCLUDE_PATH  =
DOT_GRAPH_MAX_NODES    = 50
MAX_DOT_GRAPH_DEPTH    = 0
DOT_TRANSPARENT        = NO
DOT_MULTI_TARGETS      = NO
GENERATE_LEGEND         = YES
DOT_CLEANUP            = YES
```

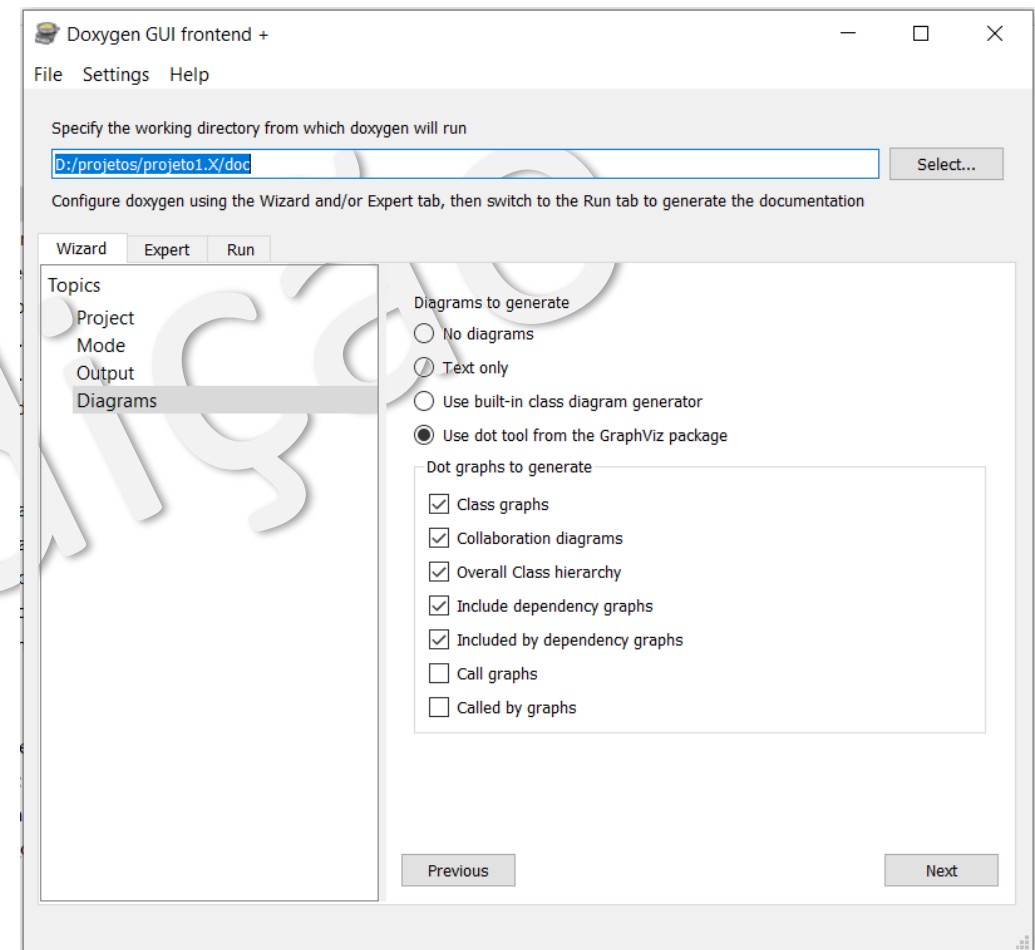
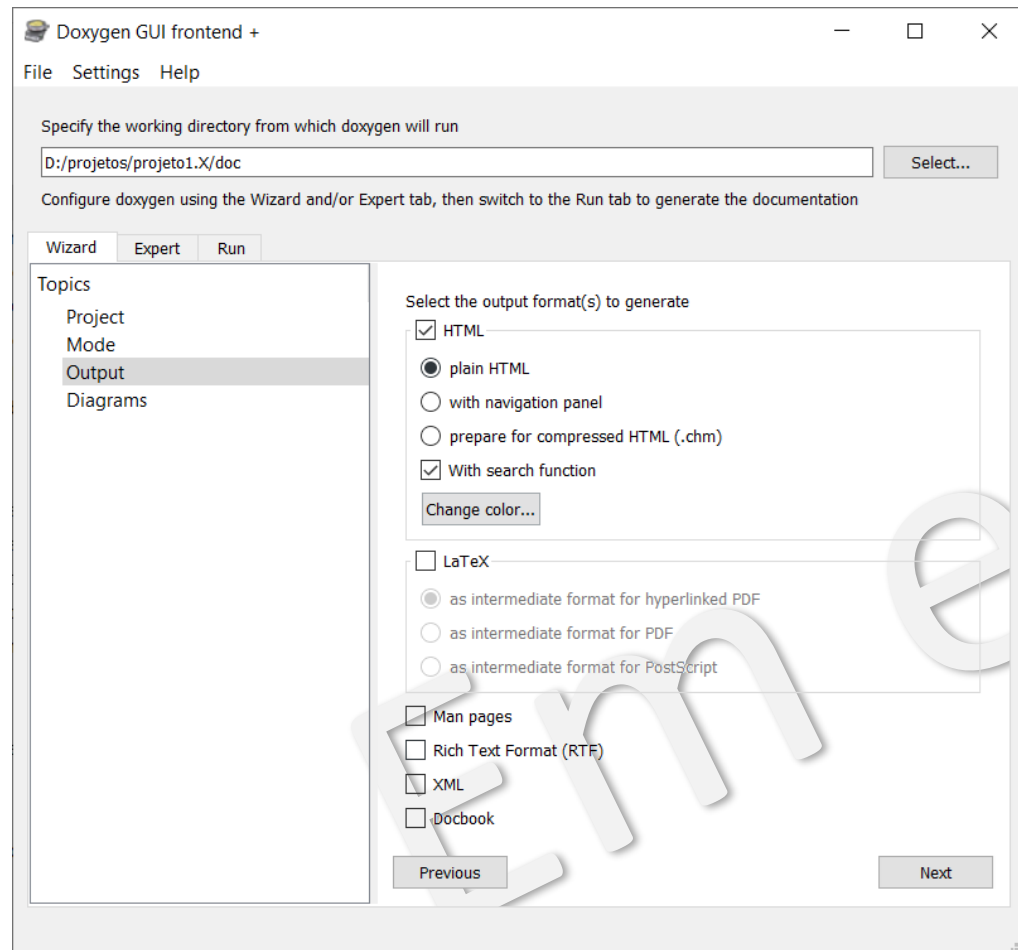


# Documentação do firmware ...

- Também pode ser usada a aplicação **Doxywizard**, instalado junto com o Doxygen, para fazer a criação e edição do arquivo de configuração. Coloco a seguir alguma telas como exemplo.

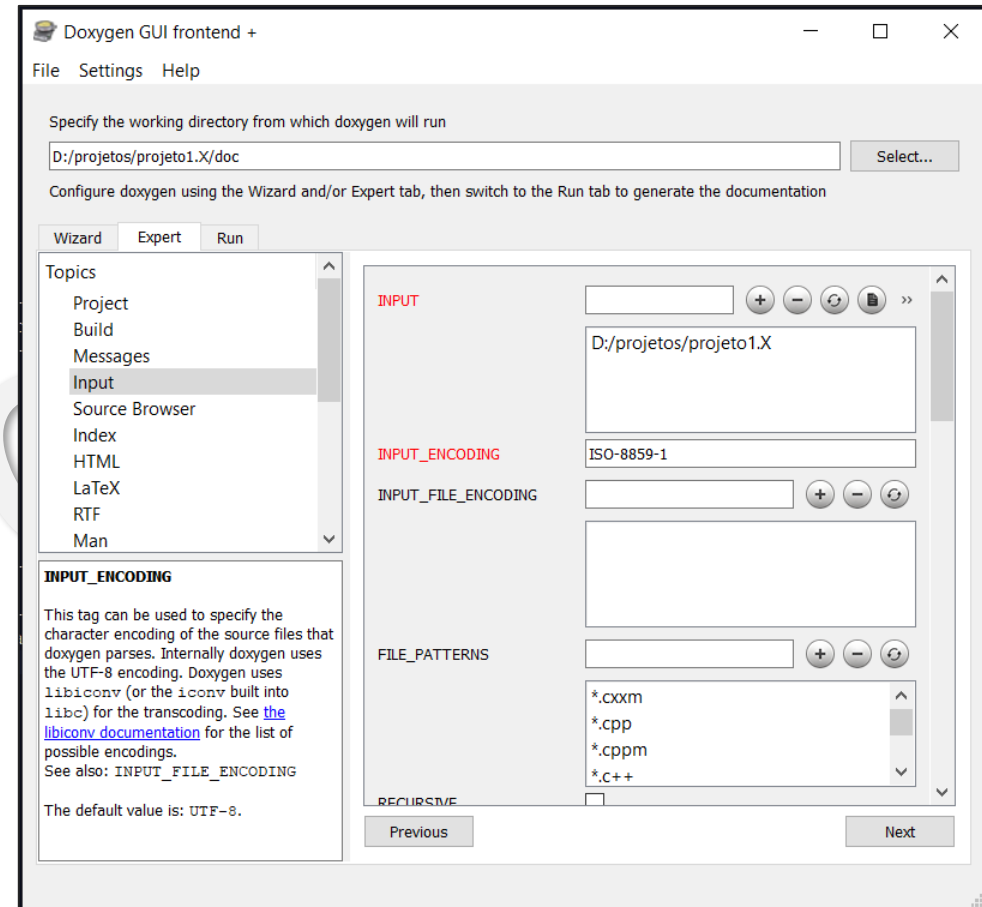
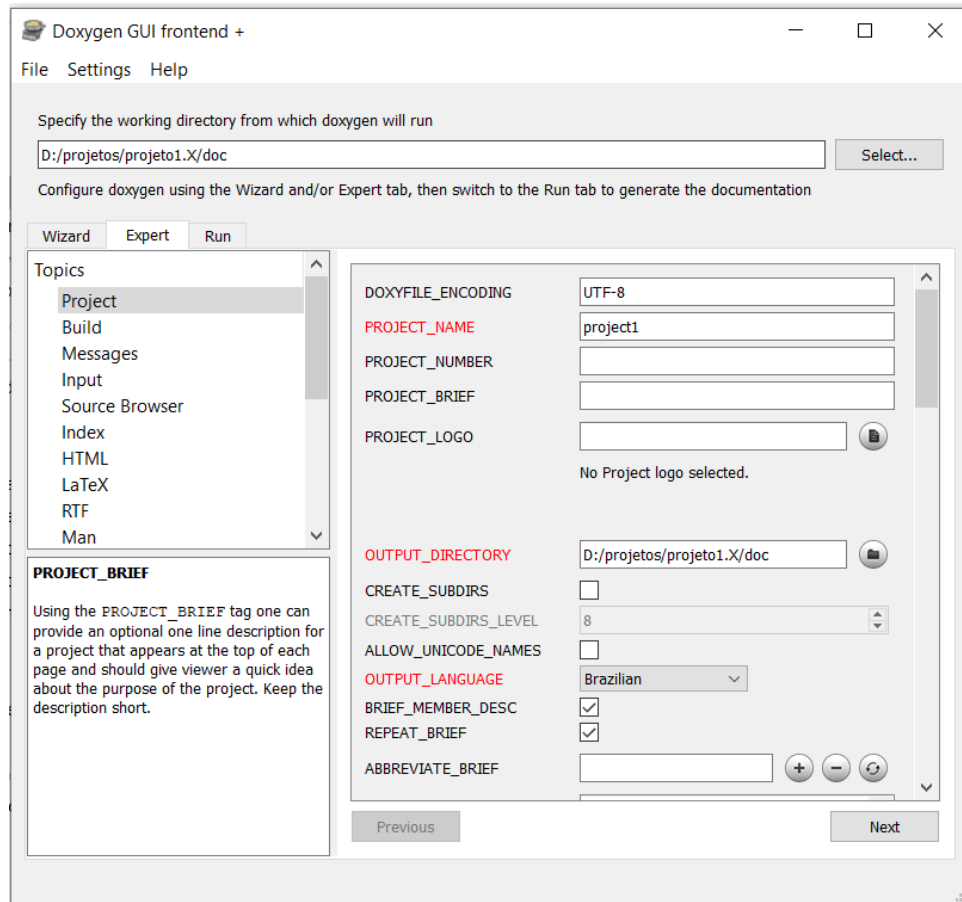


# Documentação do firmware ...



# Documentação do firmware ...

- Dependendo do projeto, será necessário fazer algumas mudanças no modo **experto** do **Doxywizard**.





# Documentação do firmware ...

- Agora estamos prontos para documentar o projeto. Para isso, no arquivo .h, colocamos antes de cada função `/**` e damos Enter. Deve aparecer uma estrutura como a que se exibe a seguir, onde colocaremos um texto descritivo do que faz a função e de cada parâmetro de entrada e saída.

```
/**  
 * Desliga CCP e zera pino so SCR, desligando o motor.  
 */  
void motorOff();
```

```
/**  
 * Filtra as medições de um sensor e converte para unidades de engenharia.  
 * @param sensor  
 * @return Valor em unidades de engenharia.  
 */  
float HX711_GetMeasure(HX711_sensor_e sensor);
```

# Documentação do firmware ...

- Também conseguimos documentar variáveis simples, estruturas e constantes.

```
/**
 * Identificador dos circuitos HX711.
 */
typedef enum{
    hx711_sensor1,    ///< Sensor e circuito 1.
    hx711_sensor2     ///< Sensor e circuito 2.
}HX711_sensor_e;

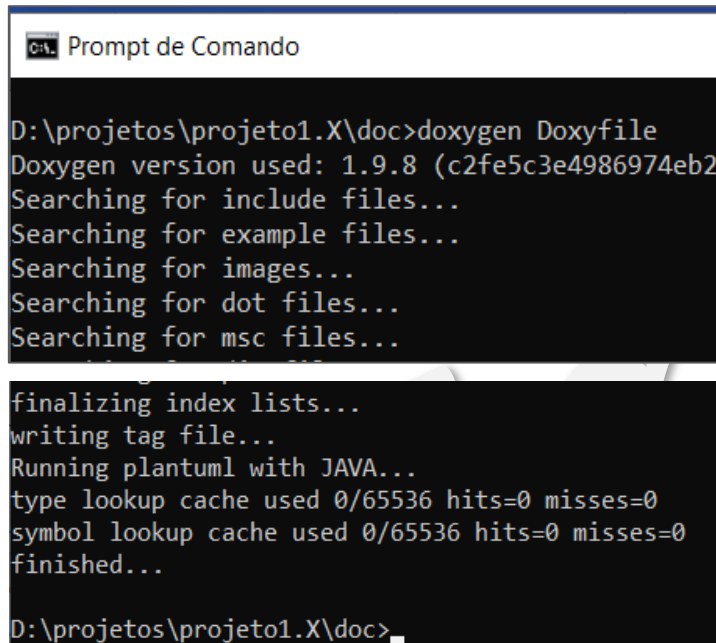
#define HX711_FILTER 10 ///< Número de elementos do filtro de janela móvel usado para promediar as medições dos HX711.

/**
 * Estrutura usada armazenar os dados dos sensores de carga e deformação.
 */
typedef struct{
    int24_t    offset;    ///< Offset usado na conversão para unidades de engenharia.
    float      gain;      ///< Ganho usado na conversão para unidades de engenharia.
    int24_t    adc[HX711_FILTER]; ///< Arranjo para armazenar os valores do filtro de janela móvel.
}hx711_sensor_t;
```

- Outras formas de documentação podem ser estudadas no link <https://www.doxygen.nl/manual/docblocks.html>

# Documentação do firmware ...

- Para gerar a documentação usando o plugin do MPLABX devemos clicar acima do nome do projeto com o botão direito do mouse e selecionar a opção **Create Doxygen**. O MPLABX exibirá uma mensagem na aba Output quando concluir a geração.
- Caso se use a linha de comandos, deve ser executado o comando **doxygen Doxyfile** na pasta do projeto
- Se usamos o **Doxywizard** seria apenas avançar até a aba Run e clicar em **Run doxygen**.



```

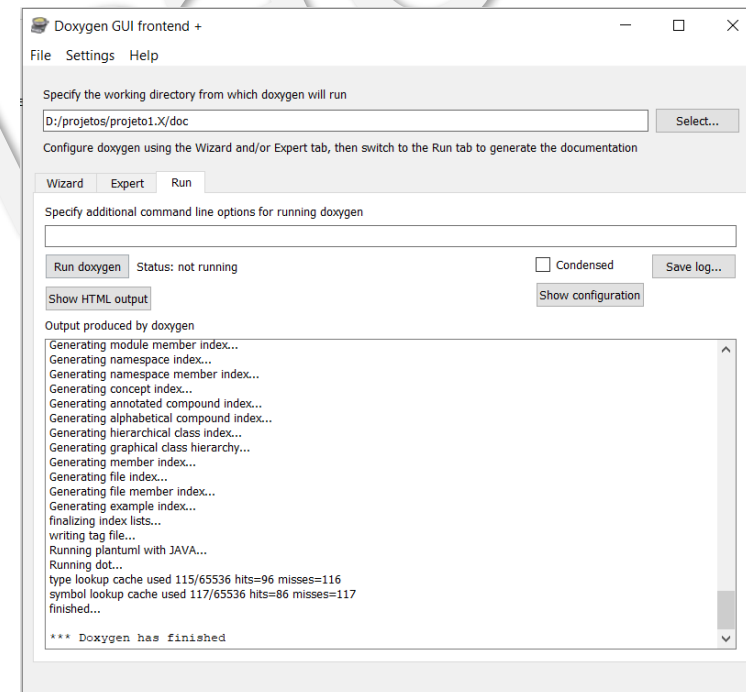
Prompt de Comando

D:\projetos\projeto1.X\doc>doxygen Doxyfile
Doxygen version used: 1.9.8 (c2fe5c3e4986974eb2)
Searching for include files...
Searching for example files...
Searching for images...
Searching for dot files...
Searching for msc files...

finalizing index lists...
writing tag file...
Running plantuml with JAVA...
type lookup cache used 0/65536 hits=0 misses=0
symbol lookup cache used 0/65536 hits=0 misses=0
finished...

D:\projetos\projeto1.X\doc>

```



# Documentação do firmware ...

- Para visualizar a documentação desde o MPLABX, clicar acima do nome do projeto com o botão direito do mouse e selecionar a opção **Doxygen HTML Output View**. Em seguida deve aparecer no navegador o arquivo principal da documentação.
- Se usamos a linha de comandos, devemos abrir o arquivo *index.html* da pasta doc.
- Se usamos o **Doxywizard**, podemos clicar no botão show HTML output.

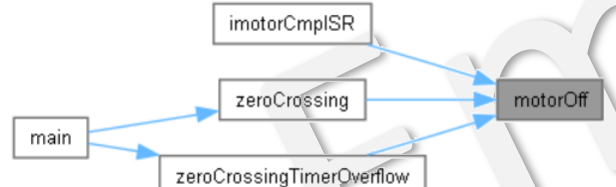
◆ motorOff()

```
void motorOff ( )
```

Desliga CCP e zera pino so SCR, desligando o motor.

Definição na linha 5 do arquivo **motor.c**.

Esse é o diagrama das funções que utilizam essa função:



```

graph LR
    main --> motorOff
    imotorCmplSR --> motorOff
    zeroCrossing --> motorOff
    zeroCrossingTimerOverflow --> motorOff
  
```

◆ HX711\_GetMeassure()

```
float HX711_GetMeassure ( HX711_sensor_e sensor )
```

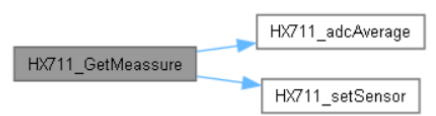
Filtra as medições de um sensor e converte para unidades de engenharia.

**Parâmetros**  
sensor

**Retorna**  
Valor em unidades de engenharia.

Definição na linha 32 do arquivo **HX711.c**.


Este é o diagrama das funções utilizadas por essa função:



```

graph LR
    HX711_GetMeassure --> HX711_adcAverage
    HX711_GetMeassure --> HX711_setSensor
  
```

Esse é o diagrama das funções que utilizam essa função:



```

graph LR
    main --> updateMeasurements
    updateMeasurements --> HX711_GetMeassure
  
```

# Documentação do firmware ...

Referência da Estrutura hx711\_sensor\_t

#include <HX711.h>

Campos de Dados

int24\_t

offset

Offset usado na conversão para unidades de engenharia.

float

gain

Ganho usado na conversão para unidades de engenharia.

int24\_t

adc [HX711\_FILTER]

Arranjo para armazenar os valores do filtro de janela móvel.

Descrição detalhada

Estrutura usada armazenar os dados dos sensores de carga e deformação.

Definição na linha 46 do arquivo HX711.h.

◆ HX711\_sensor\_e

enum HX711\_sensor\_e

Identificador dos circuitos HX711.

Enumeradores

hx711\_sensor1

Sensor e circuito 1.

hx711\_sensor2

Sensor e circuito 2.

Definição na linha 36 do arquivo HX711.h.

◆ HX711\_FILTER

#define HX711\_FILTER 10

Número de elementos do filtro de janela móvel usado para promediar as medições dos HX711.

Definição na linha 41 do arquivo HX711.h.