



**UFC**

**UNIVERSIDADE FEDERAL DO CEARÁ - CAMPUS SOBRAL**

**CURSO DE ENGENHARIA DA COMPUTAÇÃO**

**AUTOR: LUAN GOMES MAGALHÃES LIMA**

**MATERIAL AUXILIAR**

**TUTORIAL DE UTILIZAÇÃO DO MPLABX,**  
**PROTEUS E PICKIT3**

**SOBRAL - CE**

**2023**

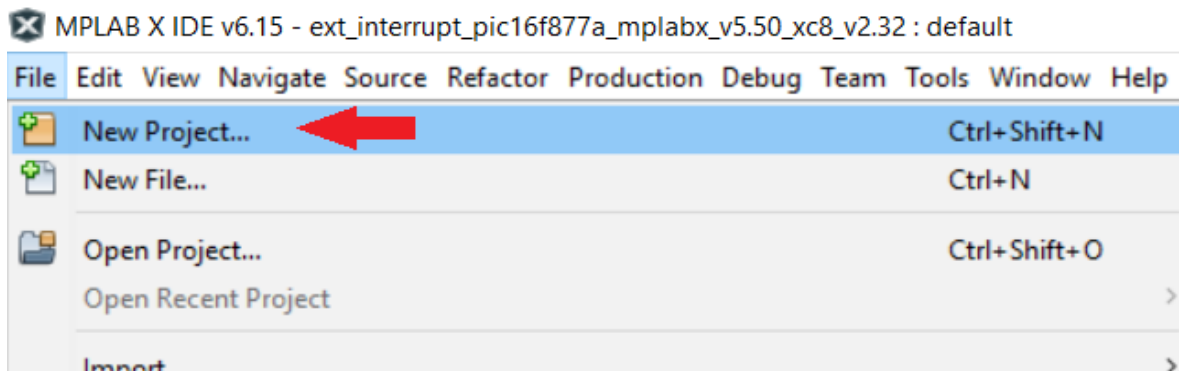
Este documento foi construído como parte de uma atividade de Estágio Supervisionado no Laboratório de Controle da Universidade Federal do Ceará - Campus Sobral.

O tutorial foi construído com base nos seguintes recursos:

- **Softwares:**
  - MPLABX - Versão: 6.15
  - Proteus - Versão: 8.13
- **Hardwares:**
  - PICkit3
  - Microprocessador PIC16F877A

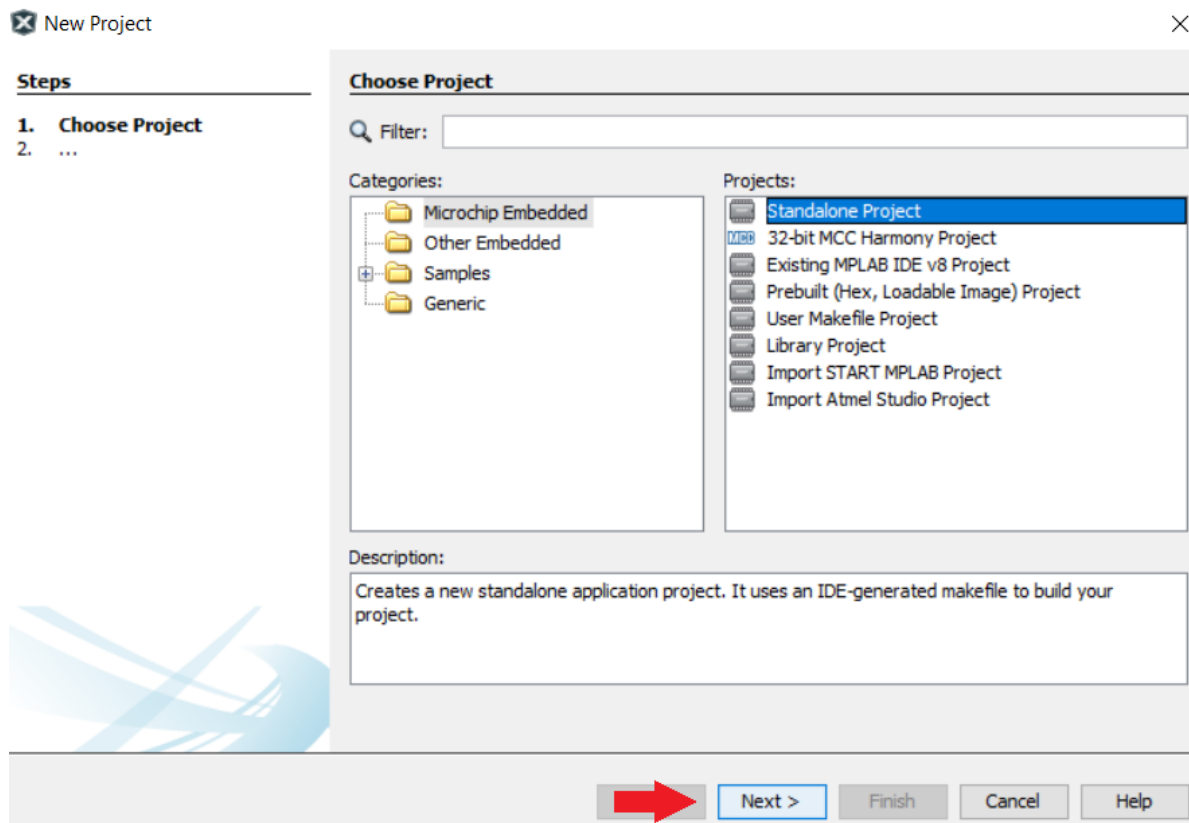
## 1. Abra um novo projeto no MPLABX:

### 1.1. File → New Project



## 2. Escolha a categoria e o tipo do projeto:

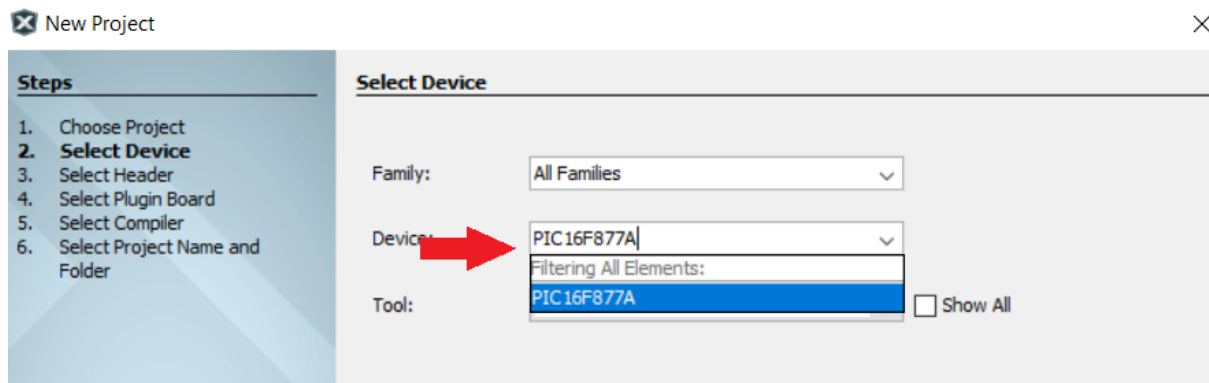
### 2.1. Microchip Embedded → Standalone Project → Next



## 3. Escolha o tipo do microprocessador que será utilizado, neste tutorial será utilizado o PIC16F877A, e a ferramenta de simulação:

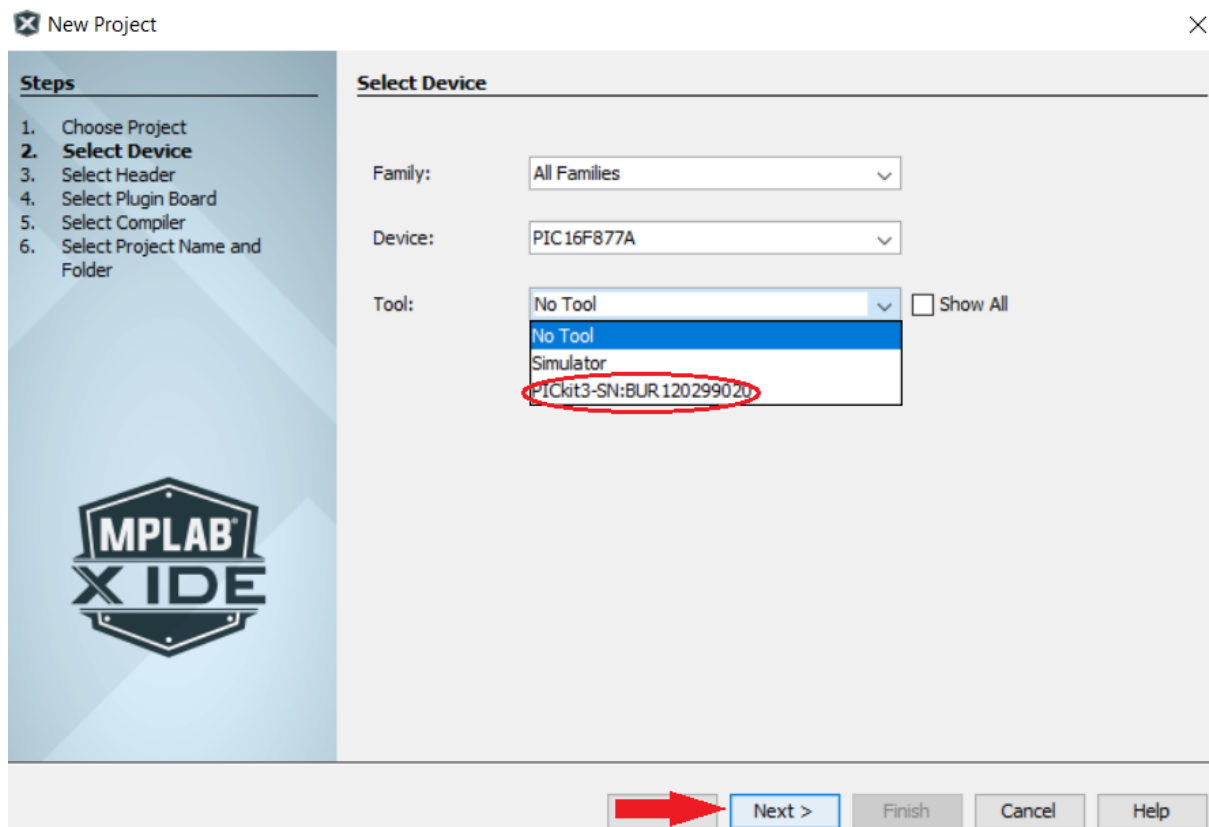
### 3.1. Device → PIC16F877A\*

\*O tipo de micro depende das especificações do projeto a ser construído.



### 3.2. Tool → PICKit3 → Next

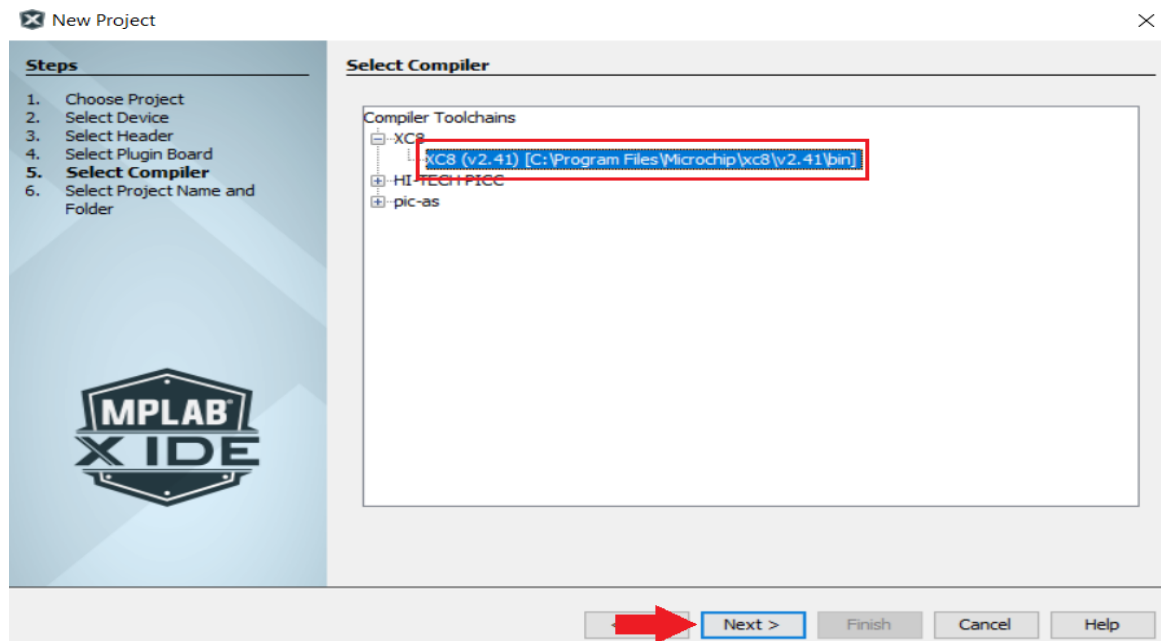
Caso o PICKit esteja plugado na entrada USB do computador, ele aparecerá para ser selecionado, caso contrário selecione “Simulator”.



## 4. Selecione o compilador utilizado, o exemplo deste tutorial será construído utilizando a linguagem C:

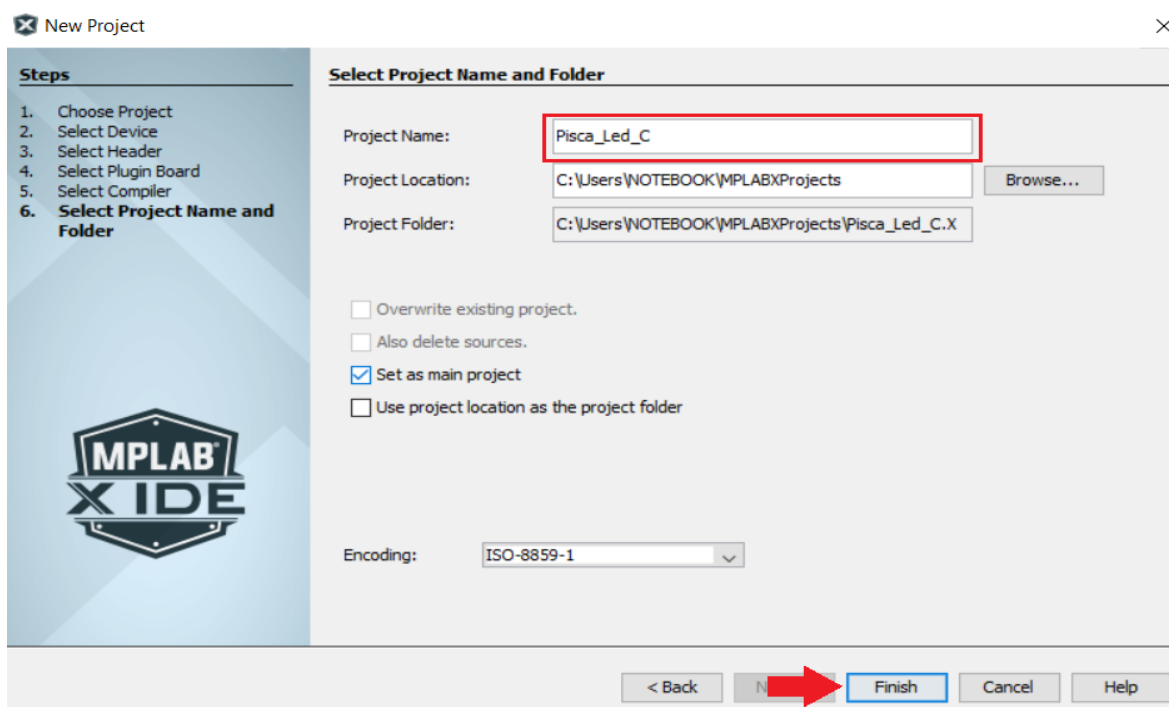
### 4.1. Compiler Toolchains → XC8\* → Next

\*Se o código do projeto será feito utilizando a linguagem assembly, selecione “pic-as”, caso seja feito utilizando a linguagem C, selecione “XC8”, nesse caso será necessário baixar o compilador XC8, este compilador está disponível no site da microchip (<https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers>)



## 5. Nomeie para o seu projeto e após isso está criado o seu projeto

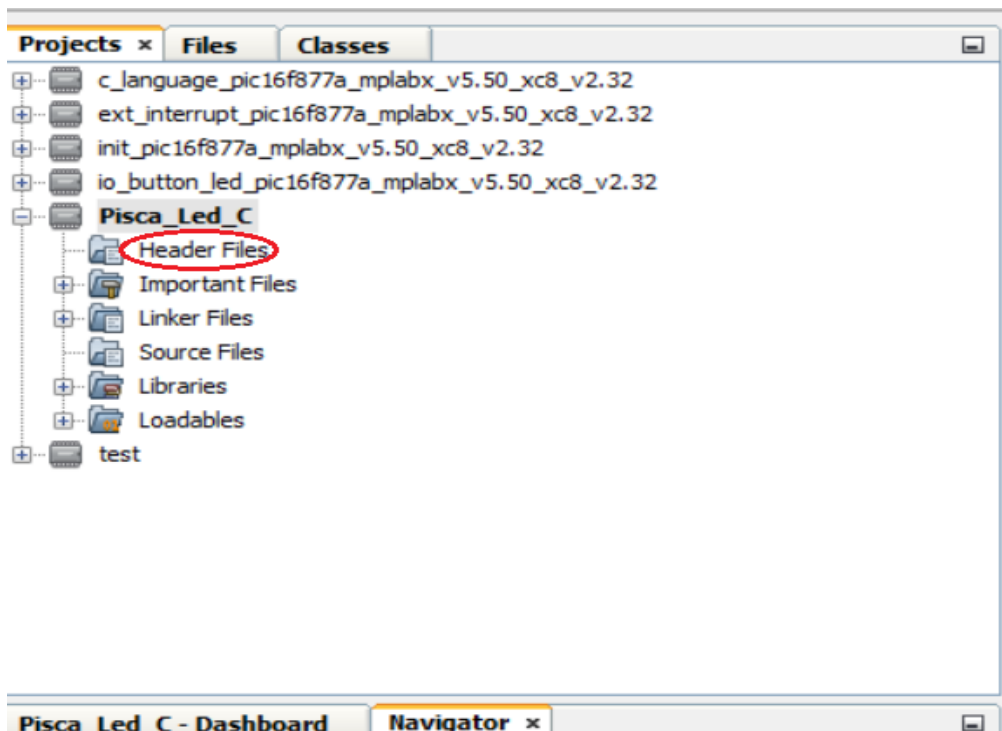
5.1. Project Name → “Nome\_do\_Projeto” → Finish



## 6. Crie os arquivos do programa (main.c e main.h)

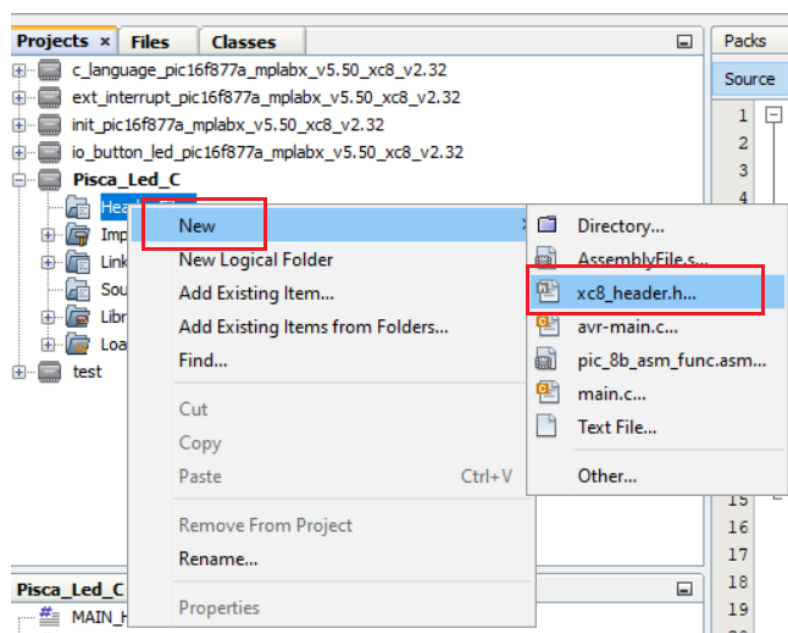
6.1. Para criar o arquivo main.h, siga as instruções abaixo:

6.1.1. Clique com o botão direito do mouse sobre “Header Files”

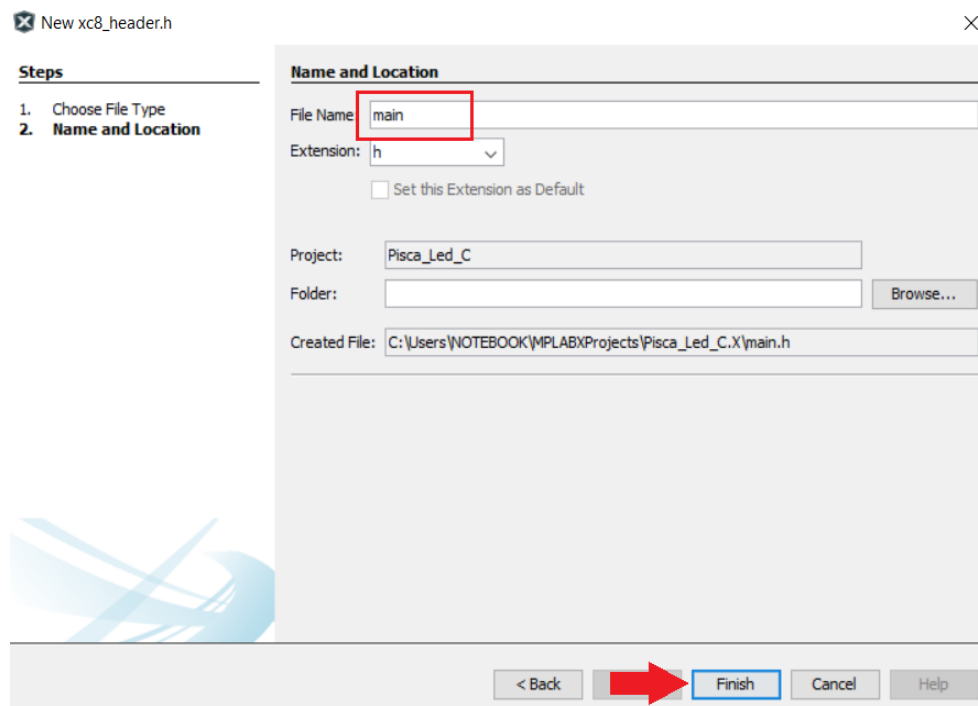


6.1.2. Header Files → New → xc8\_header.h

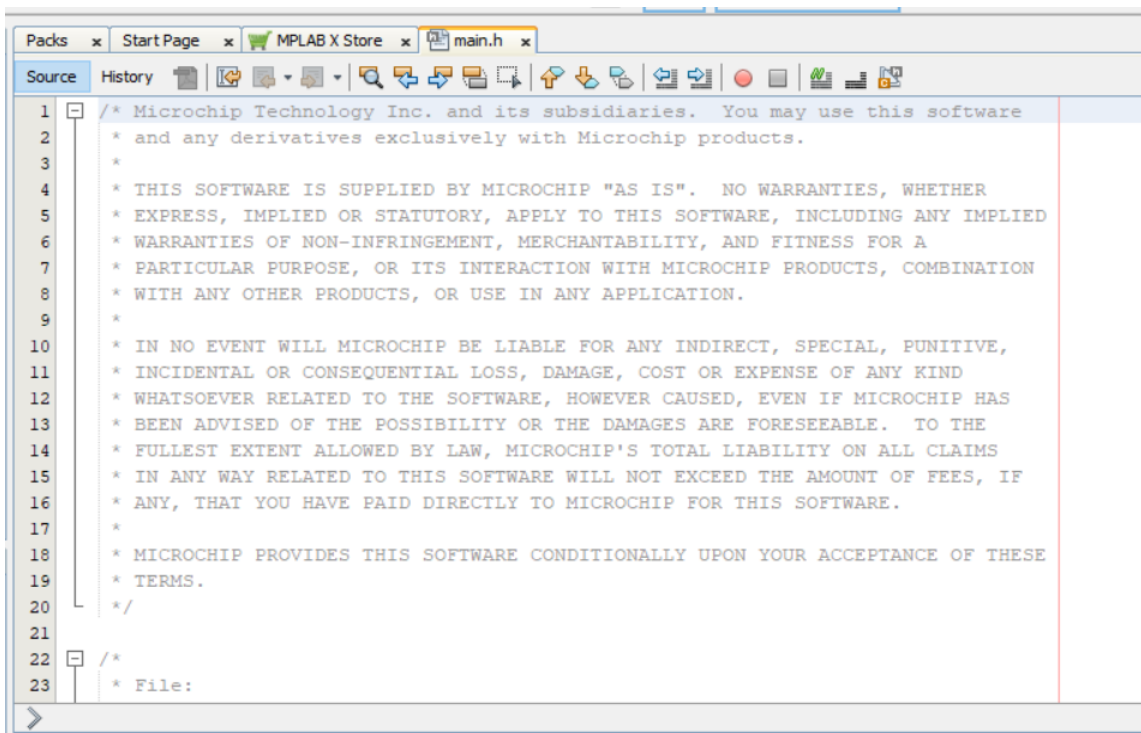
Como se trata de um código escrito em C utiliza-se a opção “xc8\_header.h”, mas dependendo do projeto, a escolha do arquivo header pode ser diferente.



### 6.1.3. Em seguida nomeie o arquivo e aperte em “Finish”

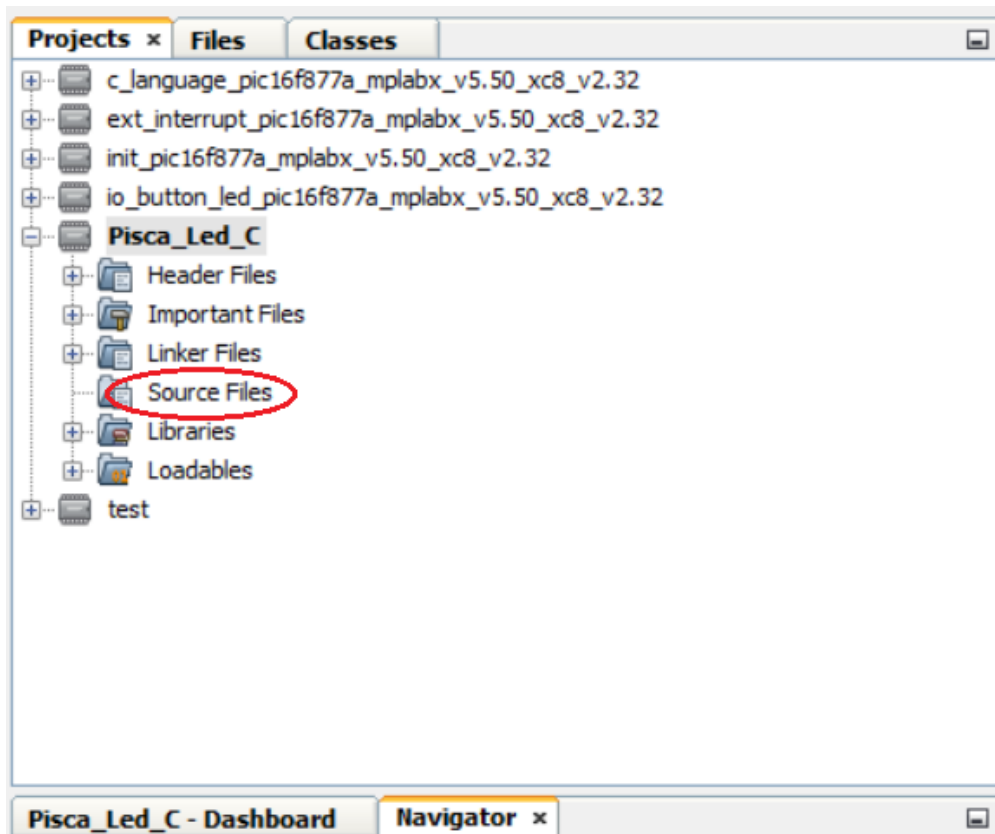


### 6.1.4. O modelo padrão será criado, coloque todos os códigos associados ao header nesse arquivo

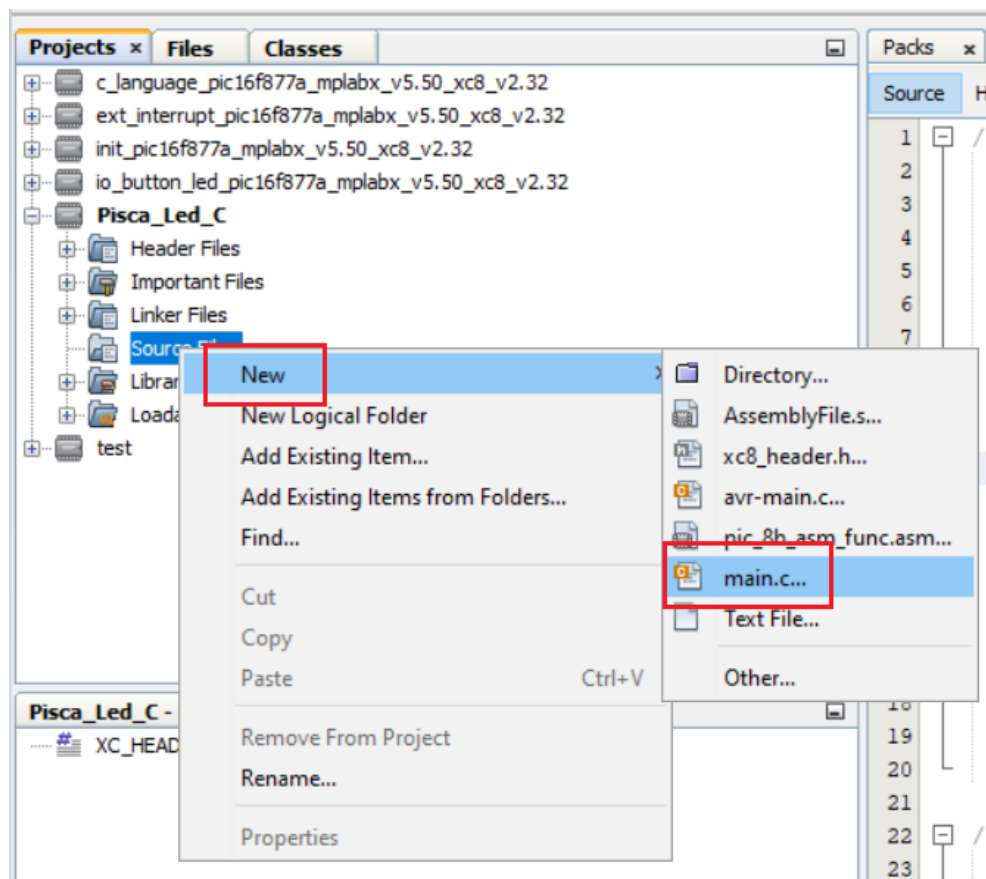


## 6.2. Para criar o arquivo main.c, siga as instruções abaixo:

### 6.2.1. Clique com o botão direito do mouse sobre “Source Files”

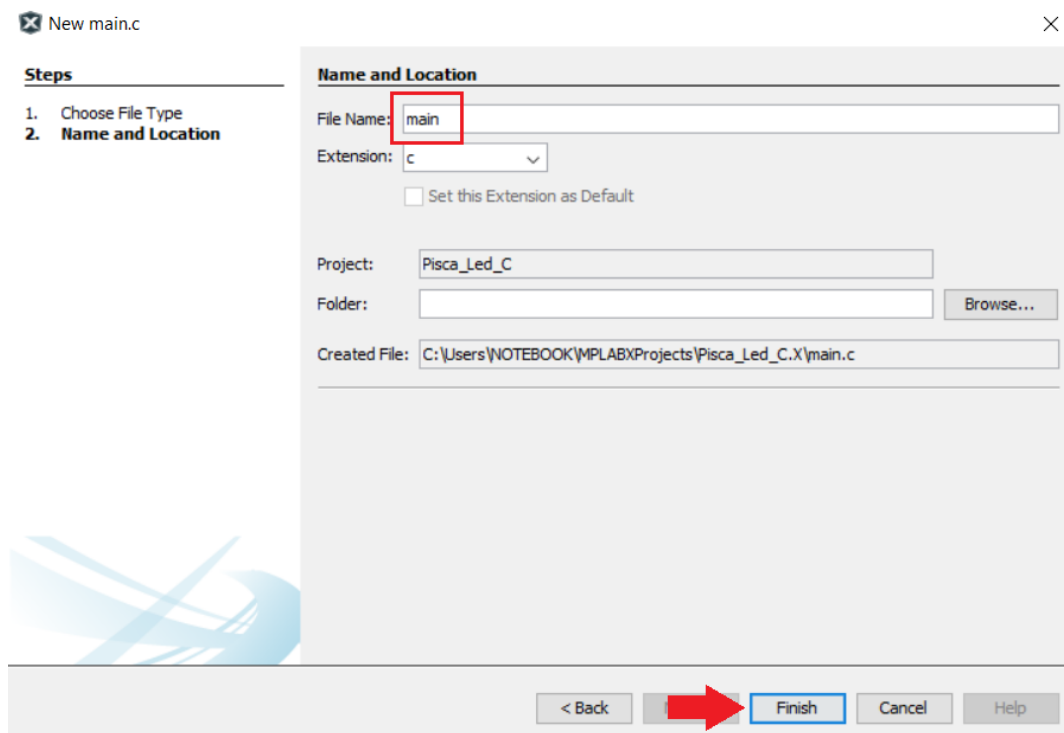


### 6.2.2. Source Files → New → main.c

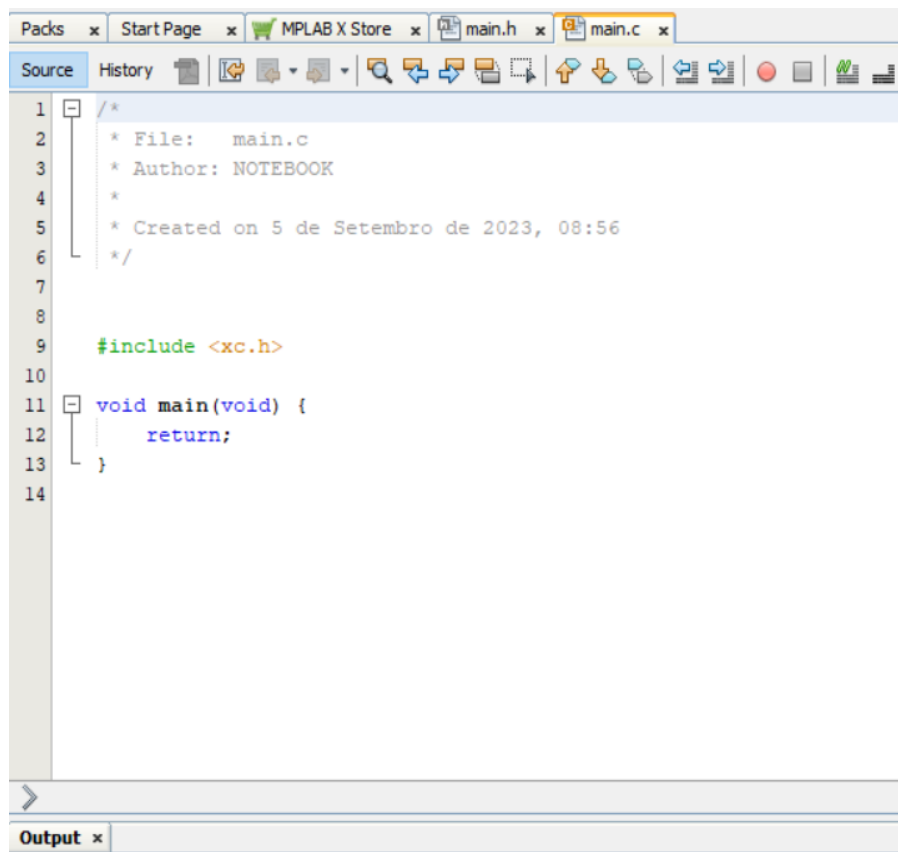




### 6.2.3. Em seguida nomeie o arquivo e aperte em “Finish”

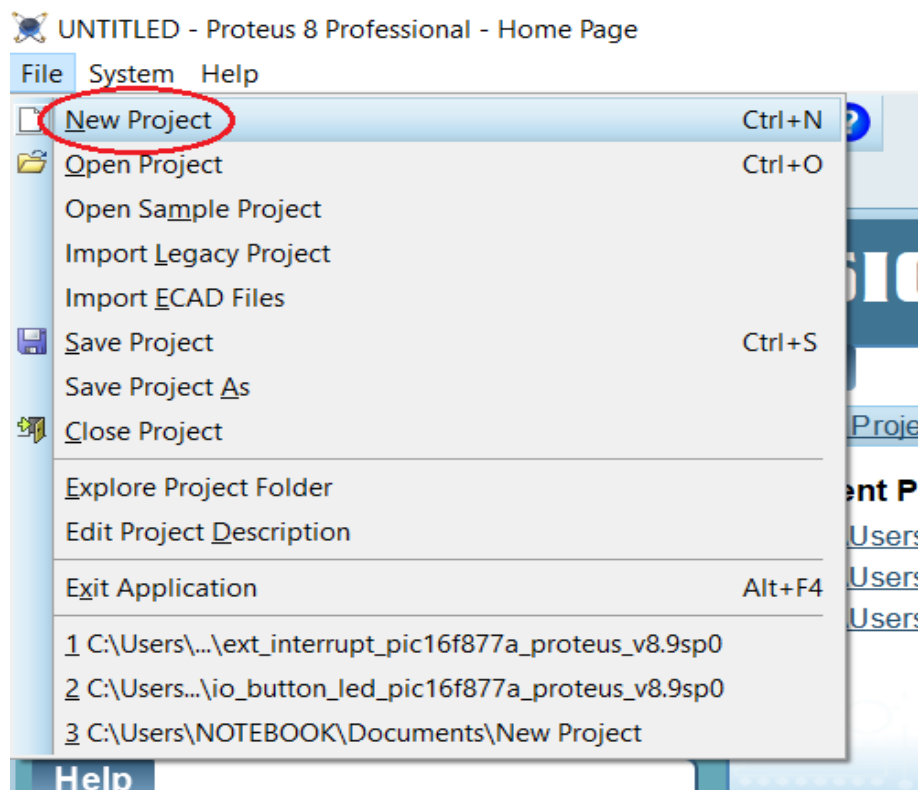


### 6.2.4. O modelo padrão será criado, coloque todos os códigos associados ao main nesse arquivo

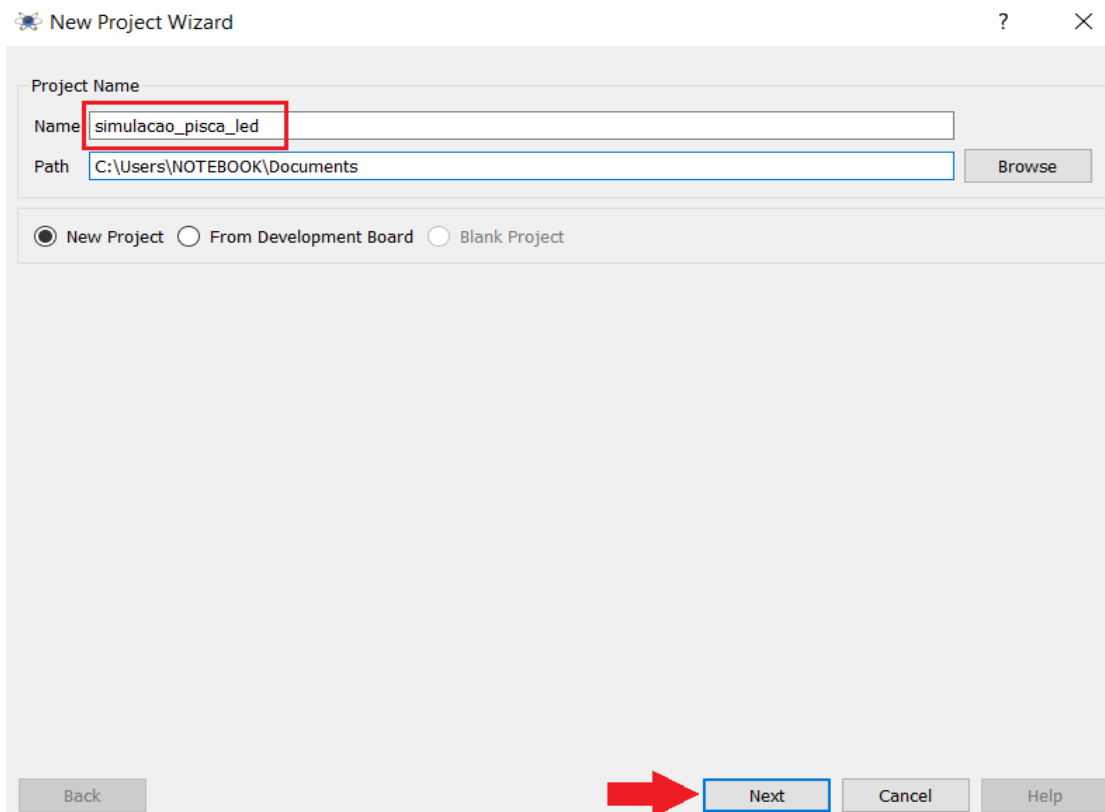


## 7. No Proteus crie o modelo de simulação para testar o código feito no MPLABX

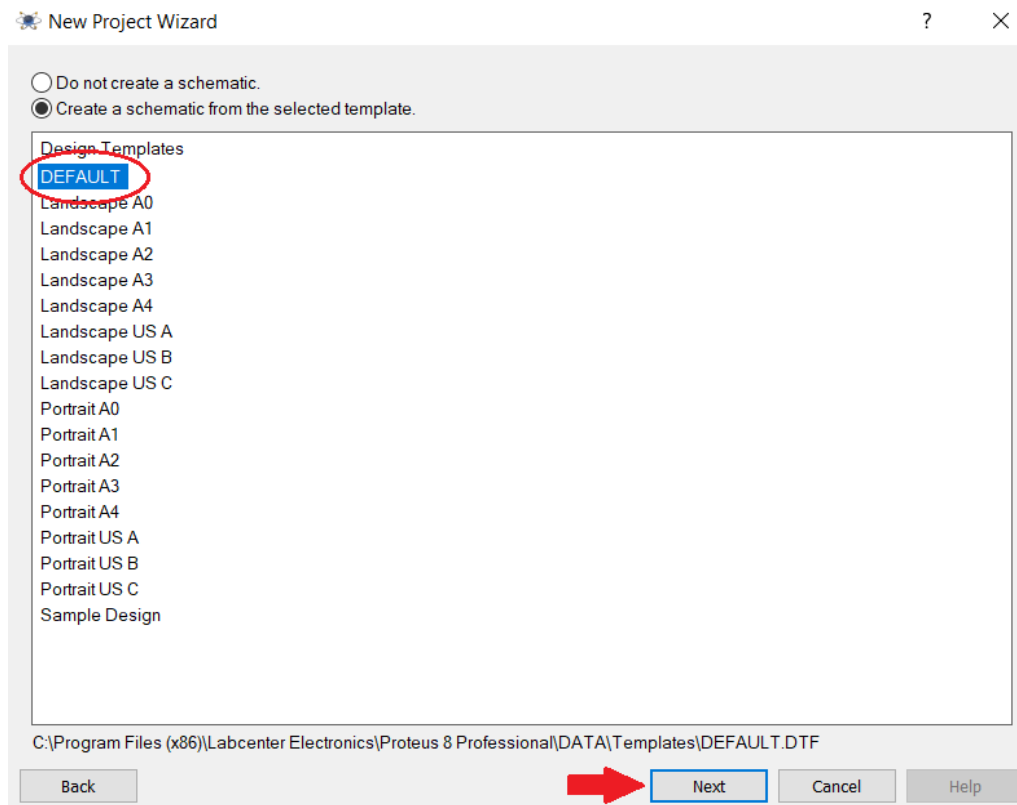
### 7.1. File → New Project



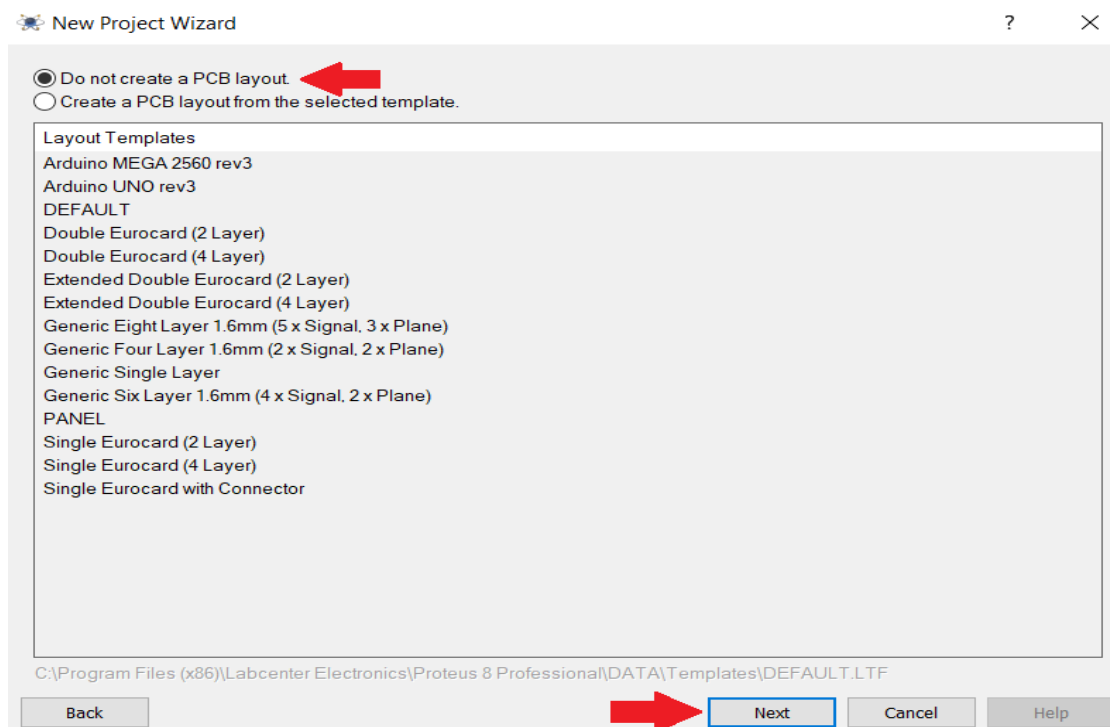
### 7.2. Nomeie o arquivo e pressione em “Next”



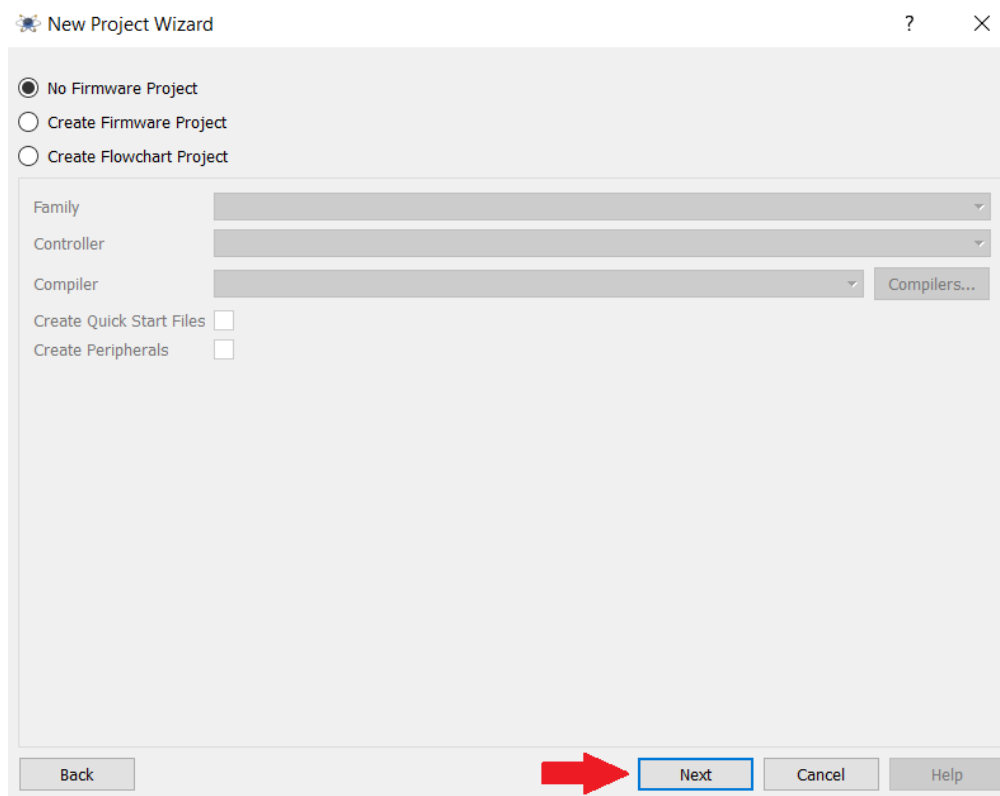
7.3. Selecione o modelo de template, em geral utiliza-se o template padrão, “DEFAULT”. Em seguida, pressione “Next”



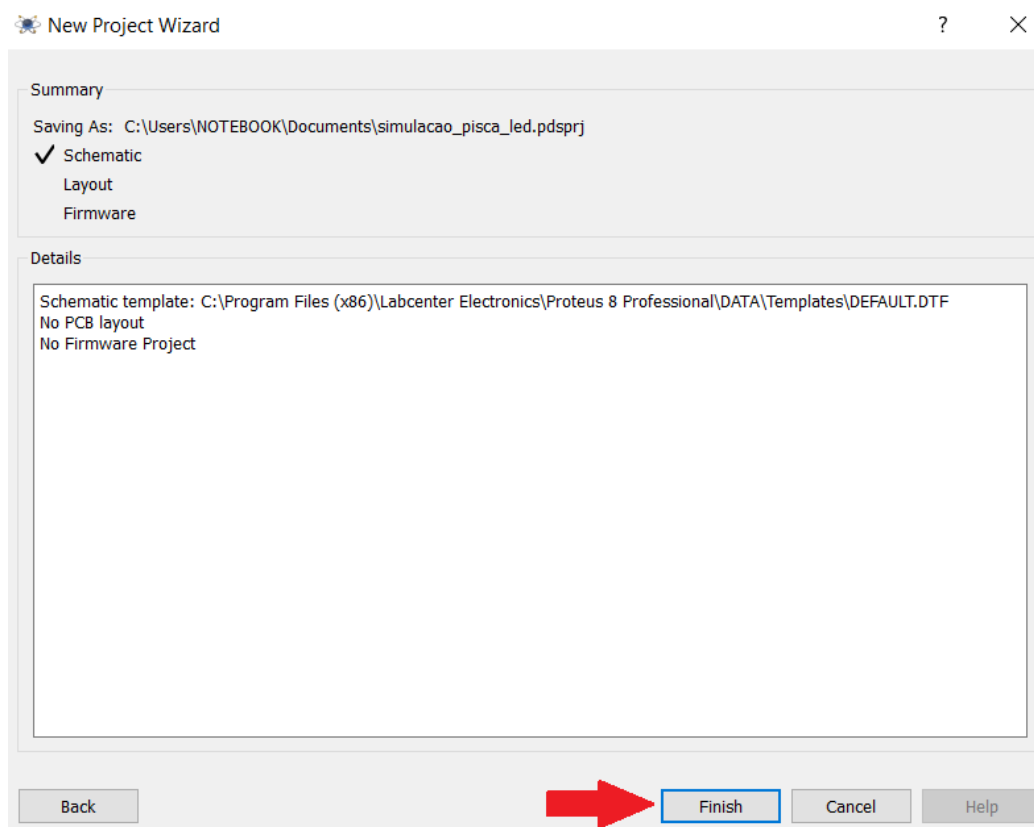
7.4. Se for construir algum projeto de PCB, escolha o template que melhor se adequa ao seu projeto, caso contrário selecione “Do not create a PCB layout” e pressione “Next”.



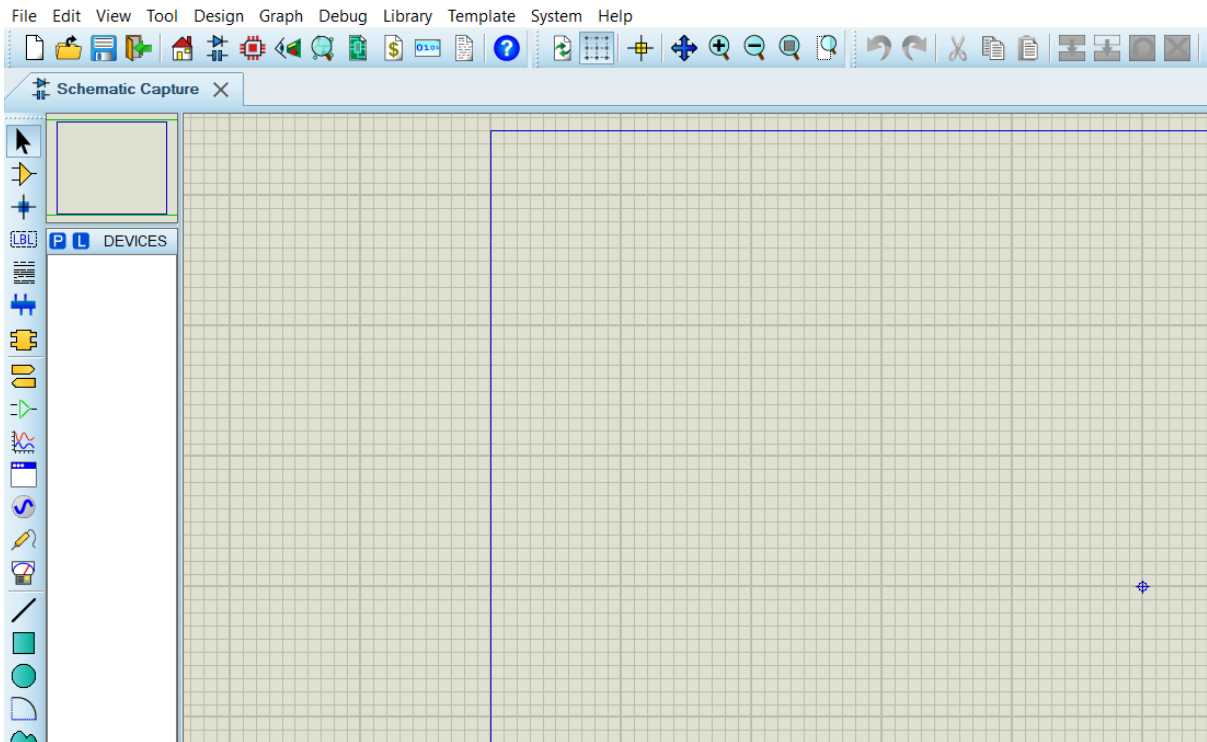
## 7.5. Pressione “Next” novamente



## 7.6. Pressione “Finish”

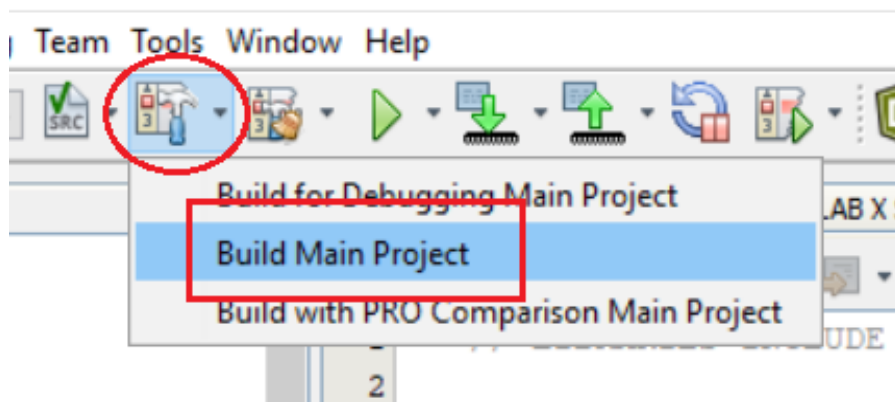


7.7. Seu modelo estará criado, agora é só construir o esquemático do projeto



Após construir os códigos no MPLABX e no PROTEUS, verifique a compilação e simulação.

8. Selecione o ícone mostrado e pressione “Build Main Project”



Após isso, o arquivo hexadecimal será criado. Conforme, mostrado abaixo.

```
Output - Pisca_Led_C (Build, Load) x
Configuration bits used 1h ( 1) of 1h word (100.0%)
ID Location space used 0h ( 0) of 4h bytes ( 0.0%)

make[2]: Leaving directory 'C:/Users/NOTEBOOK/MPLABXProjects/Pisca_Led_C.X'
make[1]: Leaving directory 'C:/Users/NOTEBOOK/MPLABXProjects/Pisca_Led_C.X'

BUILD SUCCESSFUL (total time: 6s)
Loading code from C:/Users/NOTEBOOK/MPLABXProjects/Pisca_Led_C.X/dist/default/production/Pisca_Led_C.X.production.hex...
Program loaded with pack, PIC16Fxxx_DFP, 1.4.149, Microchip
Loading completed
```

## 9. Descarregue o arquivo .hex no modelo do PROTEUS

9.1. Dê um duplo clique sobre o modelo. A seguinte janela será exibida

**Edit Component**

Part Reference: U1 Hidden: ☐

Part Value: PIC16F877A Hidden: ☐

Element:  New

PCB Package: DIL40  Hide All

Program File: ..\\firmware\\io\_button\_led\_pic16f87  Hide All

Processor Clock Frequency: 20MHz Hide All

Program Configuration Word: 0x3FFB Hide All

Advanced Properties:

Randomize Program Memory?  No  Hide All

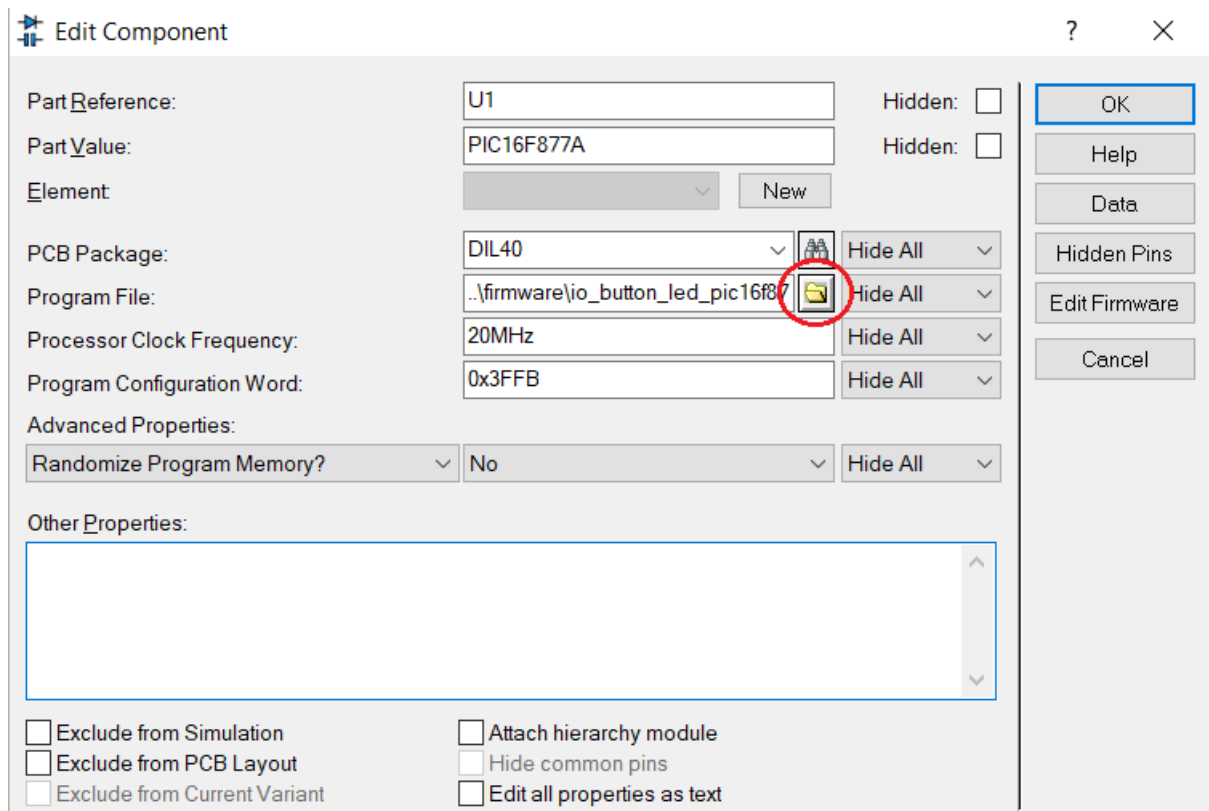
Other Properties:

☐ Exclude from Simulation ☐ Attach hierarchy module

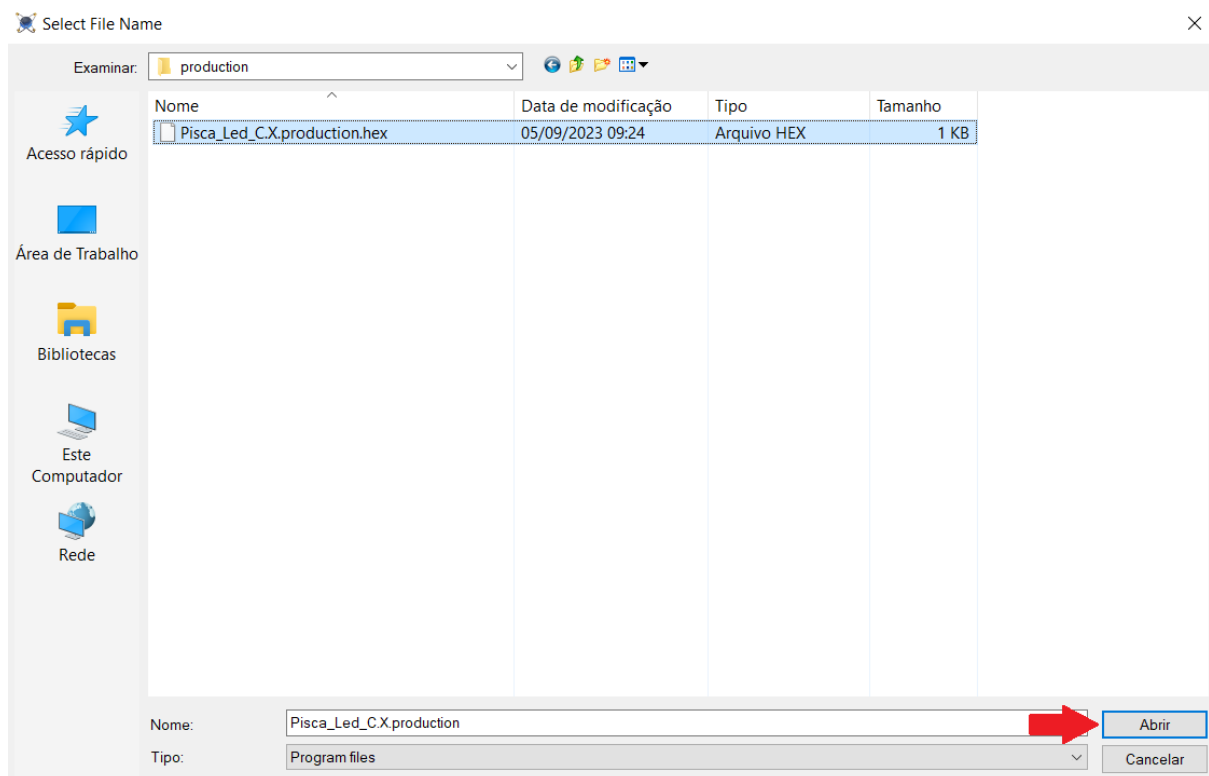
☐ Exclude from PCB Layout ☐ Hide common pins

☐ Exclude from Current Variant ☐ Edit all properties as text

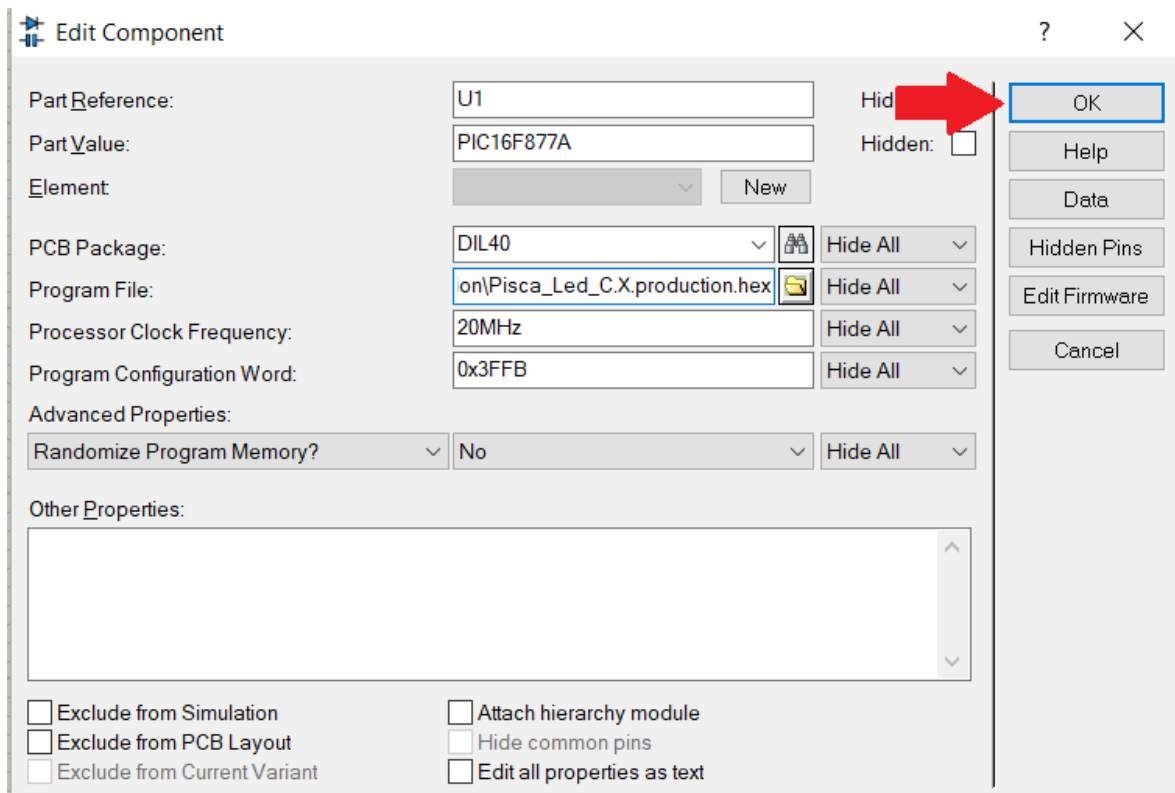
9.2. Escolha o caminho que está arquivo .hex



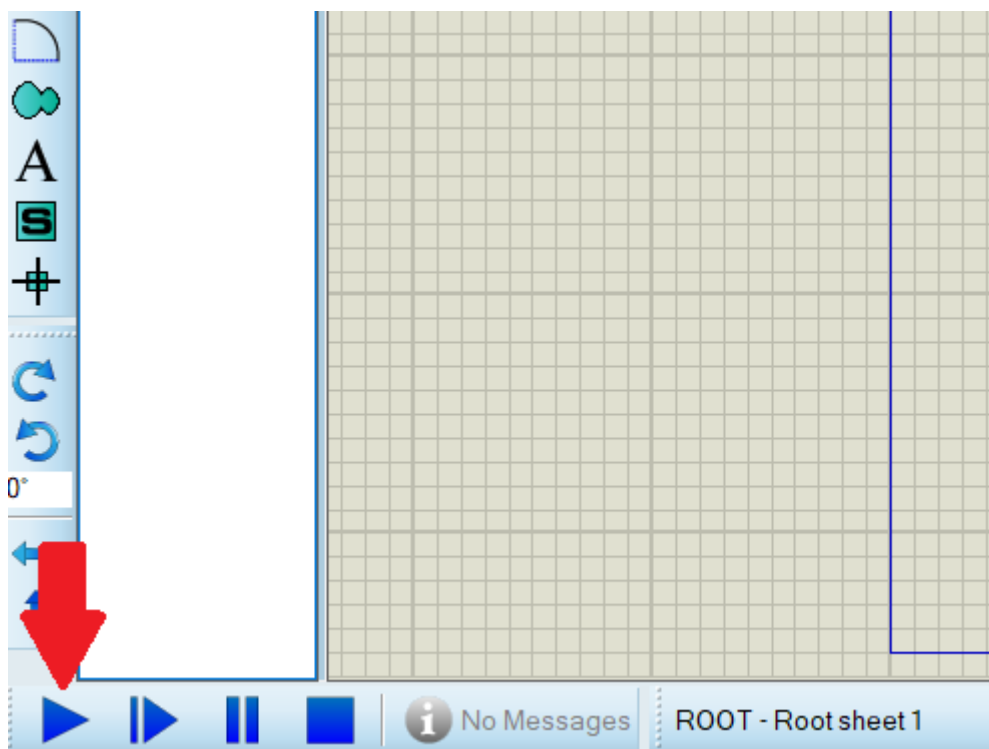
### 9.3. Seleção o arquivo e pressione “Abrir”



### 9.4. Após isso pressione “Ok”



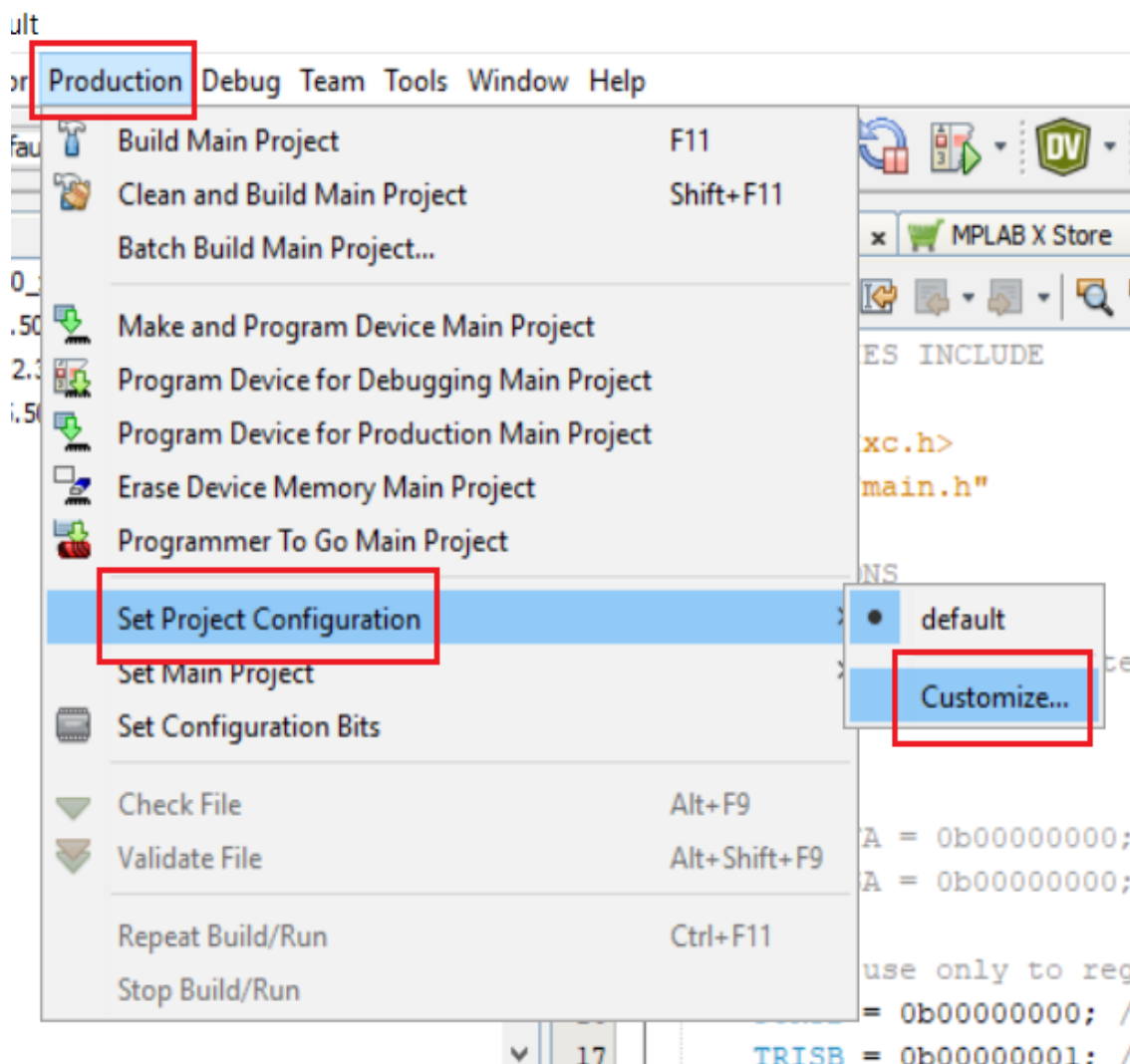
10. No canto inferior esquerdo selecione o ícone mostrado para verificar a simulação



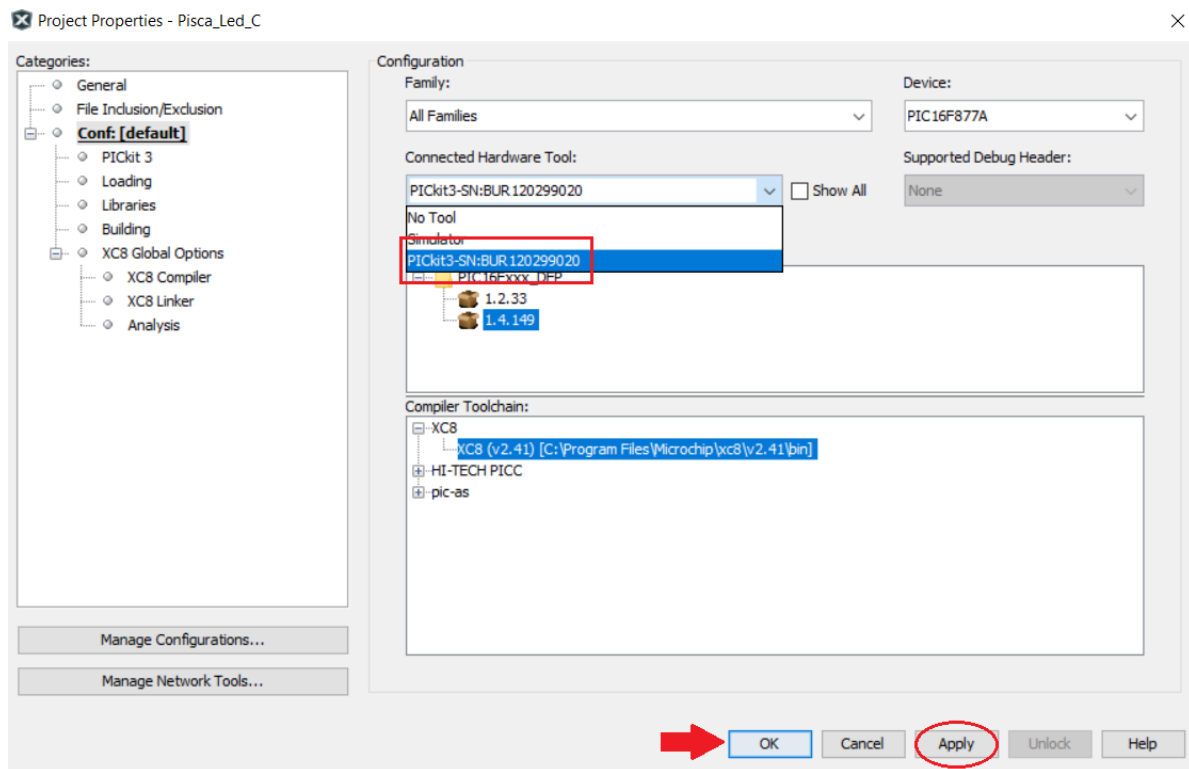
11. Para descarregar o arquivo no PIC16F877A, siga as etapas abaixo:



- 11.1. Conecte o PicKit3 na entrada USB do seu computador, caso ainda não esteja conectado
- 11.2. Se durante a criação do projeto no MPLABX você já tivesse selecionado o gravador, não será necessário executar as etapas 11.2 e 11.3. Caso contrário ou então se o gravador não tiver selecionado siga as instruções: Production → Set Project Configuration → Customize

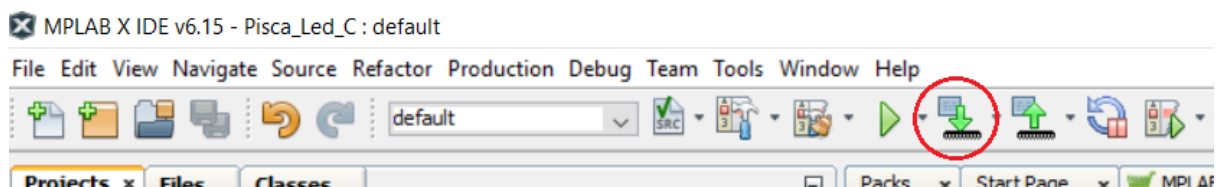


- 11.3. Connected Hardware Tool → PicKit3 (Modelo do gravador) → Apply → OK

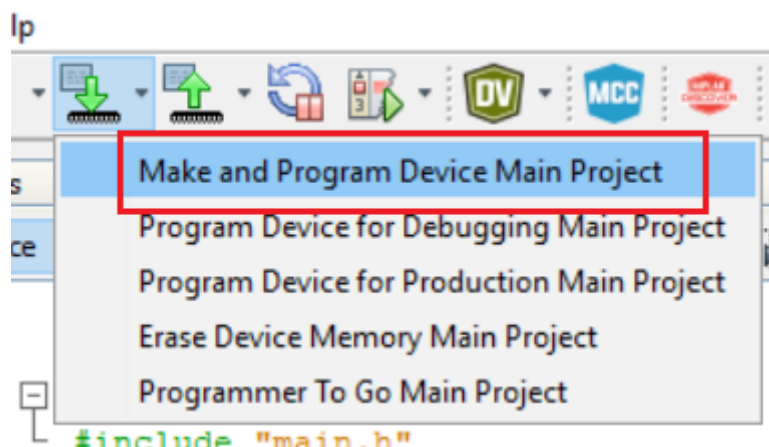


11.4. Conecte o Pickit3 na entrada de dados do micros, vale destacar que as conexões devem está corretamente conectadas

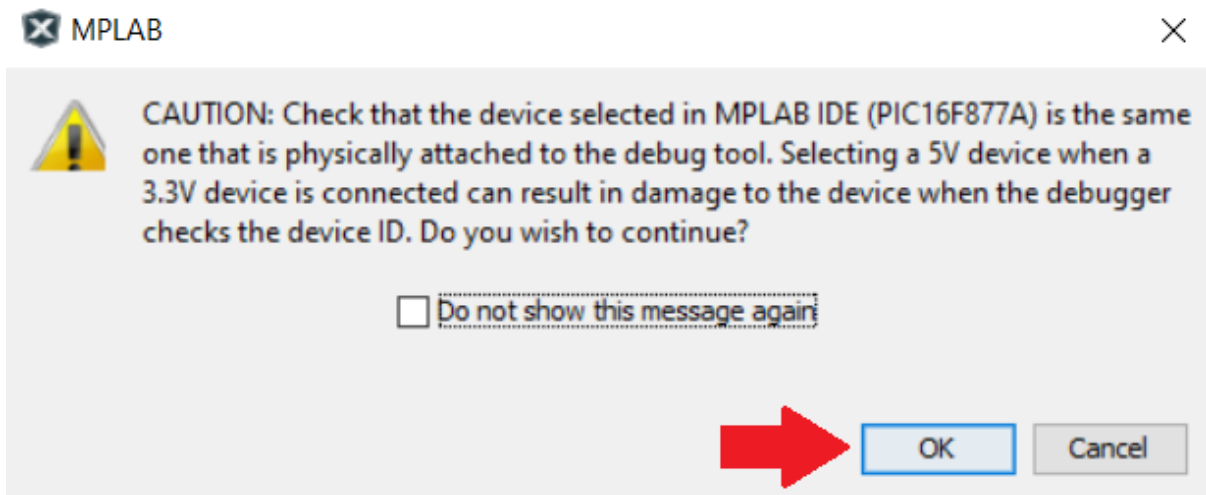
11.5. No canto superior selecione o ícone mostrado abaixo:



Outra alternativa seria selecionar a seta ao lado ícone mostrado acima e selecionar a opção “Make and Program Device Main Project”

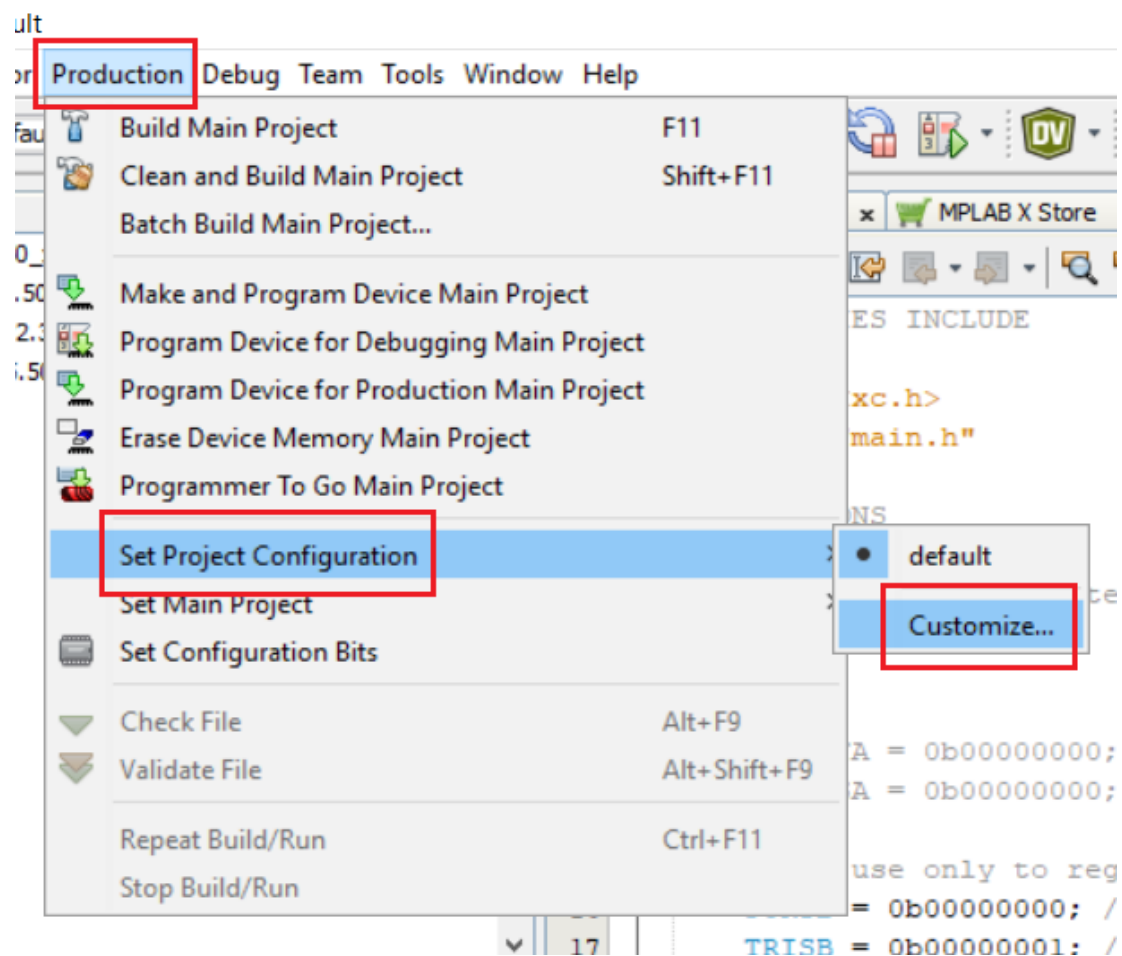


- 11.6. A seguinte janela será aberta, indicando o cuidado com a alimentação do micro. Pressione OK

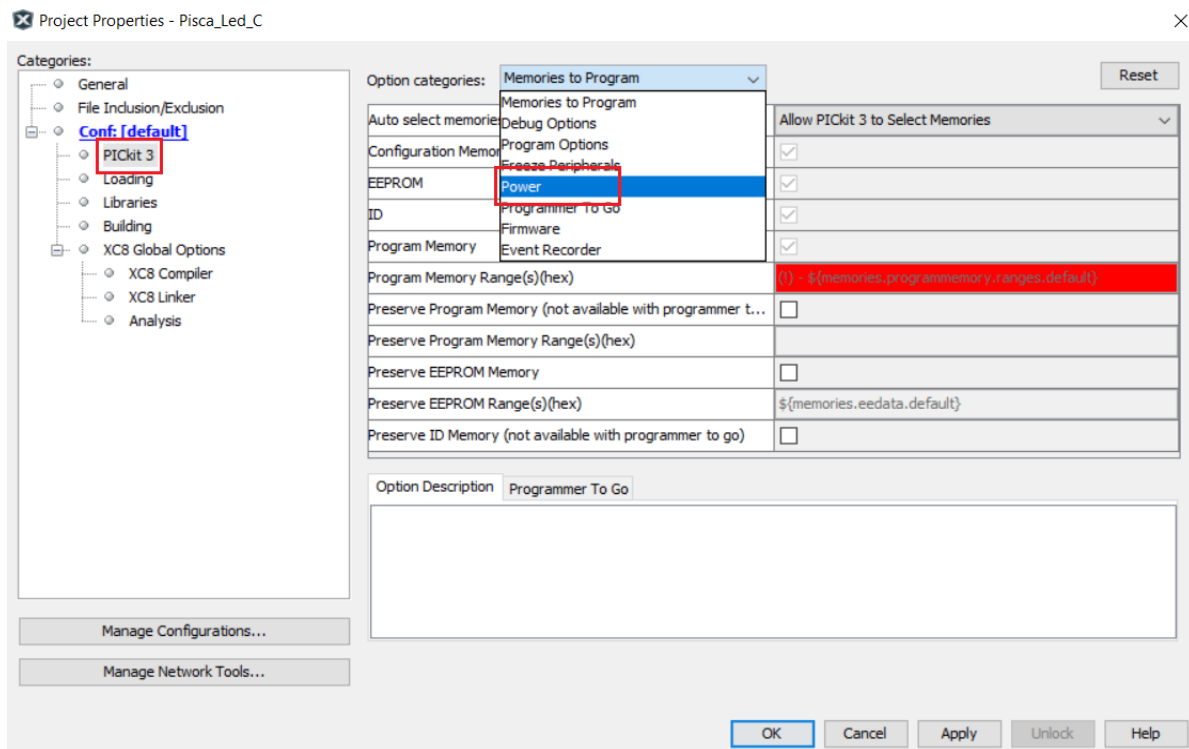


12. Caso deseje fornecer a tensão pelo próprio PicKit3, siga as instruções abaixo:

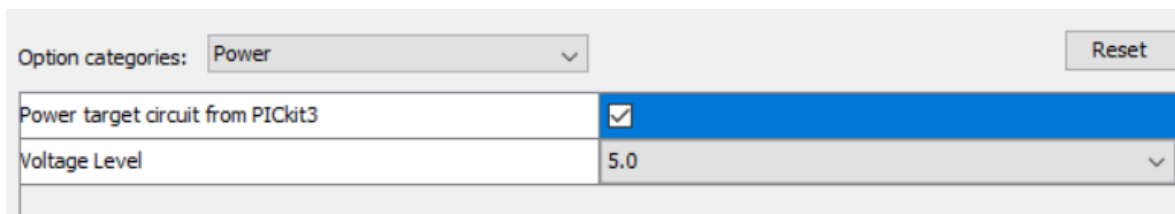
12.1. Production → Set Project Configuration → Customize



## 12.2. PICKit 3 → Power

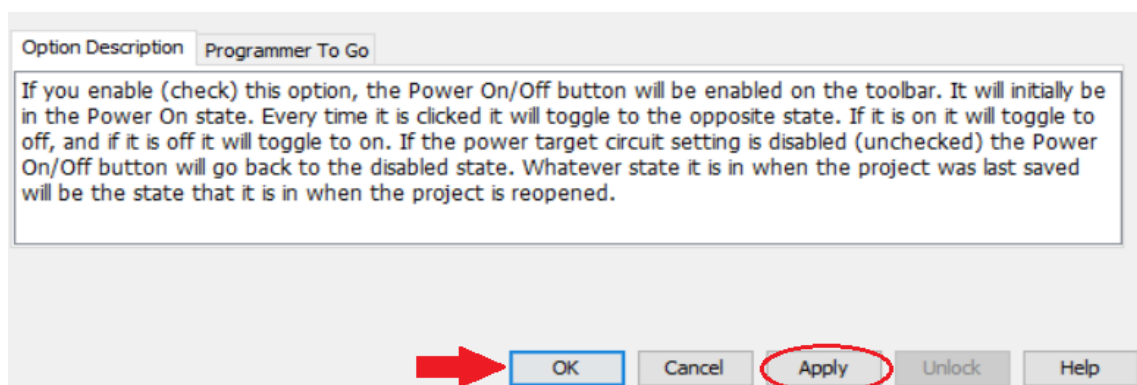


## 12.3. Selecione a opção “Power target circuit from PICKit3”

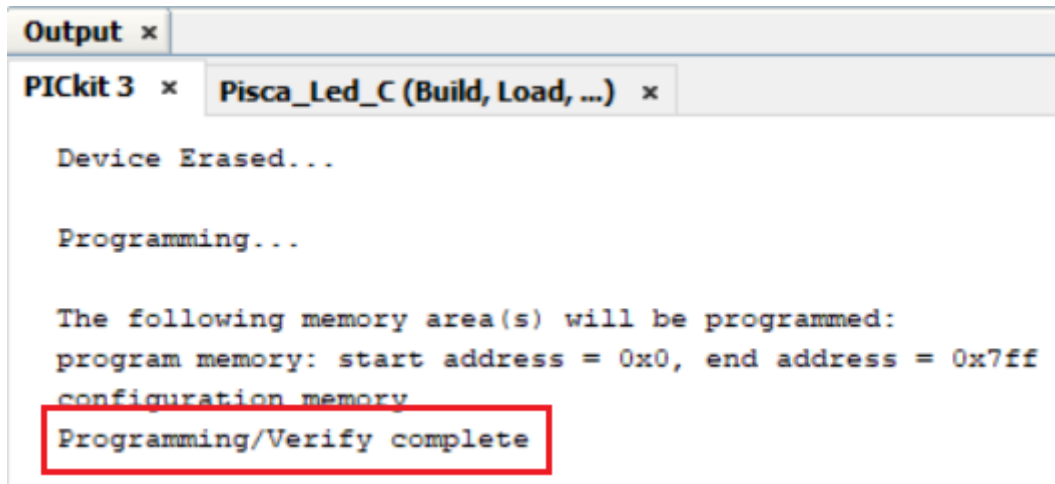


Isso faz com que o PickKit3 forneça a tensão de 5 V para o micro.

## 12.4. Em seguida, no canto inferior da janela. Pressione “Apply” e depois “OK”



13. Caso a gravação ocorra de forma correta, a seguinte mensagem aparecerá:



The screenshot shows the MPLABX Output window with the following text:

```
Output x
PICKit 3 x  Pisca_Led_C (Build, Load, ...) x

Device Erased...

Programming...

The following memory area(s) will be programmed:
program memory: start address = 0x0, end address = 0x7ff
configuration memory
Programming/Verify complete
```

The message "Programming/Verify complete" is highlighted with a red rectangular box.

#### Observações:

- No arquivo main.h, a configuração associada ao Low-Voltage Programming (LVP) permite a programação com tensão mais baixa, essa configuração é extremamente útil para gravar o micro com voltagens reduzidas. Para ativar essa configuração, faça:

#pragma config LVP = ON

Contudo é necessário desativar essa configuração caso queira utilizar o micro com uma fonte de tensão externa, sem a necessidade do PicKit3 está plugado no projeto. Para isso, faça:

#pragma config LVP = OFF

- Para gerar os bits de configuração de cada micro, é possível fazer automaticamente no próprio MPLABX. As informações relacionadas a isso podem ser vistas no seguinte link:  
<https://microchipdeveloper.com/mplabx:view-and-set-configuration-bits>