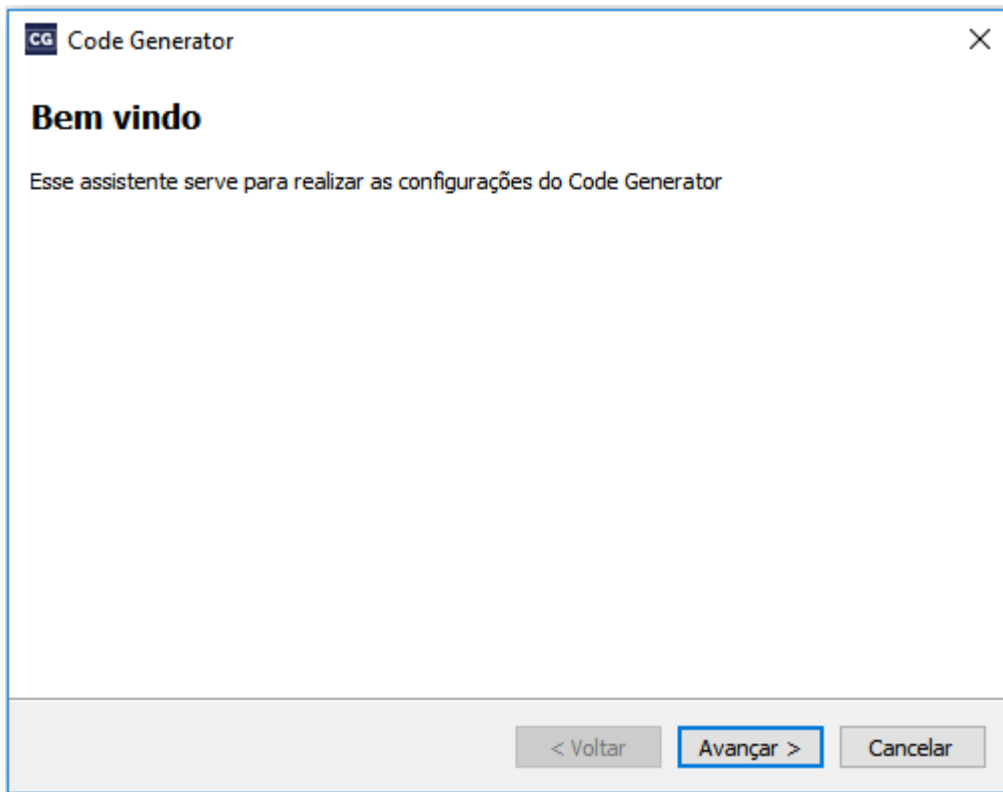


Tutorial do Code Generator

1 - Tela inicial

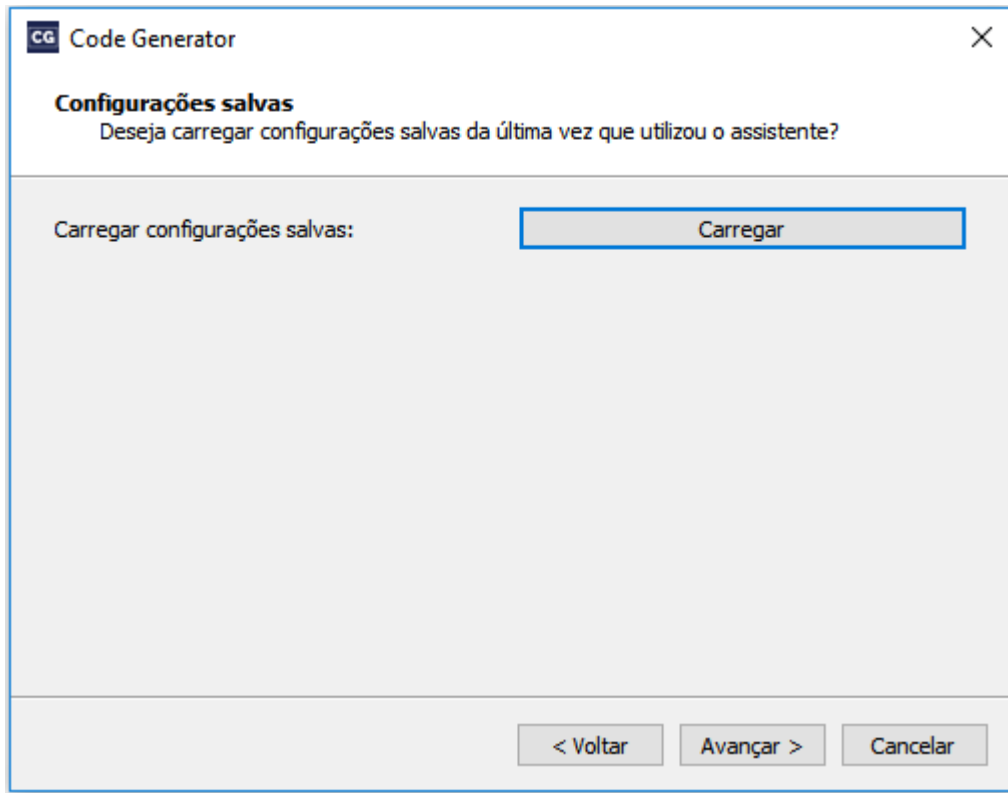
A tela inicial do assistente Code Generator é apresentada na imagem a seguir.



- Para prosseguir o usuário deve clicar no botão **Avançar**.

2 - Tela para carregar configurações salvas

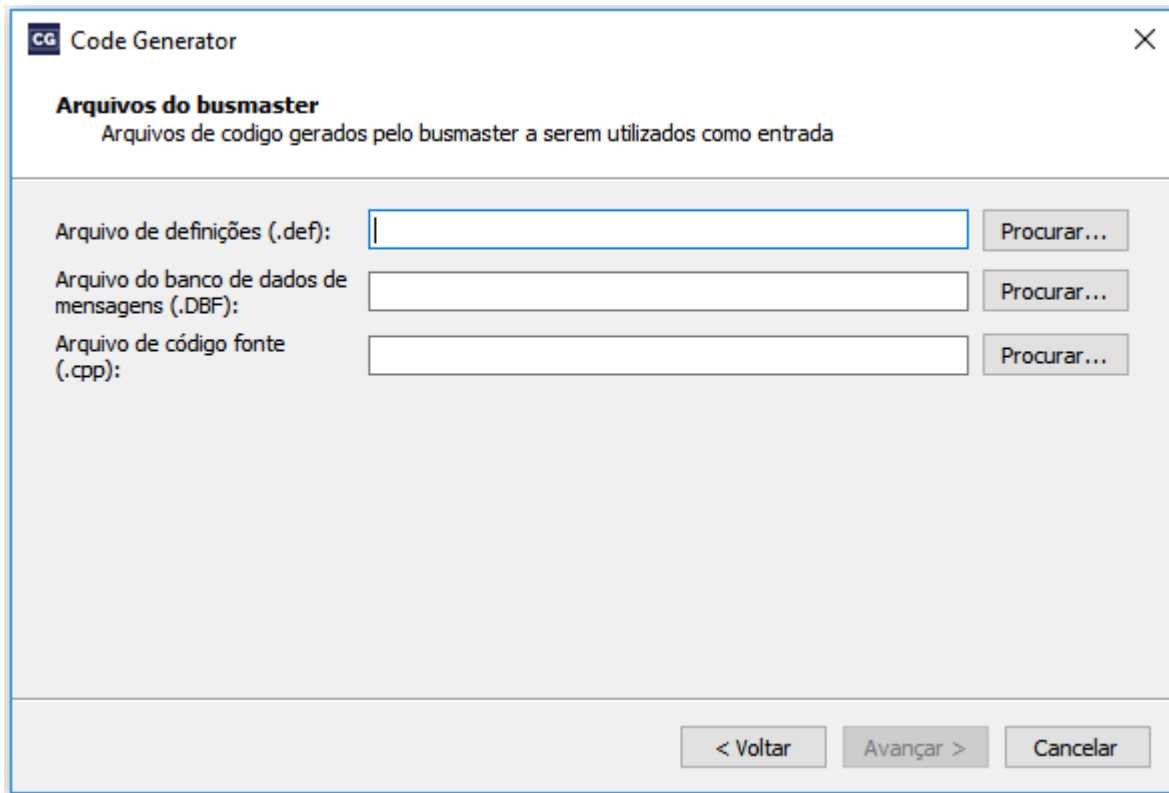
A tela seguinte permite que o usuário escolha se deseja carregar as informações salvas ou não.



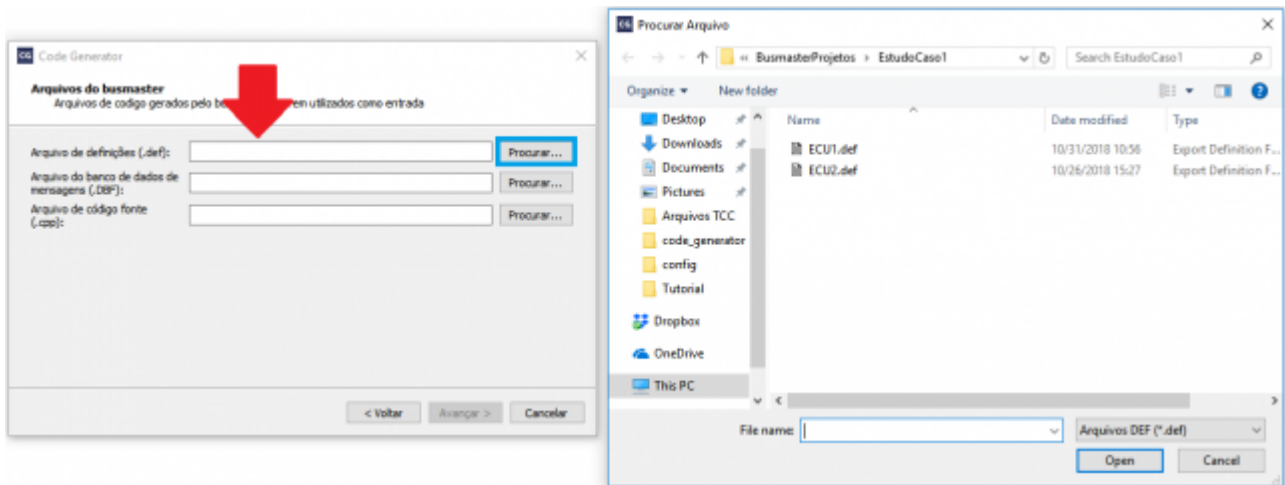
- Caso o usuário deseje carregar as informações salvas ele deve clicar no botão **Carregar**.
- Em seguida o usuário deve clicar no botão **Avançar**.

3 - Tela para selecionar arquivos de entrada

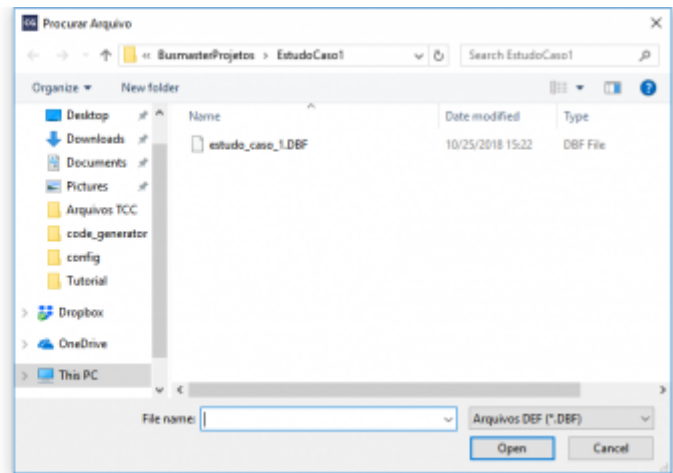
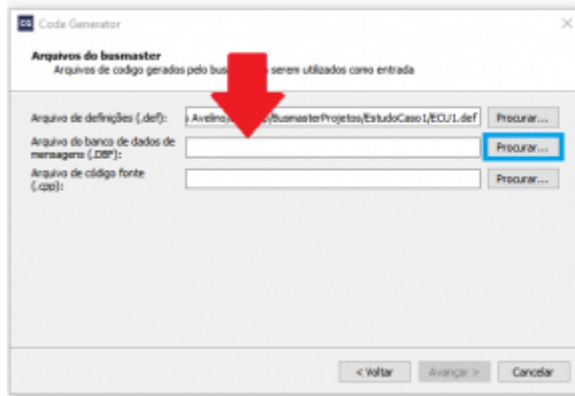
Na tela seguinte o usuário deve informar os arquivos de código fonte desenvolvidos através do software **Busmaster**. Para auxiliar a preencher o nome destes arquivos, pode se utilizar o explorador de arquivos através do botão **Procurar...** associado a cada campo de preenchimento.



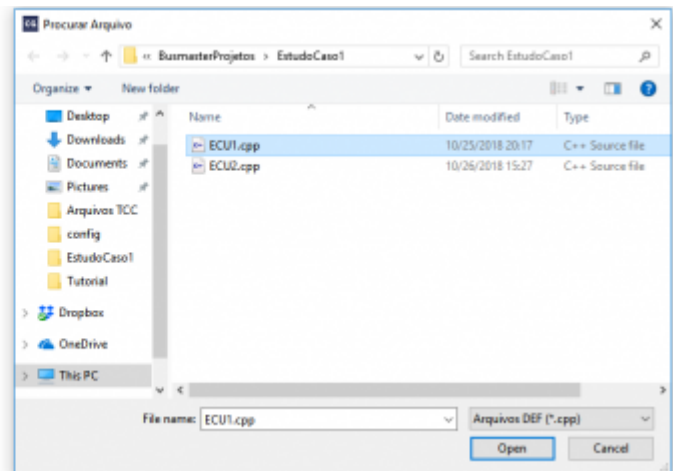
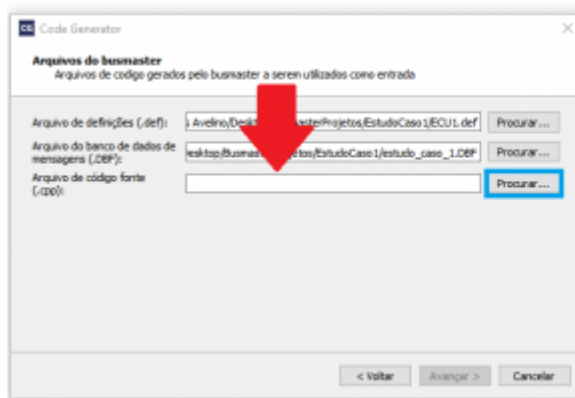
- Para seleccionar o arquivo de definições pode-se clicar no botão **Procurar...** a ele associado.



- Para seleccionar o arquivo do banco de dados de mensagens pode-se fazer o mesmo.



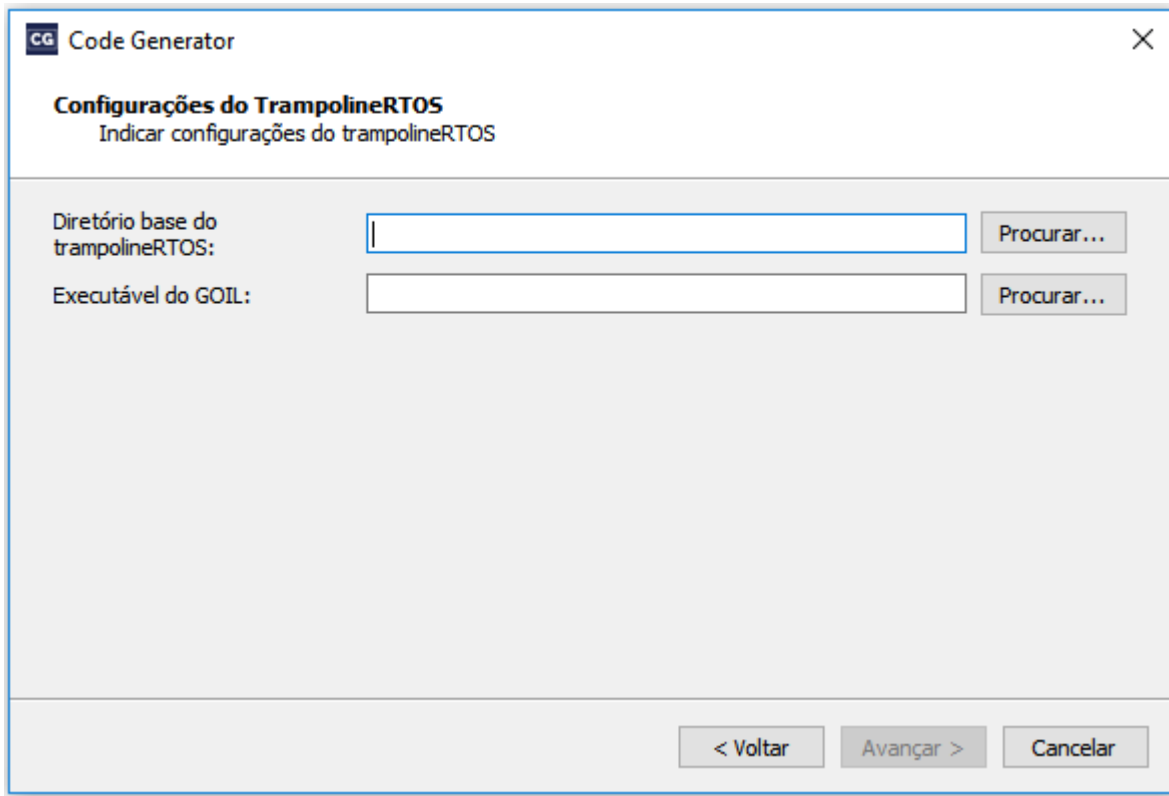
- Para seleccionar o arquivo de código fonte C++ também pode-se fazer o mesmo.



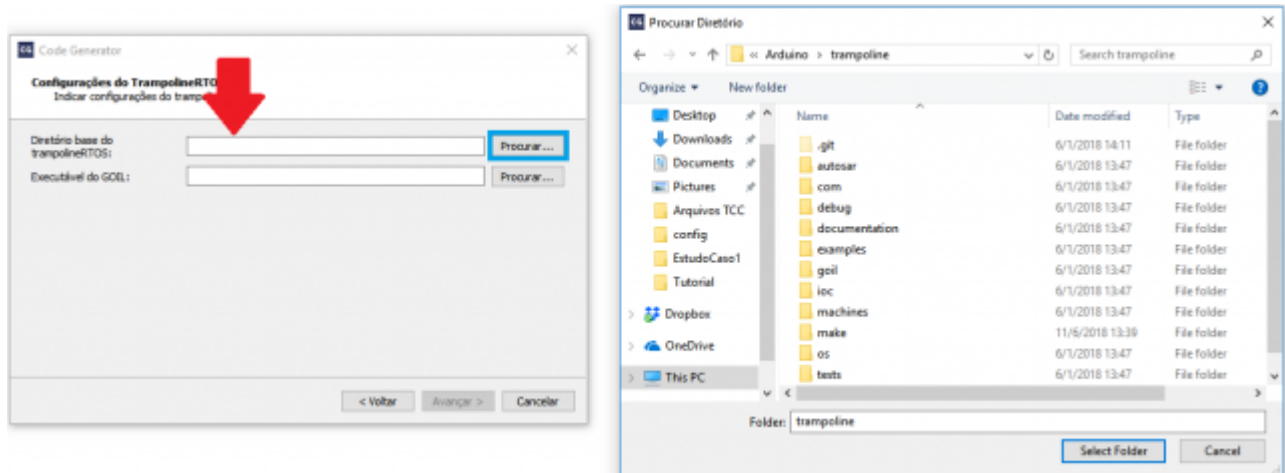
- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

4 - Tela para configurações do TrampolineRTOS

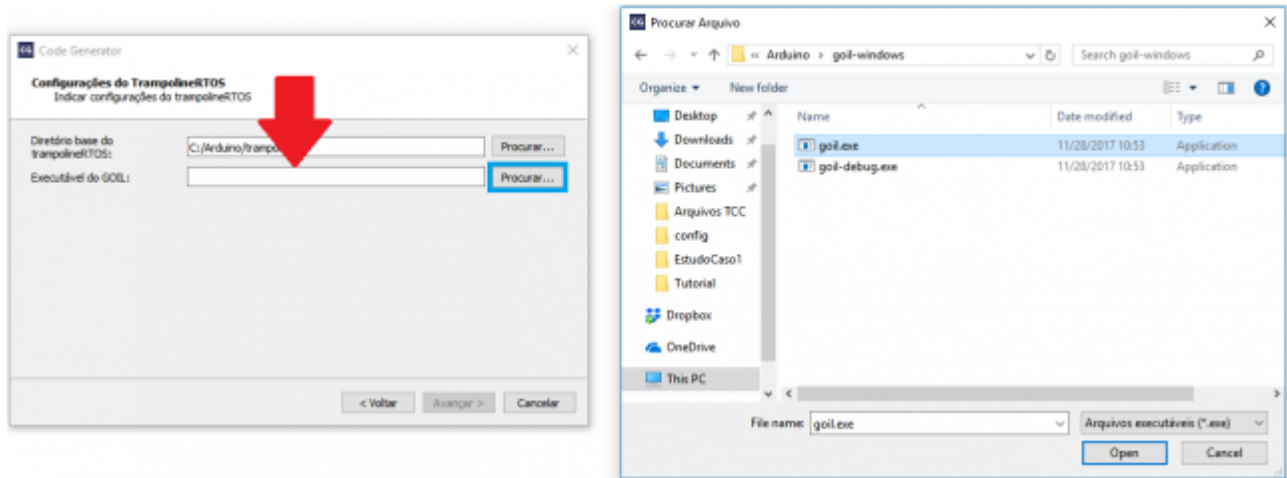
Na tela seguinte o usuário deve informar o diretório principal do **TrampolineRTOS** e o caminho para o executável do **GOIL**. Para auxiliar a preencher os caminhos, pode-se utilizar o explorador de arquivos através do botão **Procurar...** associado a cada campo de preenchimento.



- Para preencher o caminho do diretório base do **TrampolineRTOS** o usuário pode clicar no botão **Procurar...**



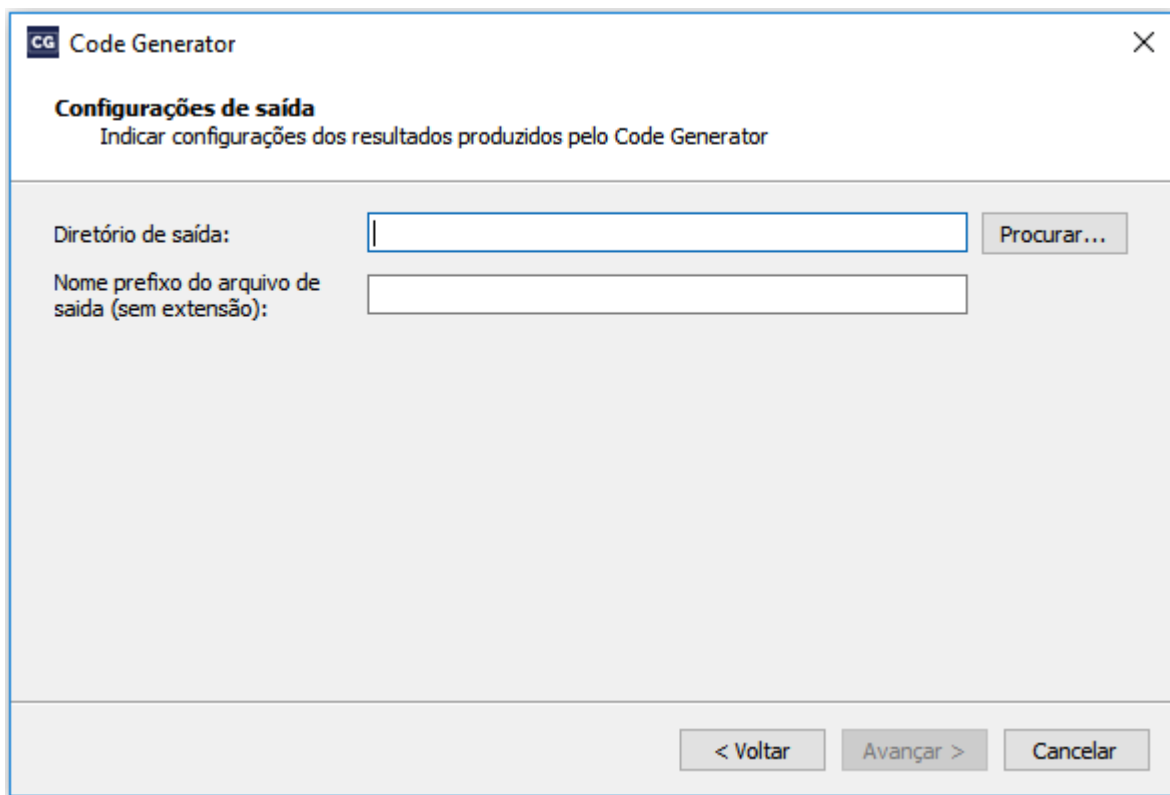
- Para preencher o caminho do executável do **GOIL** o usuário pode clicar no botão **Procurar...**



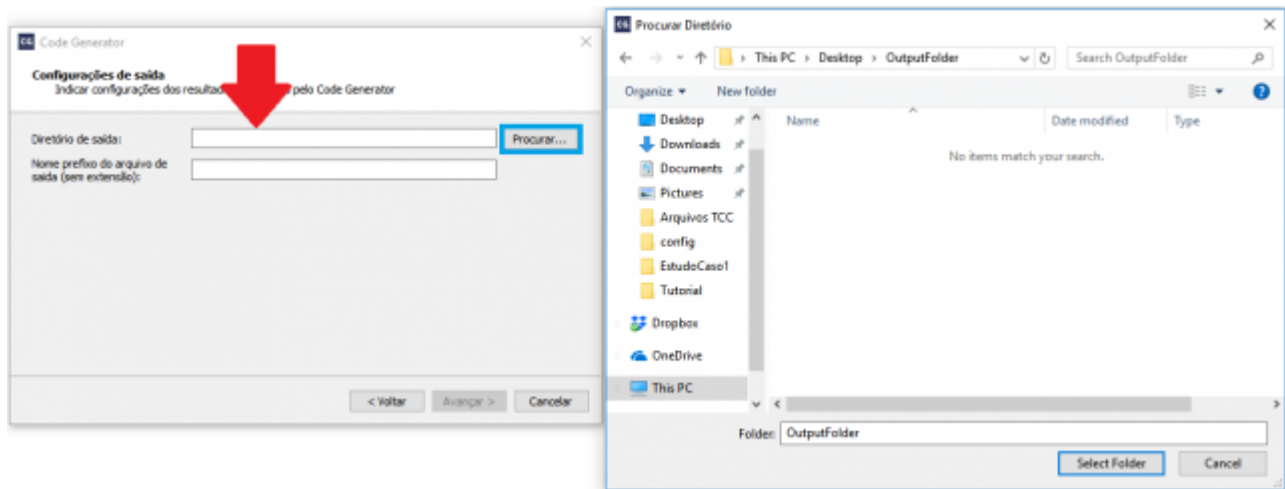
- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

5 - Tela de configurações dos arquivos de saída

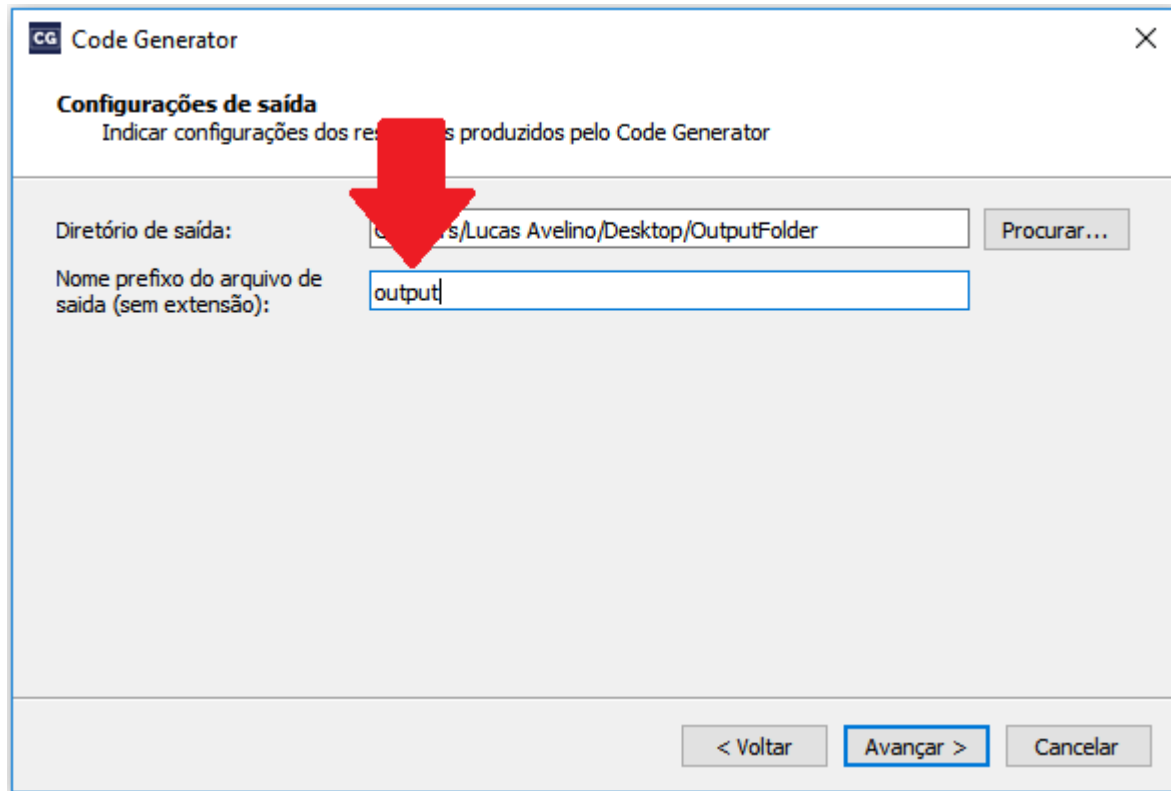
Na tela seguinte o usuário deve informar o diretório onde deve ser salvo os arquivos de código fonte e o binário produzidos. O usuário também deve informar o prefixo do nome dos arquivos produzidos.



- Para preencher o caminho do diretório de saída o usuário pode clicar no botão **Procurar....**



- O usuário deve preencher o prefixo dos arquivos de saída manualmente e este prefixo não pode conter extensão.



- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

6 - Tela de configurações dos pinos

Na tela seguinte o usuário pode realizar o mapeamento das teclas associadas as funções *key handlers* no

Busmaster para pinos no microcontrolador. Também pode selecionar a plataforma de hardware utilizada, indicar

o tipo de pino e caso o pino seja do tipo digital, informar qual nível lógico do pino corresponde ao estado ativo. A figura da plataforma de hardware com a informação da pinagem serve para auxiliar o usuário no mapeamento de teclas para pinos.

Nesta tela, apenas aparecem apenas as teclas associada a funções dos *key handlers* que foram extraídos dos arquivos de entrada.

CG

Code Generator


✕

Configuração dos pinos

Associar os pinos do arduino com as teclas relacionadas ao métodos OnKey gerados pelo busmaster

Arduino NANO

▼



Tecda

<i>:

<d>:

Tipo de entrada

digital

▼

Pino

4

▼

Ativo em nível lógico

1 (alto)

▼

Tipo de entrada

digital

▼

Pino

5

▼

Ativo em nível lógico

1 (alto)

▼

< Voltar

Avançar >

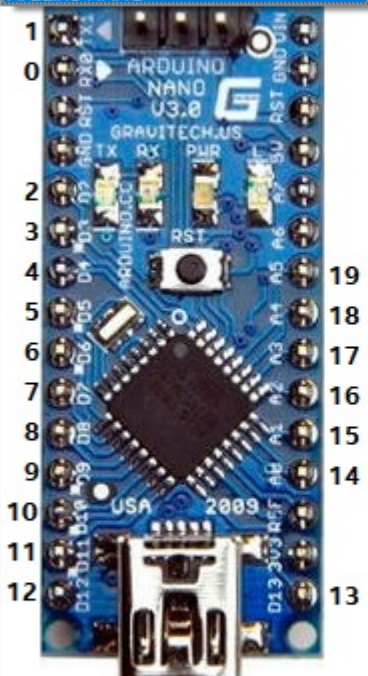
Cancelar

- Para selecionar a plataforma de hardware o usuário deve clicar no campo de seleção acima da imagem do arduino.

CG Code Generator

Configuração dos pinos
Associar os pinos do arduino com as teclas relacionadas ao métodos OnKey gerados pelo busmaster

Arduíno NANO
Arduíno UNO
Arduíno NANO



<i>:
<d>:

Tipo de entrada	Pino	Ativo em nível lógico
digital	4	1 (alto)
digital	5	1 (alto)

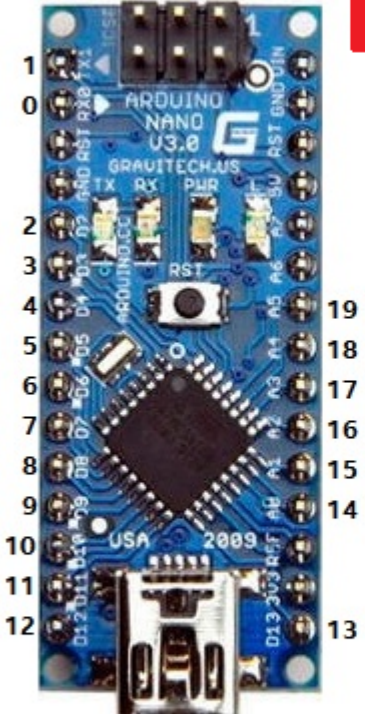
< Voltar Avançar > Cancelar

- Para seleccionar o tipo de entrada deve-se clicar no campo de seleção do tipo de entrada para cada tecla e seleccionar o tipo adequado (digital ou analógico).

Code Generator

Configuração dos pinos
Associar os pinos do arduino com as teclas relacionadas ao métodos OnKey gerados pelo busmaster

Arduino NANO



Tecla **<d>**

Tipo de entrada	Pino	Ativo em nível lógico
digital	4	1 (alto)
digital	5	1 (alto)
analógico		

< Voltar Avançar > Cancelar

- Para selecionar o pino associado a cada tecla deve-se clicar no campo de seleção do pino. Os pinos permitidos para seleção estão de acordo com o tipo de entrada (digital ou analógica). E os pinos digitais

“2,10,11,12,13” não podem ser selecionados, pois são utilizados na ligação com o módulo MCP2515.

cg


Code Generator

✕

Configuração dos pinos

Associar os pinos do arduino com as teclas relacionadas ao métodos OnKey gerados pelo busmaster

Arduino NANO



Tecla

<|>:

<d>:

Tipo de entrada

digital

Pino

4

0

1

3

4

5

6

7

8

9

Ativo em nível lógico

1 (alto)

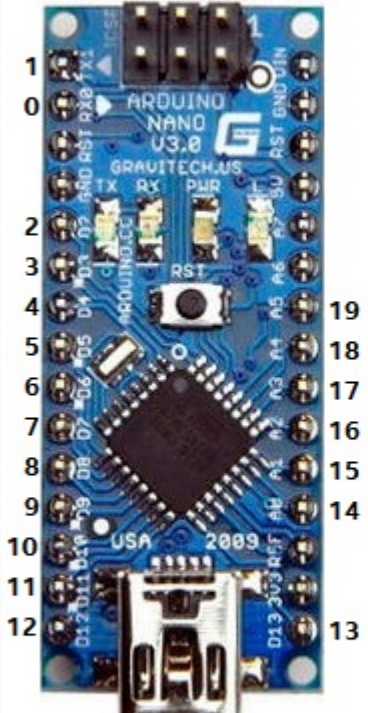
1 (alto)

< Voltar

Avançar >

Cancelar

- Para selecionar o nível lógico do pino digital que corresponde ao estado ativo para cada pino deve-se clicar no campo de seleção e escolher entre “1 (*alto*)” ou “0 (*baixo*)”.



Arduíno NANO

Tecda

<i>:

<d>:

Tipo de entrada

digital

digital

Pino

4

5

Ativo em nível lógico

1 (alto)

1 (alto)

0 (baixo)

< Voltar

Avançar >

Cancelar

- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

7 - Tela de configurações do código para a plataforma de hardware

Na tela seguinte o usuário pode realizar uma série de configurações a respeito do código fonte produzido para a plataforma de hardware.

O usuário pode informar se a ECU em questão envia mensagens pelo barramento CAN, possibilitando a instanciação da fila de envio de mensagens e da *task* responsável pelo envio.

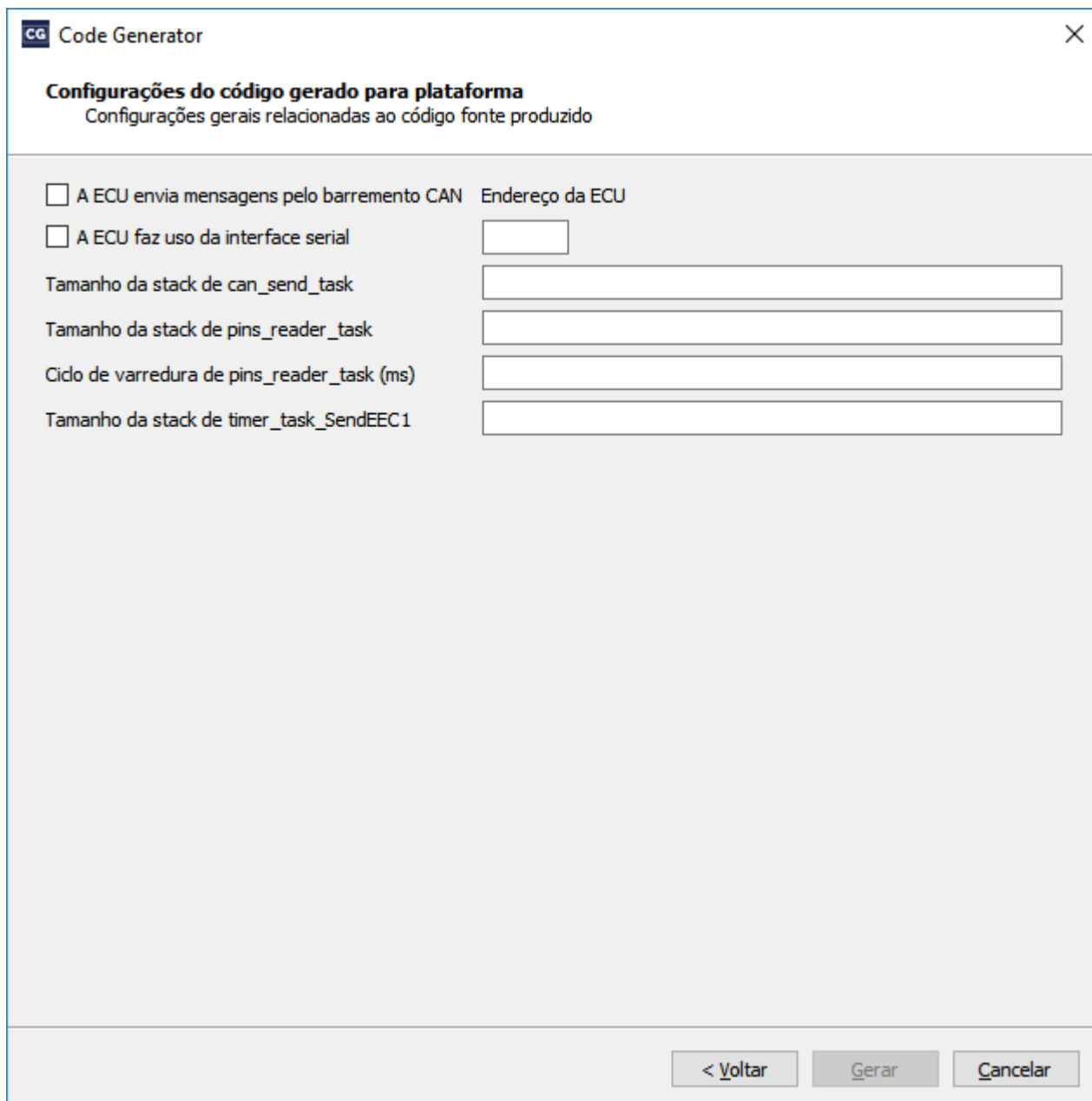
O usuário pode informar se a ECU em questão faz uso da interface serial, para que a mesma possa ser devidamente

inicializada.

O usuário deve informar o endereço da ECU em base hexadecimal (0 a FF).

O usuário pode informar o tamanho da pilha de controle, em valor decimal e múltiplo de 2, de cada *task* que será instanciada pela aplicação. Entretanto, cada *task* possui um tamanho de pilha de controle pré-definido e que será utilizado caso o usuário não deseje especificá-lo. Os tamanhos pré-definidos das pilhas de controle das *tasks* são:

- Para a *can_send_task* o tamanho padrão é de 128 bytes.
- Para a *pins_reader_task* o tamanho padrão é de 128 bytes.
- Para a *can_rcv_task* o tamanho padrão é de 256 bytes.
- Para qualquer *timer_task* o tamanho padrão é de 128 bytes.



The screenshot shows a window titled "Code Generator" with a close button (X) in the top right corner. Below the title bar, the text "Configurações do código gerado para plataforma" is displayed, followed by a subtitle "Configurações gerais relacionadas ao código fonte produzido". The main area contains several configuration options:

- ☐ A ECU envia mensagens pelo barramento CAN
- ☐ A ECU faz uso da interface serial
- Endereço da ECU: [text input field]
- Tamanho da stack de *can_send_task*: [text input field]
- Tamanho da stack de *pins_reader_task*: [text input field]
- Ciclo de varredura de *pins_reader_task* (ms): [text input field]
- Tamanho da stack de *timer_task_SendEEC1*: [text input field]

At the bottom right, there are three buttons: "< Voltar", "Gerar", and "Cancelar".

- Na Figura a seguir observamos um exemplo de preenchimento dos campos.

The screenshot shows a window titled "Code Generator" with a close button (X) in the top right corner. Below the title bar, the text "Configurações do código gerado para plataforma" is displayed, followed by "Configurações gerais relacionadas ao código fonte produzido". The main area contains several configuration options:

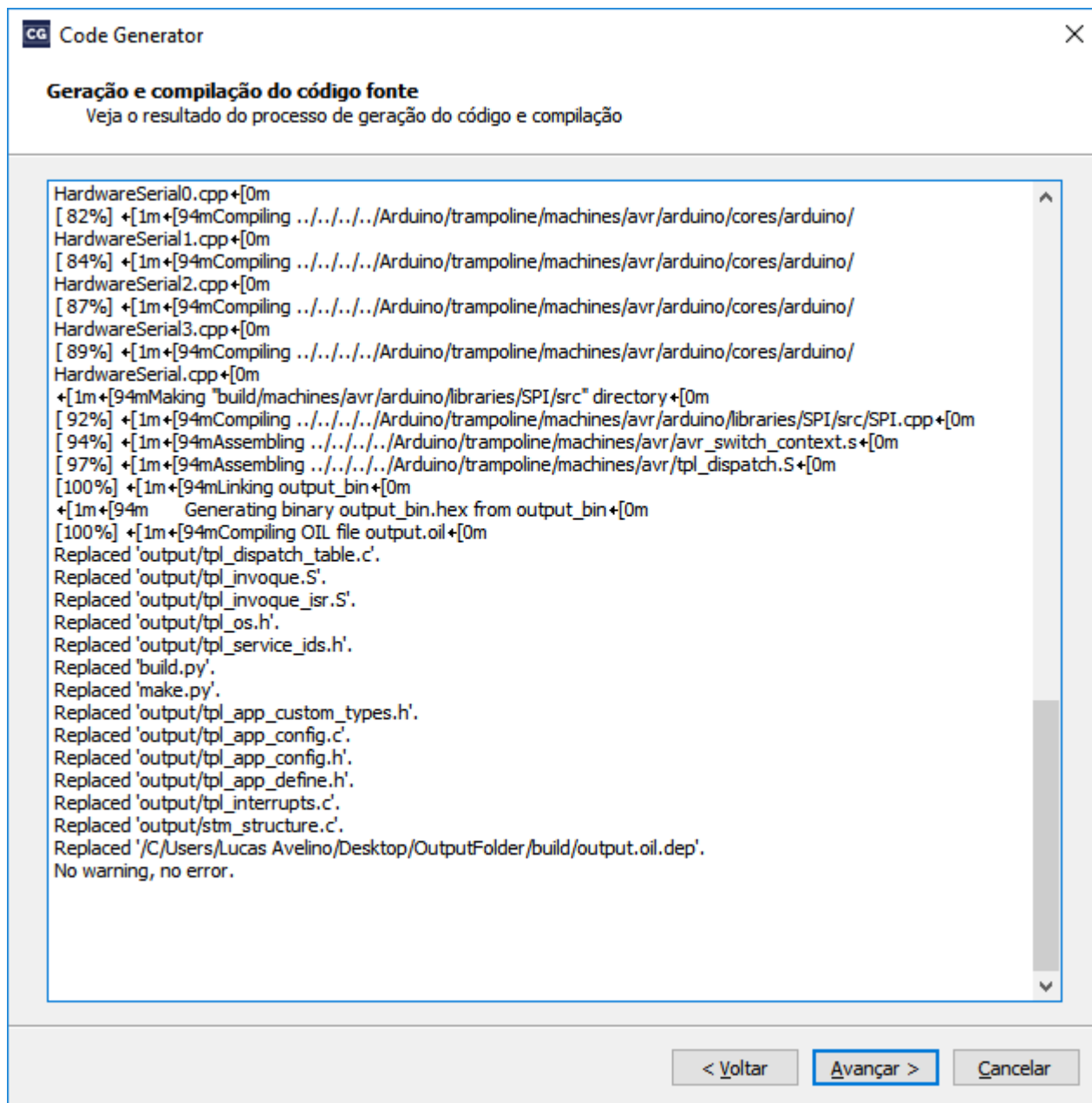
- ☒ A ECU envia mensagens pelo barramento CAN
- ☐ A ECU faz uso da interface serial
- Tamanho da stack de can_send_task
- Tamanho da stack de pins_reader_task
- Ciclo de varredura de pins_reader_task (ms)
- Tamanho da stack de timer_task_SendEEC1

To the right of these options, there are input fields. The first field, labeled "Endereço da ECU", contains the value "0". The other fields are empty. At the bottom right, there are three buttons: "< Voltar", "Gerar", and "Cancelar". The "Gerar" button is highlighted with a blue border.

- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

8 - Tela com o resultado da geração de código e compilação

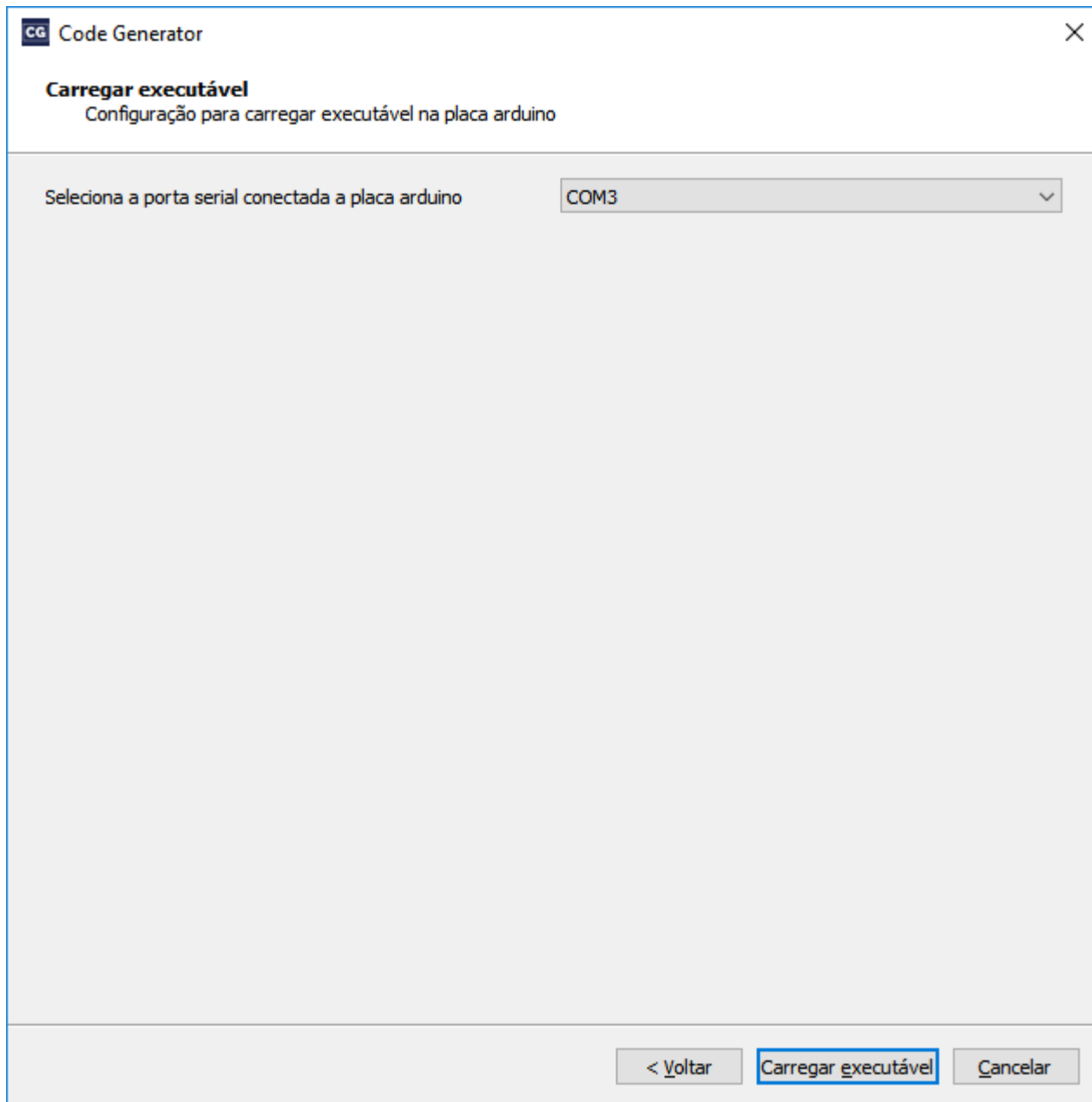
Na tela seguinte o usuário pode observar os resultados da geração de código e compilação.



- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

9 - Tela para o carregamento do executável

Na tela seguinte o usuário pode informar a porta serial a qual o arduino está conectado através da caixa de seleção.



CG Code Generator

Carregar executável
Configuração para carregar executável na placa arduino

Selecione a porta serial conectada a placa arduino

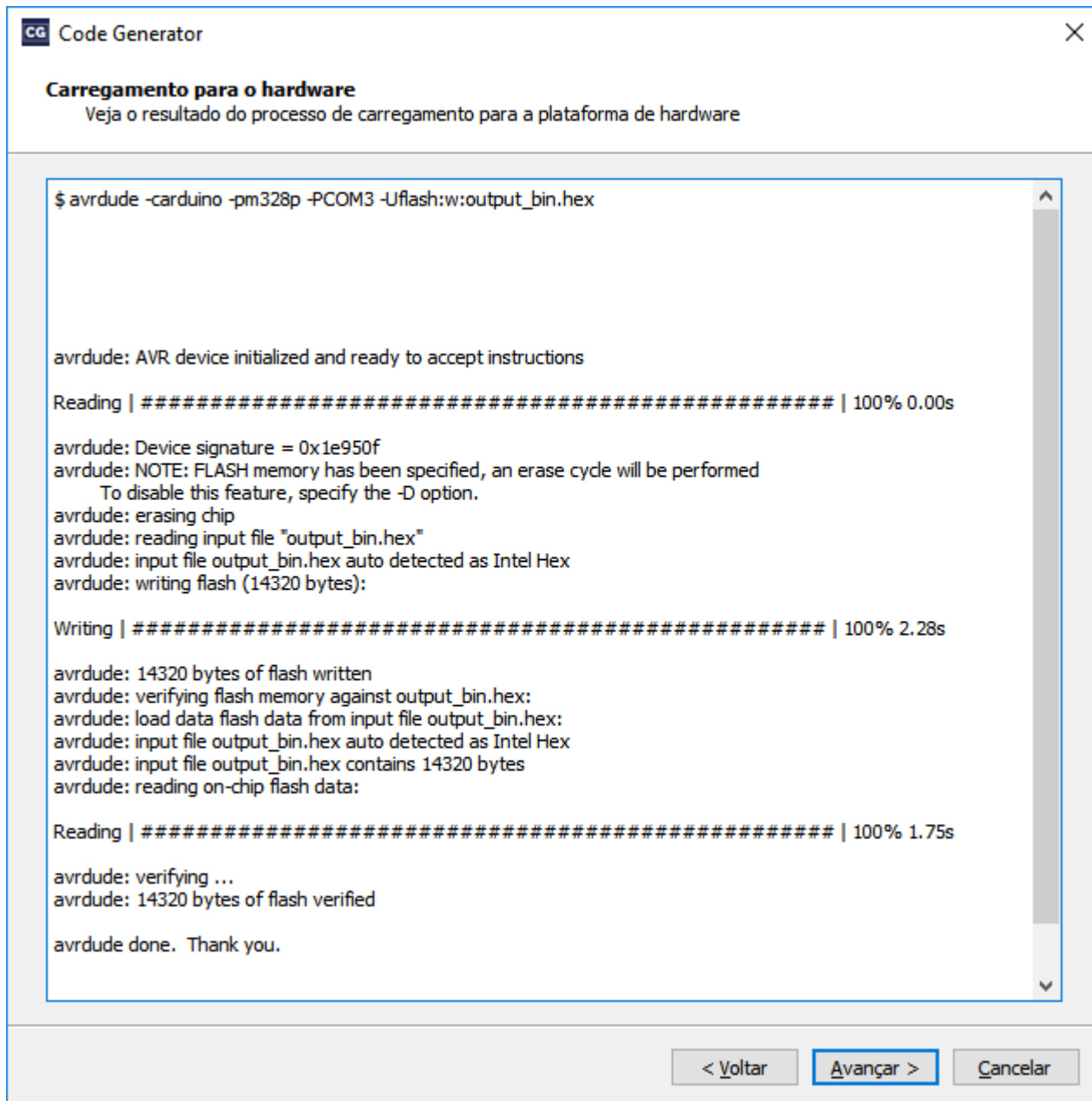
COM3

< Voltar Carregar executável Cancelar

- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

10 - Tela com o resultado do carregamento

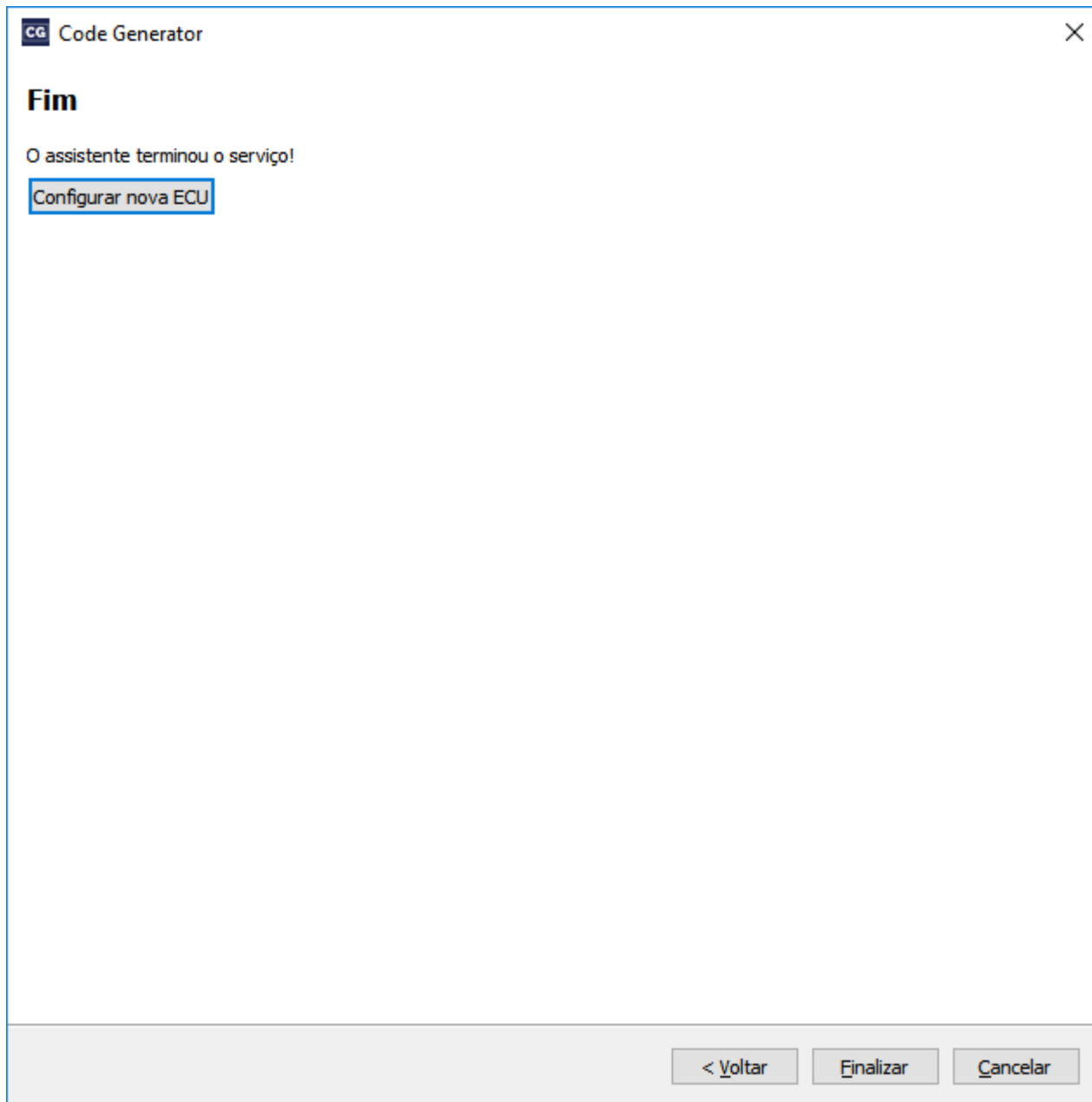
Na tela seguinte o usuário pode observar o resultado do carregamento do código executável para a plataforma de hardware.



- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

11 - Tela final

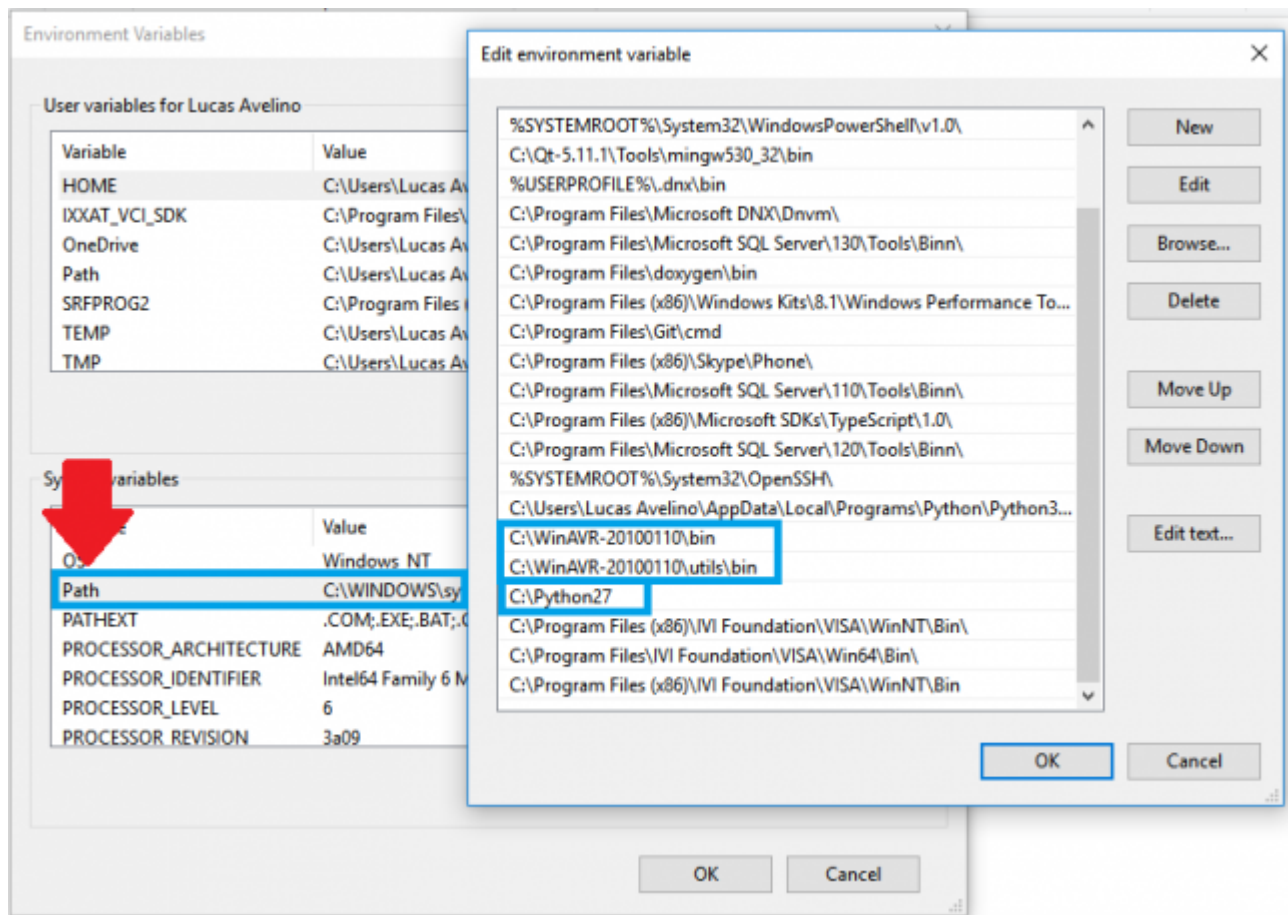
Na última tela o usuário pode clicar em **Finalizar** caso deseje encerrar ou clicar no botão **Configurar nova ECU** caso deseje prosseguir com a configuração de uma nova ECU.



Dependências

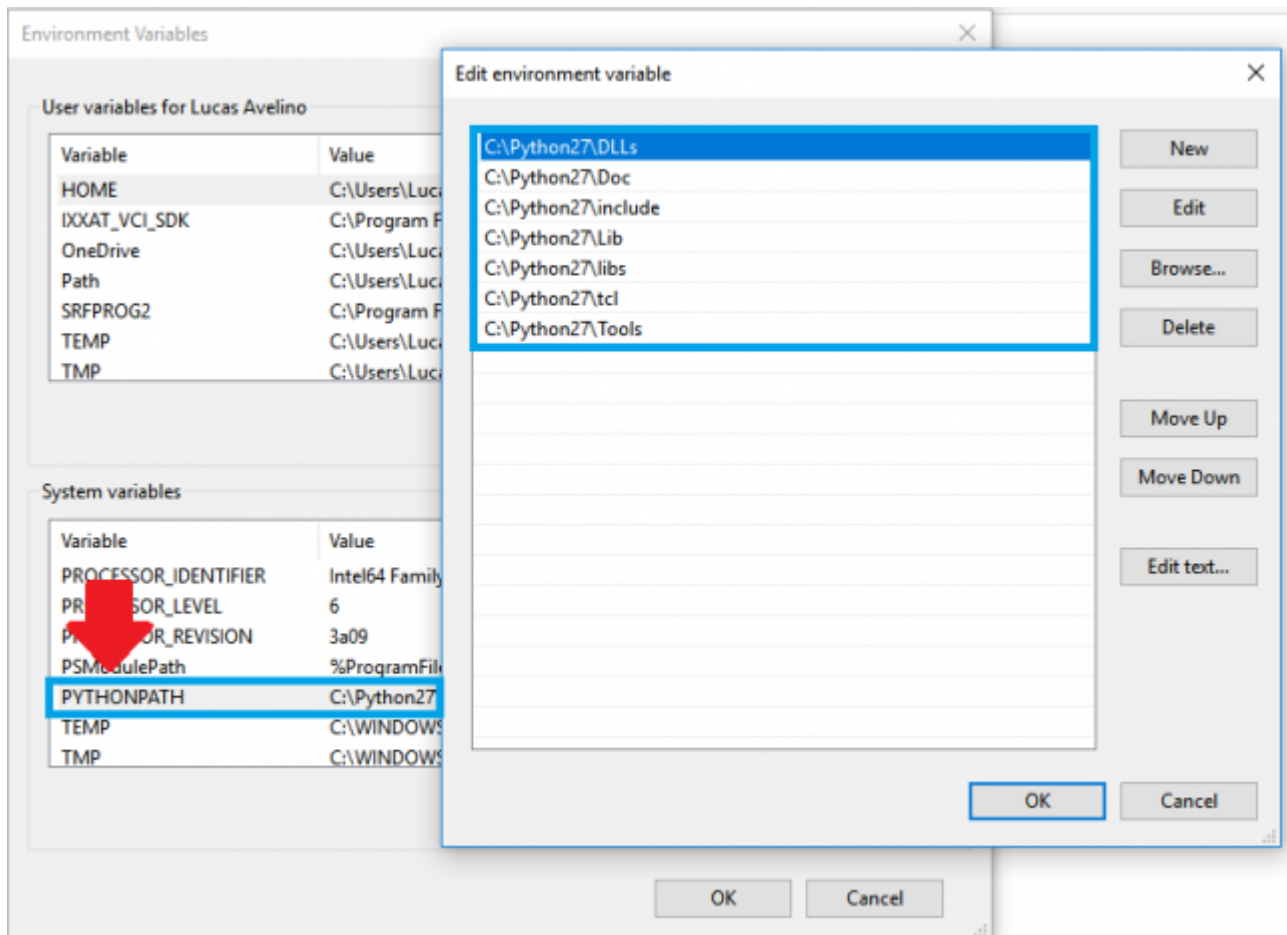
O assistente Code Generator faz uso do **TrampolineRTOS**, **Busmaster**, **Toolchain do AVR**, **Python** dentre outras ferramentas. Portanto é necessário que cada uma delas esteja devidamente configurada no ambiente.

- Devem estar presentes na variável de ambiente **Path** os diretórios de binários da toolchain do AVR e o diretório do executável do Python. Exemplo:



- Para que os scripts Python possam ser executados através do assistente é necessário a definição da variável de ambiente **PYTHONPATH** com os diretórios internos ao diretório base da instalação do

Python. Exemplo:



- Para utilizar a biblioteca MCP_CAN é necessário que ela esteja localizada no caminho `$TRAMPOLINE_ROOT_DIR$/machines/avr/arduino/libraries/mcp_can` e que esteja corretamente configurada no trampoline. Para isso, é necessário que no arquivo `$TRAMPOLINE_ROOT_DIR$/goil/templates/config/avr/arduino/config.oil` seja adicionada as configurações da biblioteca.

```
LIBRARY mcp_can {
    NEEDS = spi;
    PATH = "avr/arduino/libraries/mcp_can";
    CPPHEADER = "mcp_can_dfs.h";
    CPPHEADER = "mcp_can.h";
    CPPFILE = "mcp_can.cpp";
};
```