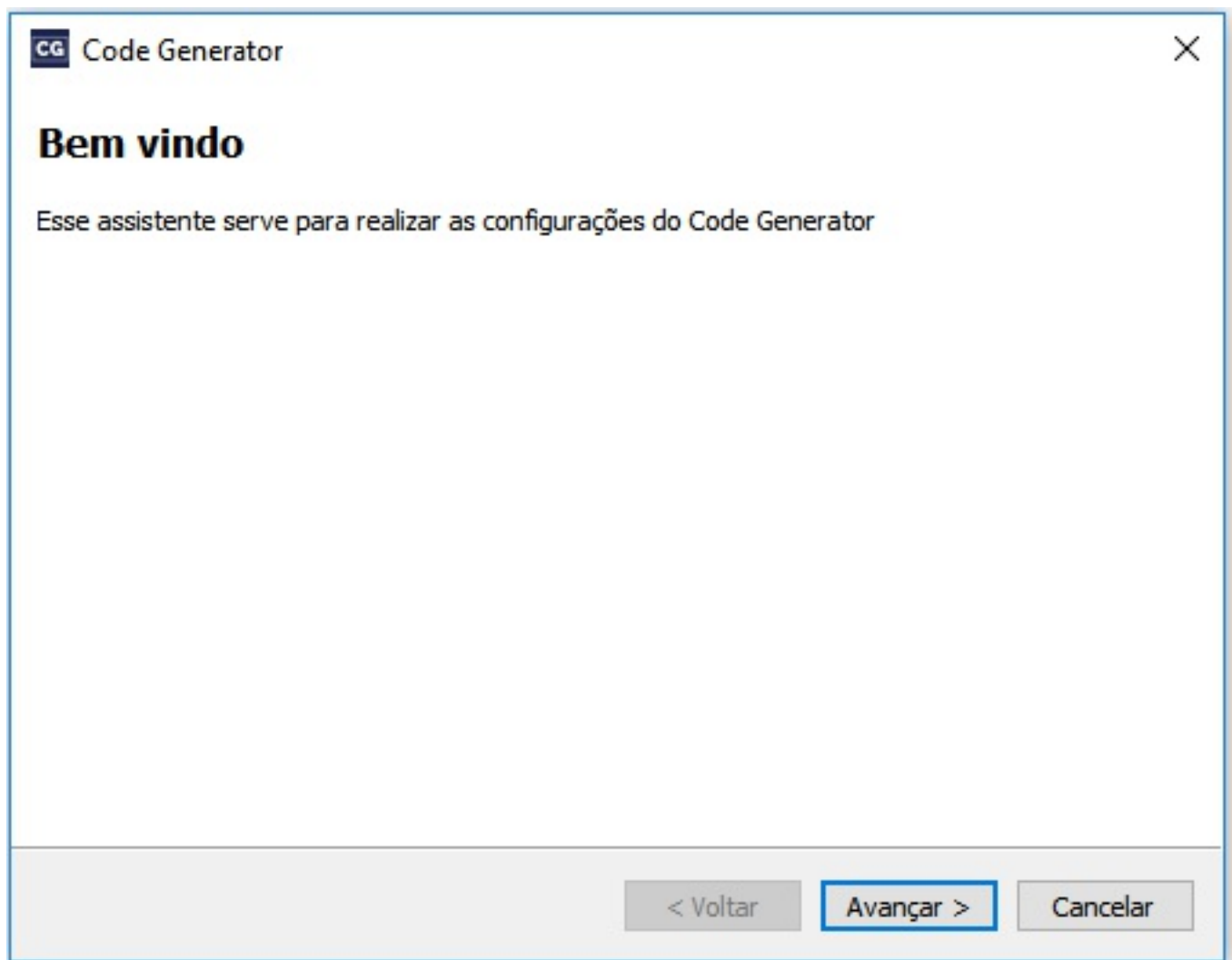


Tutorial do Code Generator

1 - Tela inicial

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

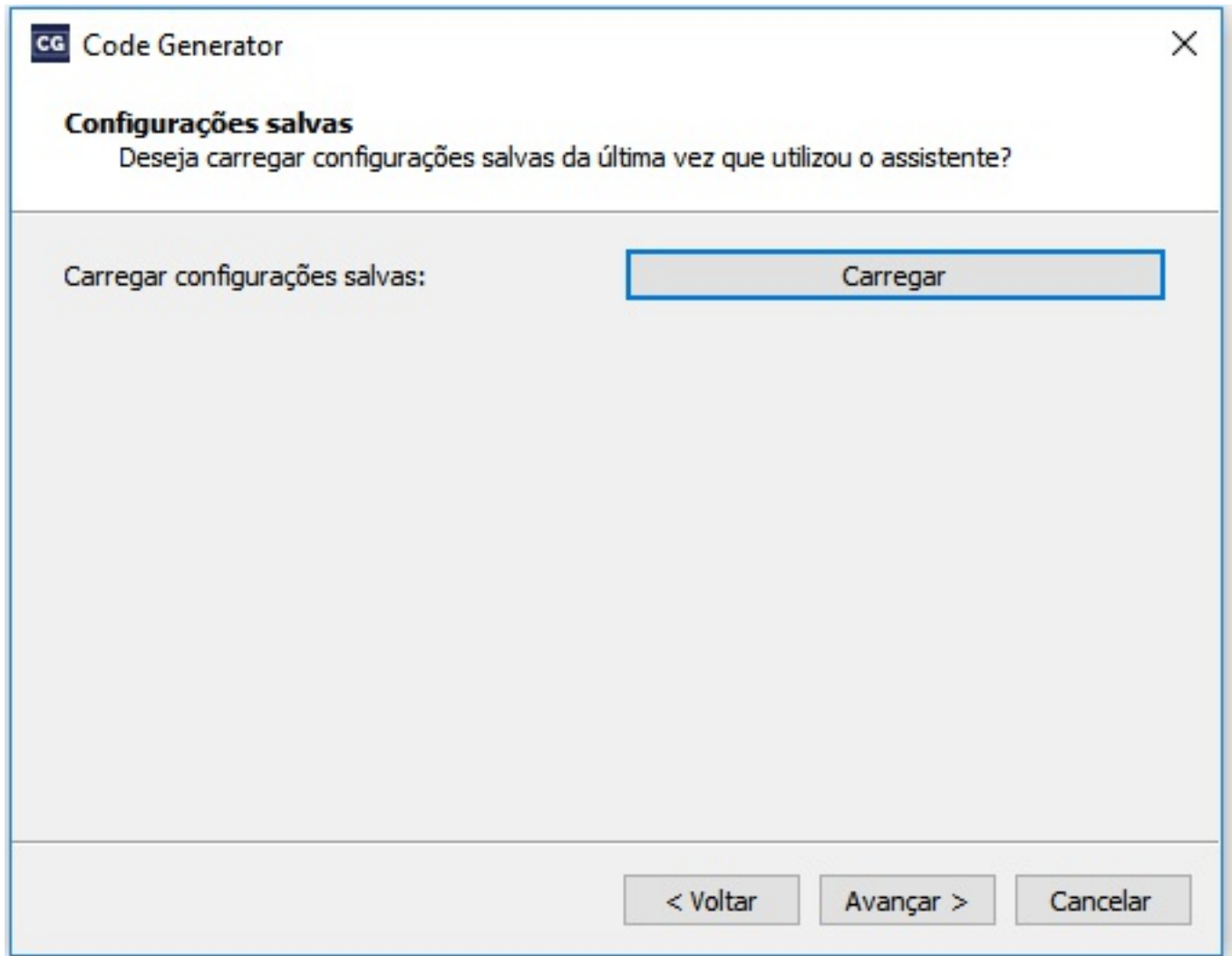


- Para prosseguir o usuário deve clicar no botão ***Avançar***.

2 - Tela para carregar

configurações salvas

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



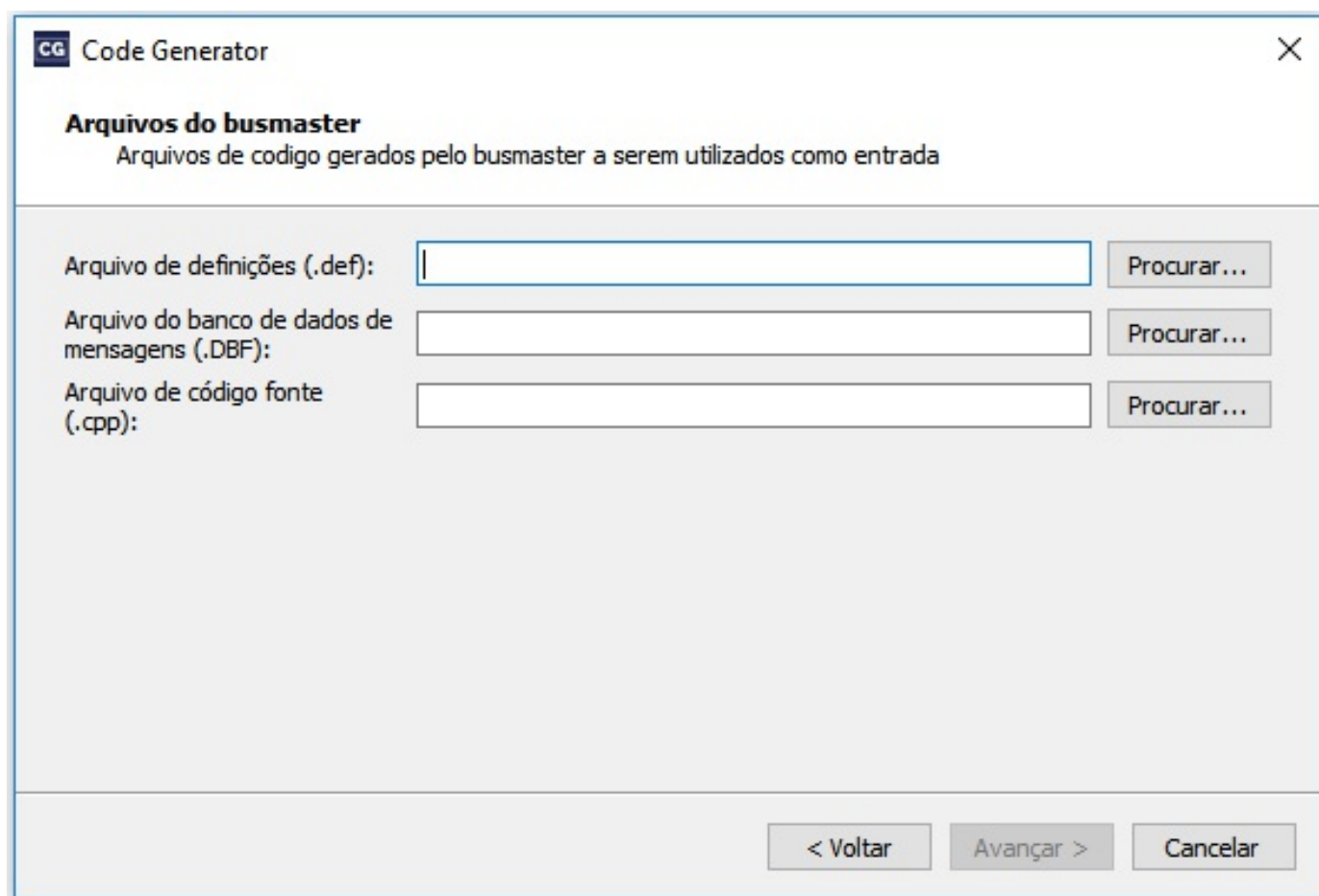
The image shows a Windows-style dialog box titled "Code Generator" with a close button (X) in the top right corner. The main title is "Configurações salvas" (Saved configurations). Below the title is a question: "Deseja carregar configurações salvas da última vez que utilizou o assistente?" (Do you want to load saved configurations from the last time you used the assistant?). The dialog has a light gray background. On the left, the text "Carregar configurações salvas:" is followed by a large, empty rectangular area. To the right of this area is a button labeled "Carregar". At the bottom of the dialog, there are three buttons: "< Voltar" (Back), "Avançar >" (Next), and "Cancelar" (Cancel).

- 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.



CG Code Generator

Arquivos do busmaster
Arquivos de código gerados pelo busmaster a serem utilizados como entrada

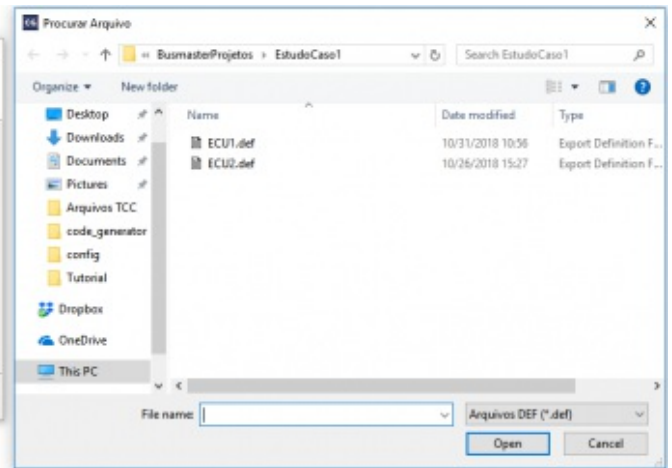
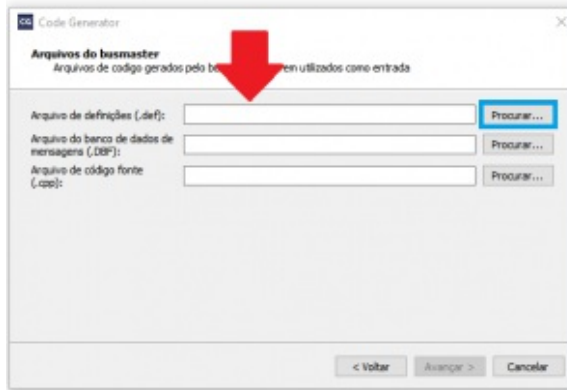
Arquivo de definições (.def): **Procurar...**

Arquivo do banco de dados de mensagens (.DBF): **Procurar...**

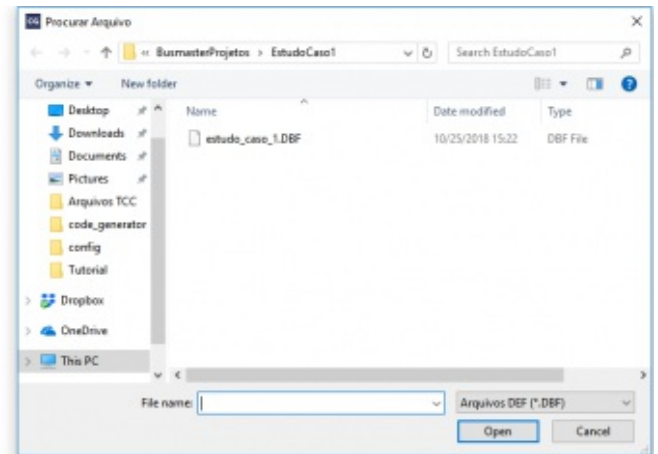
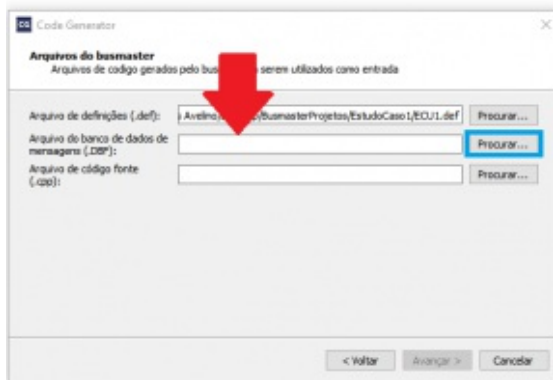
Arquivo de código fonte (.cpp): **Procurar...**

< Voltar **Avançar >** **Cancelar**

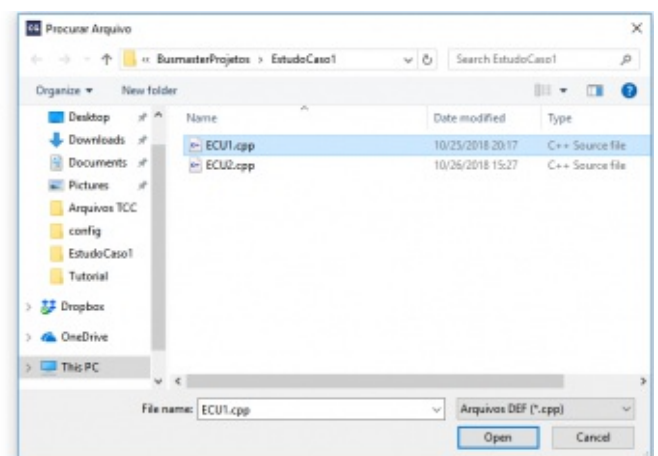
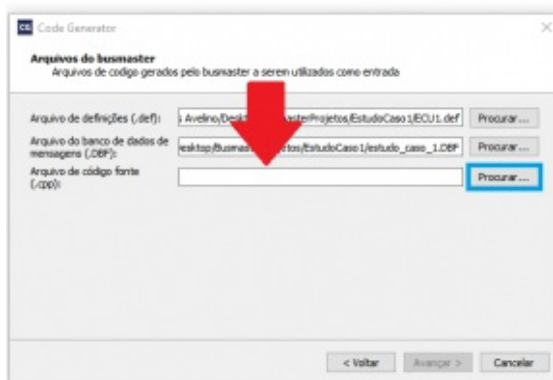
- Para selecionar o arquivo de definições pode-se clicar no botão ***Procurar...*** a ele associado.



- 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.

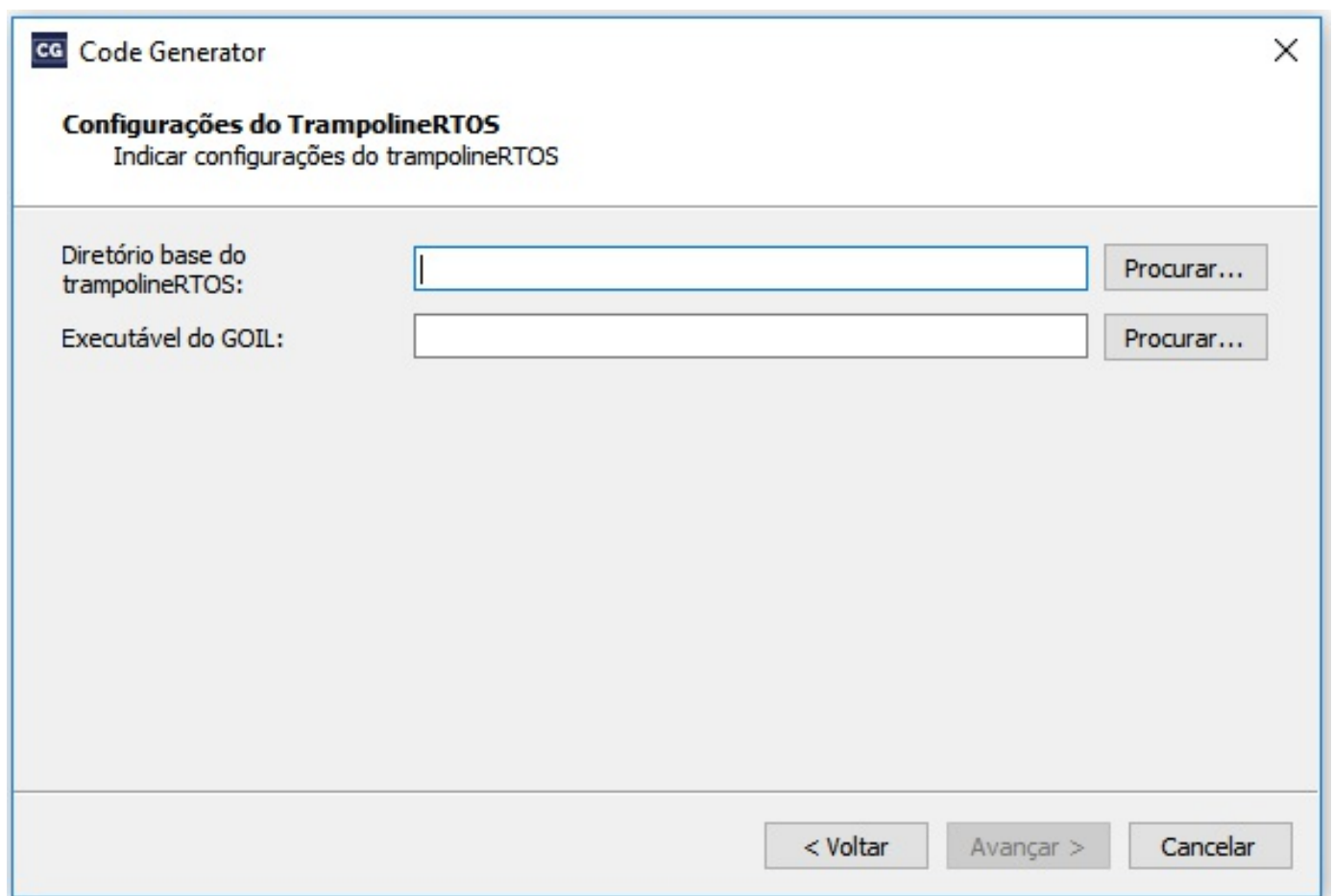


- Ao concluir o preenchimento dos campos o usuário deve clicar

em **Avançar** para prosseguir.

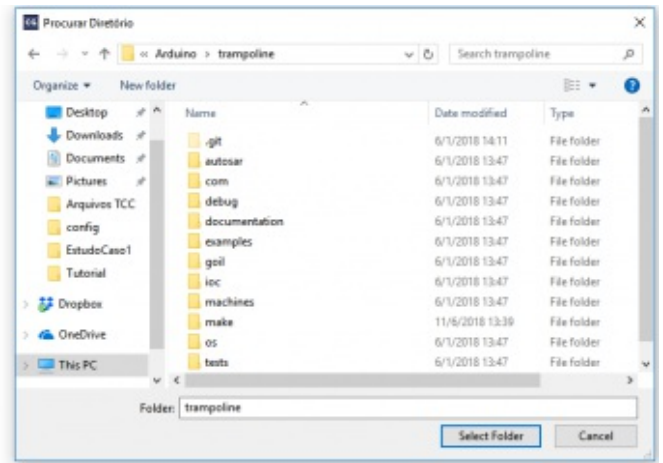
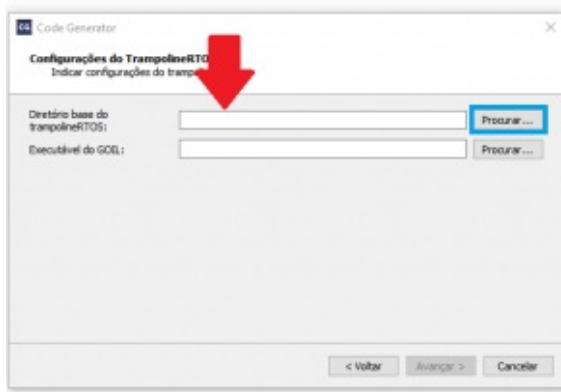
4 - Tela para configurações do TrampolineRTOS

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.

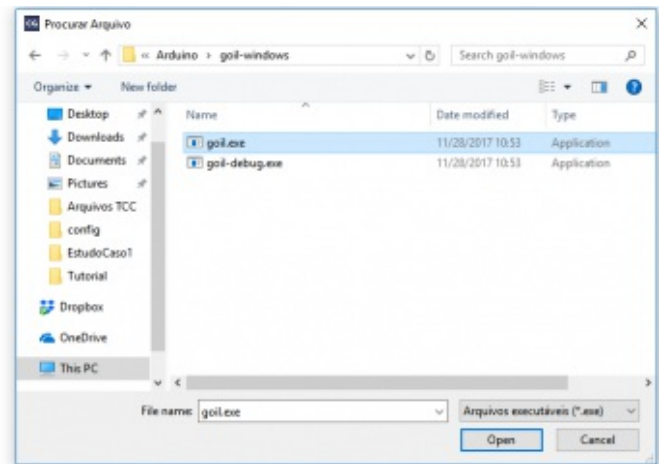
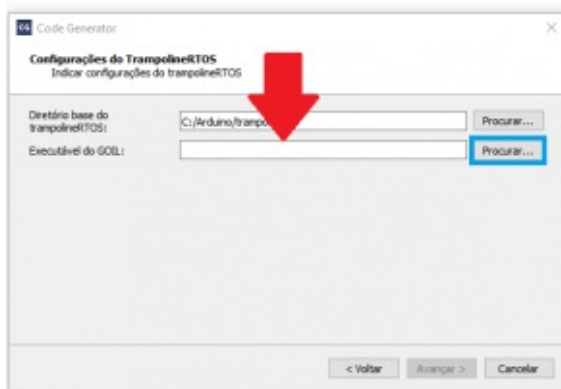


The screenshot shows a Windows-style dialog box titled "Code Generator" with a close button (X) in the top right corner. The main title is "Configurações do TrampolineRTOS" with a subtitle "Indicar configurações do trampolineRTOS". The dialog contains two rows of configuration fields. The first row is labeled "Diretório base do trampolineRTOS:" and has a text input field followed by a "Procurar..." button. The second row is labeled "Executável do GOIL:" and also has a text input field followed by a "Procurar..." button. At the bottom of the dialog, there are three buttons: "< Voltar", "Avançar >", and "Cancelar".

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



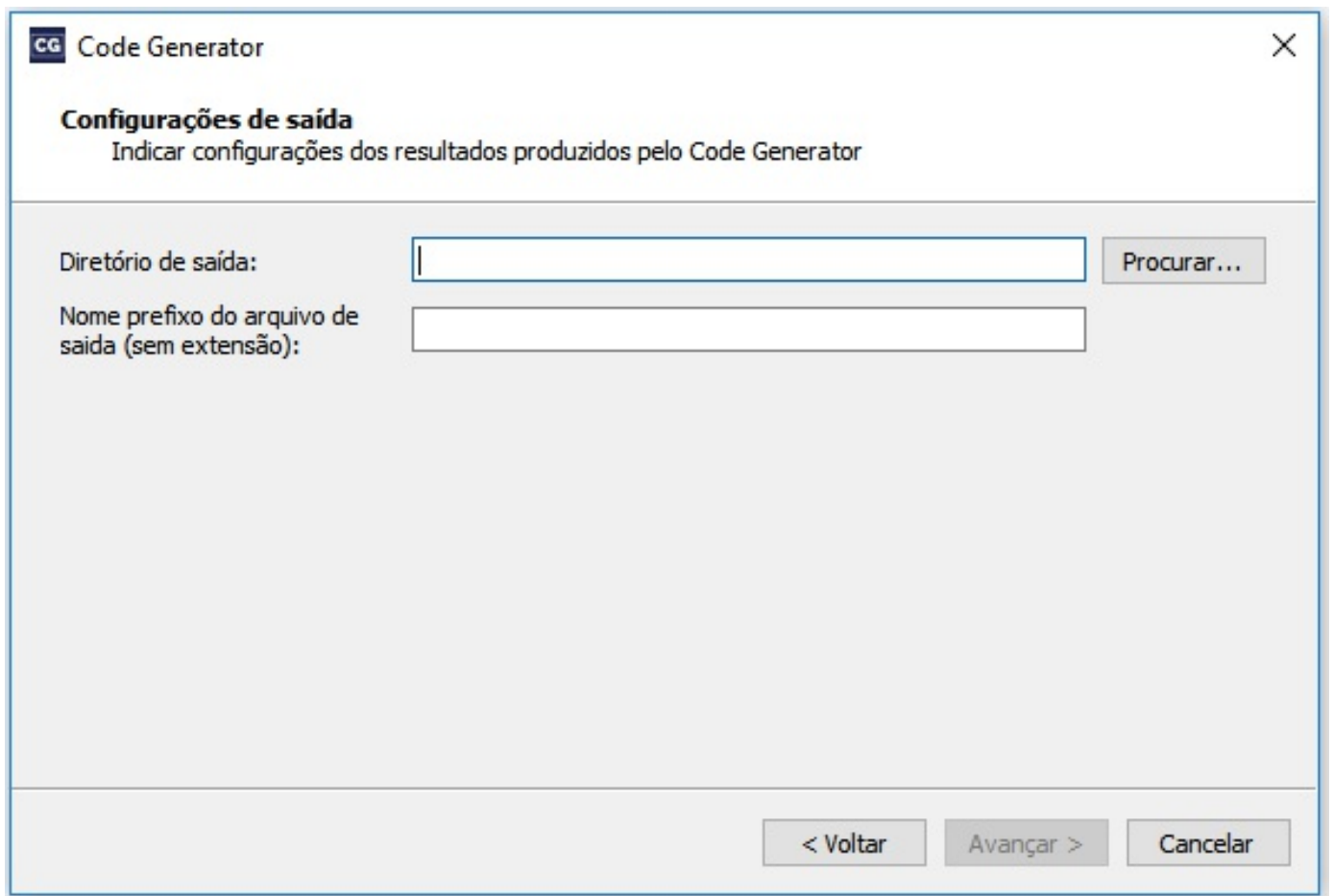
- Para preencher o caminho do executável do **GOIL** o usuário pode clicar no botão ***Procurar...***



- 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.



CG Code Generator

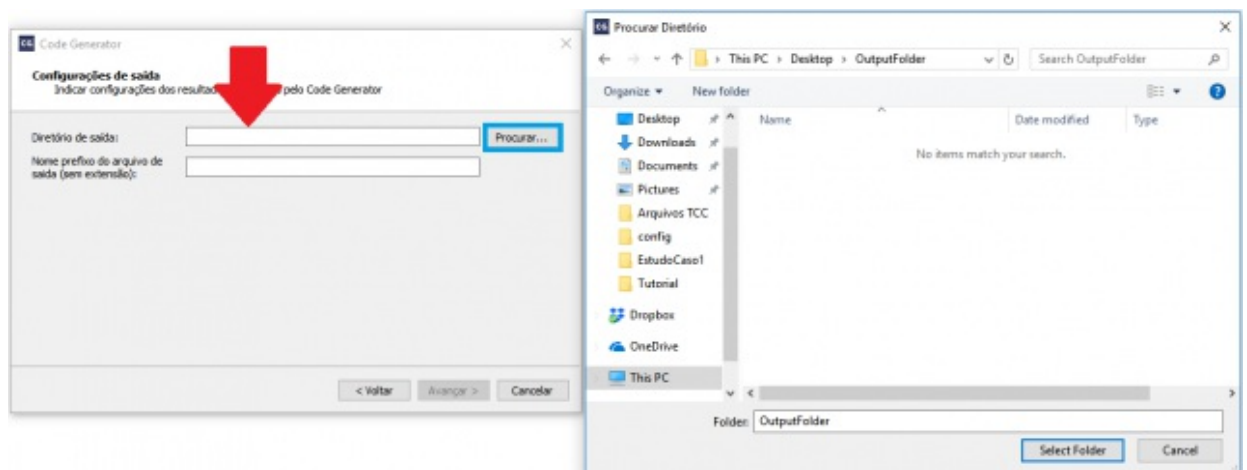
Configurações de saída
Indicar configurações dos resultados produzidos pelo Code Generator

Diretório de saída: Procurar...

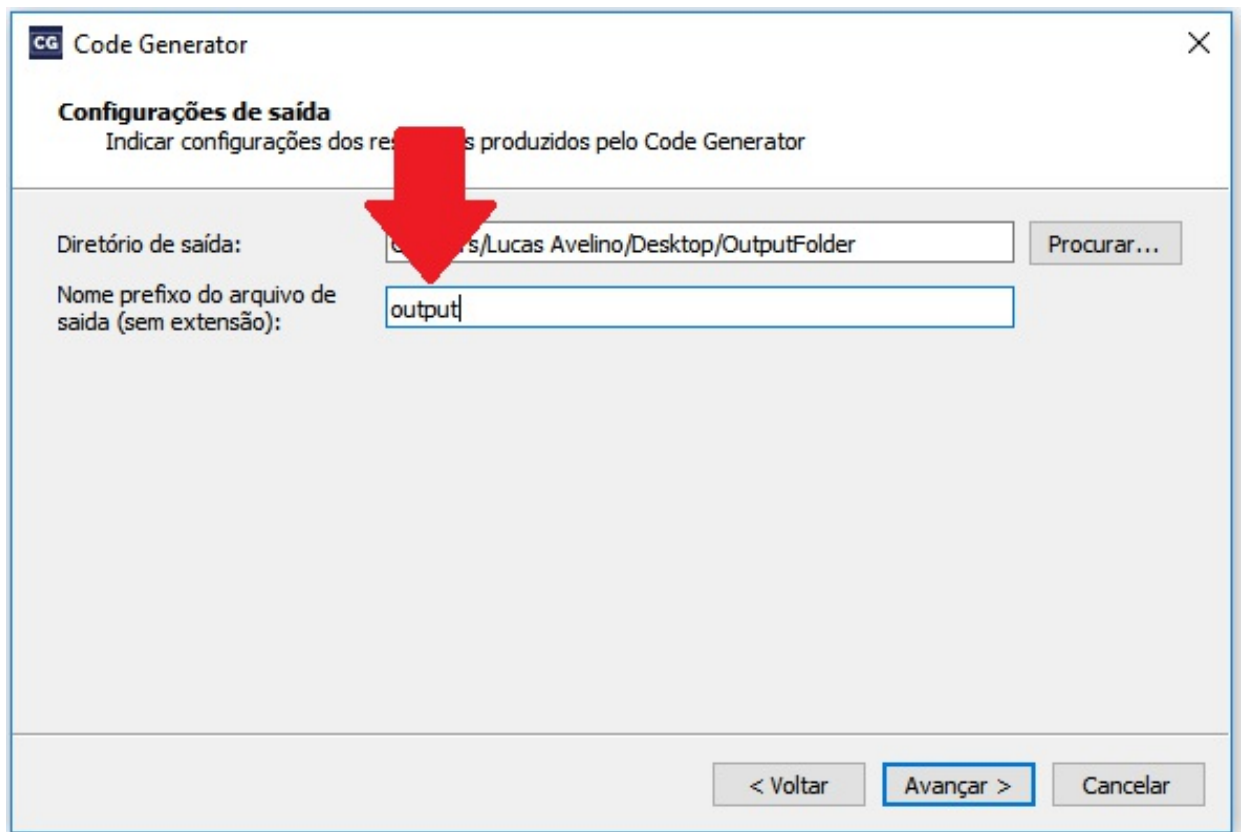
Nome prefixo do arquivo de saída (sem extensão):

< Voltar Avançar > Cancelar

- 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.

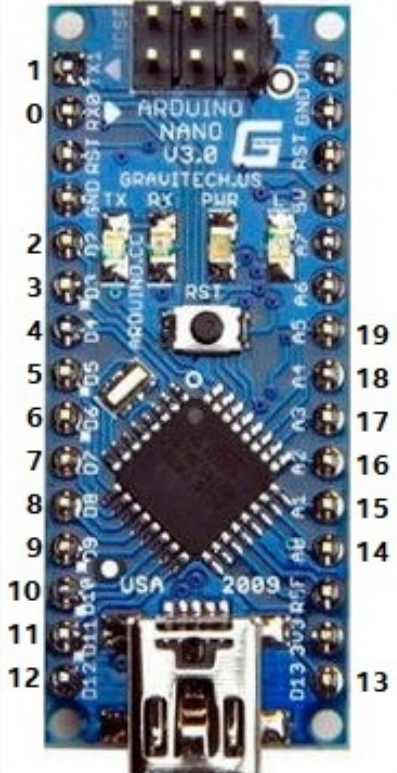
CG

Code Generator

✕

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

Arduíno NANO



Teda

<i>:

<d>:

Tipo de entrada

Pino

Ativo em nível lógico

digital

4

1 (alto)

digital

5

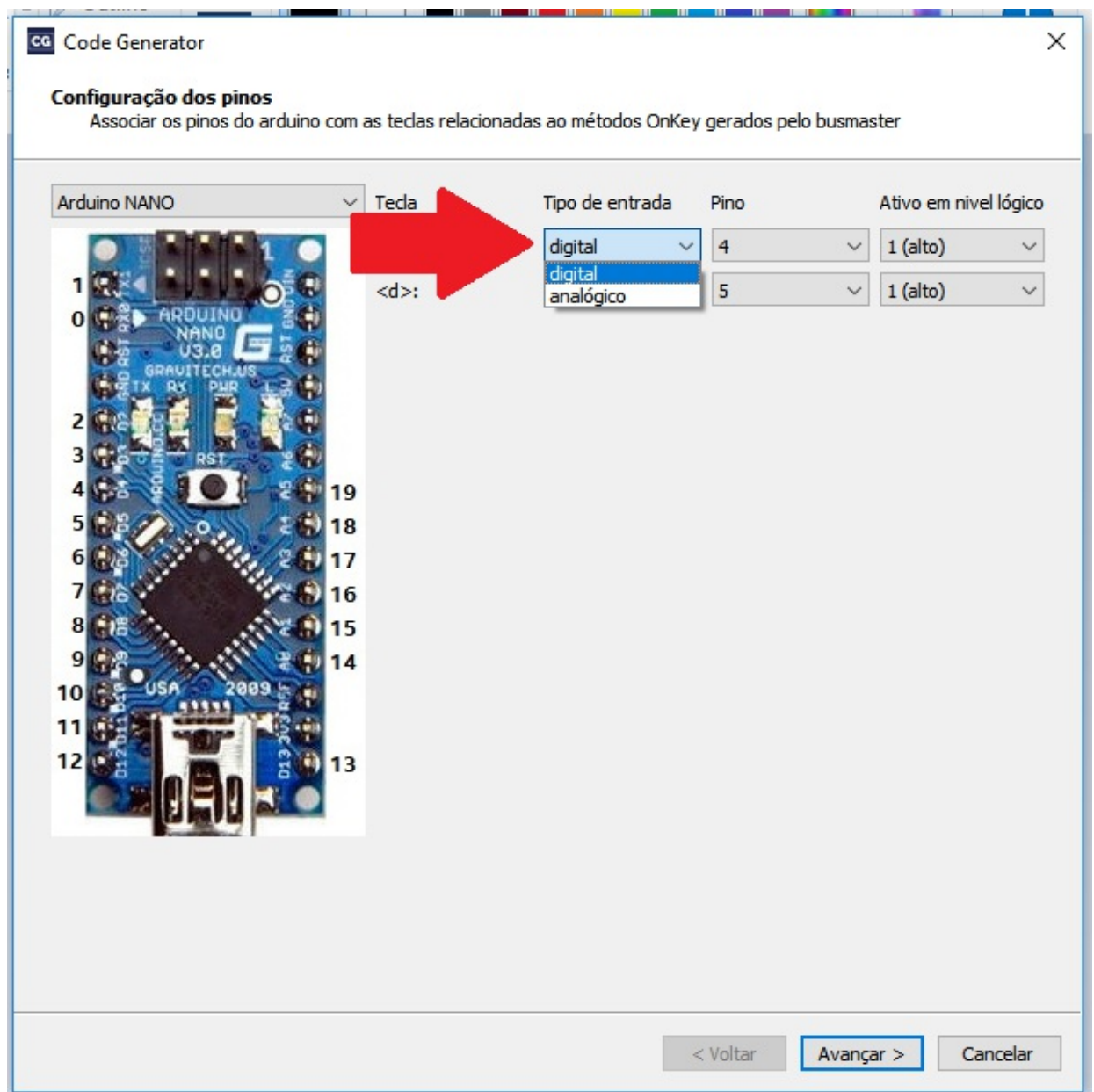
1 (alto)

< Voltar

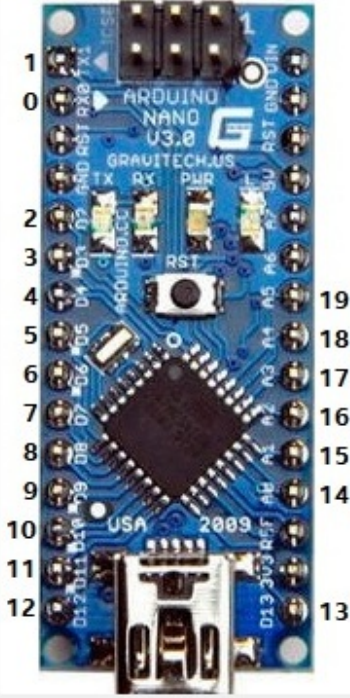
Avançar >

Cancelar

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



- Para seleccionar o pino associado a cada tecla deve-se clicar no campo de selecção do pino. Os pinos permitidos para selecção estão de acordo com o tipo de entrada (digital ou analógica). E os pinos digitais "2,10,11,12,13" não podem ser seleccionados, pois são utilizados na ligação com o módulo MCP2515.



Teda	Tipo de entrada	Pino	Ativo em nível lógico
< >:	digital	4	1 (alto)
<d>:	digital	5	1 (alto)

< 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_recv_task* o tamanho padrão é de 256 bytes.
- Para qualquer *timer_task* o tamanho padrão é de 128 bytes.

CG

Code Generator

✕

Configurações do código gerado para plataforma
Configurações gerais relacionadas ao código fonte produzido

☐ A ECU envia mensagens pelo barramento CAN

Endereço da ECU

☐ 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

< Voltar

Gerar

Cancelar

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

The screenshot shows a window titled 'Code Generator' with a close button 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:

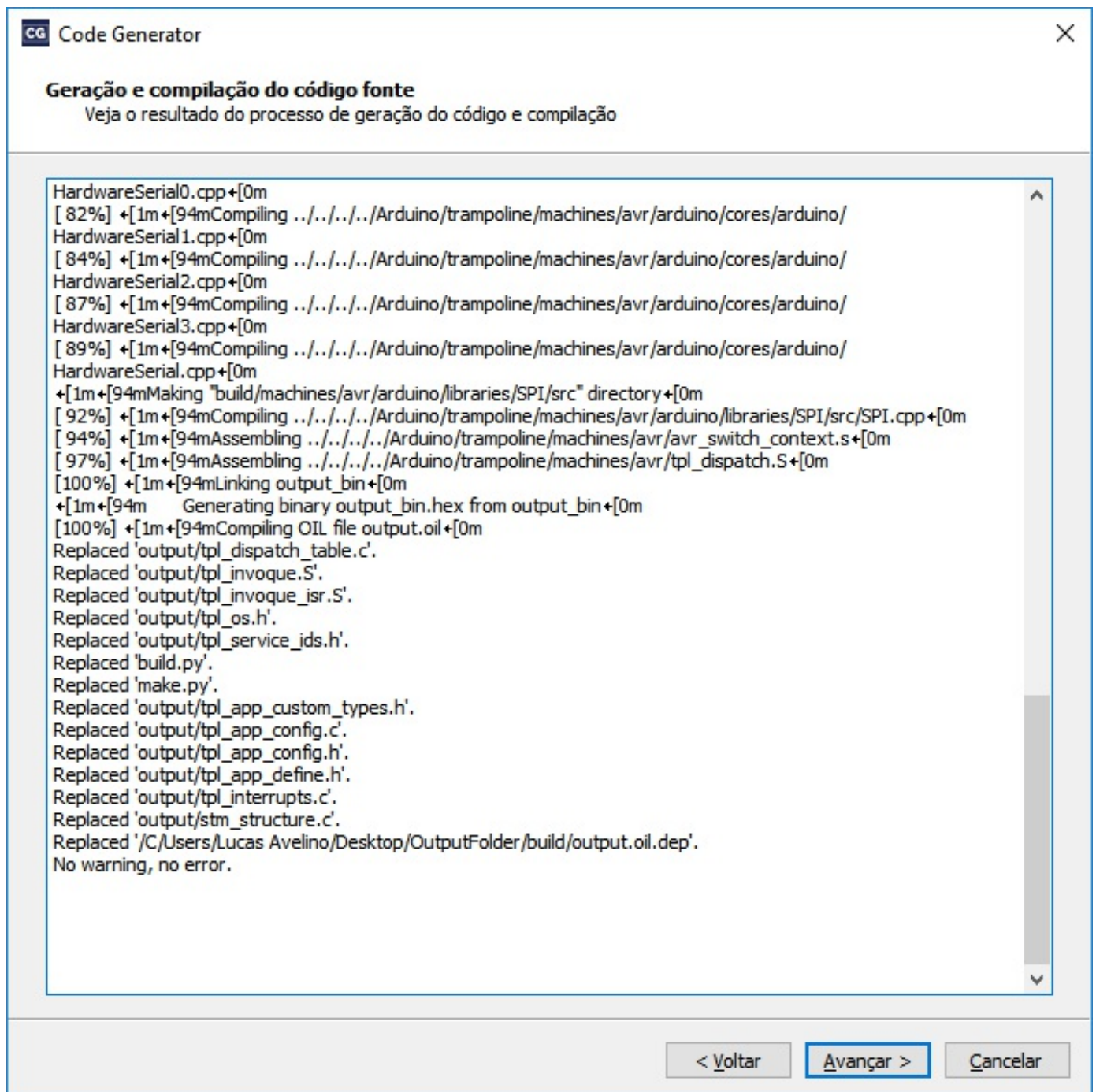
- A checked checkbox labeled 'A ECU envia mensagens pelo barramento CAN'. To its right is a text box labeled 'Endereço da ECU' containing the value '0'.
- An unchecked checkbox labeled 'A ECU faz uso da interface serial'.
- A text box labeled 'Tamanho da stack de can_send_task'.
- A text box labeled 'Tamanho da stack de pins_reader_task'.
- A text box labeled 'Ciclo de varredura de pins_reader_task (ms)'.
- A text box labeled 'Tamanho da stack de timer_task_SendEEC1'.

At the bottom right of the window, 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.

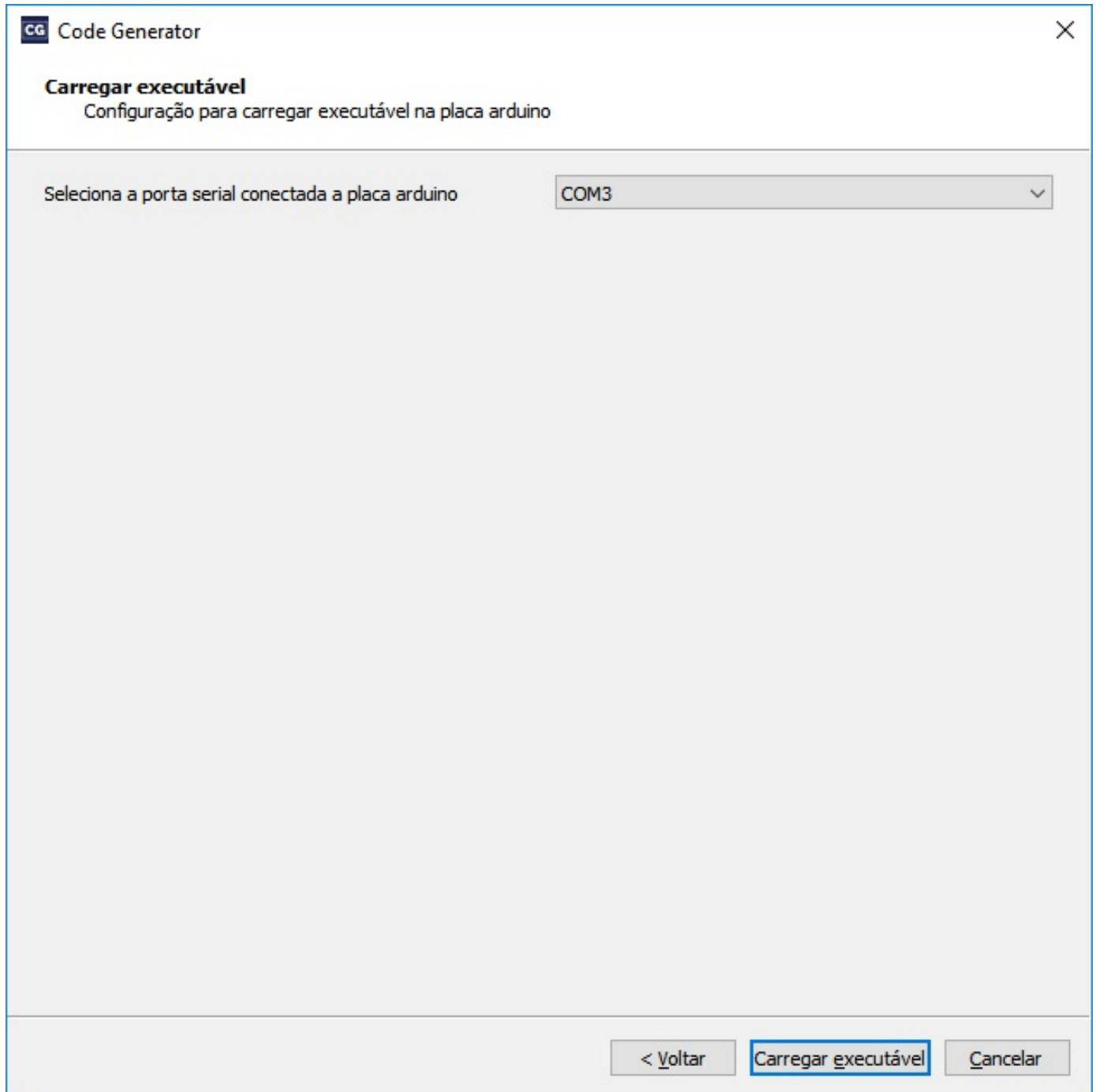


- Ao concluir o preenchimento dos campos o usuário deve clicar em **Avançar** para prosseguir.

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

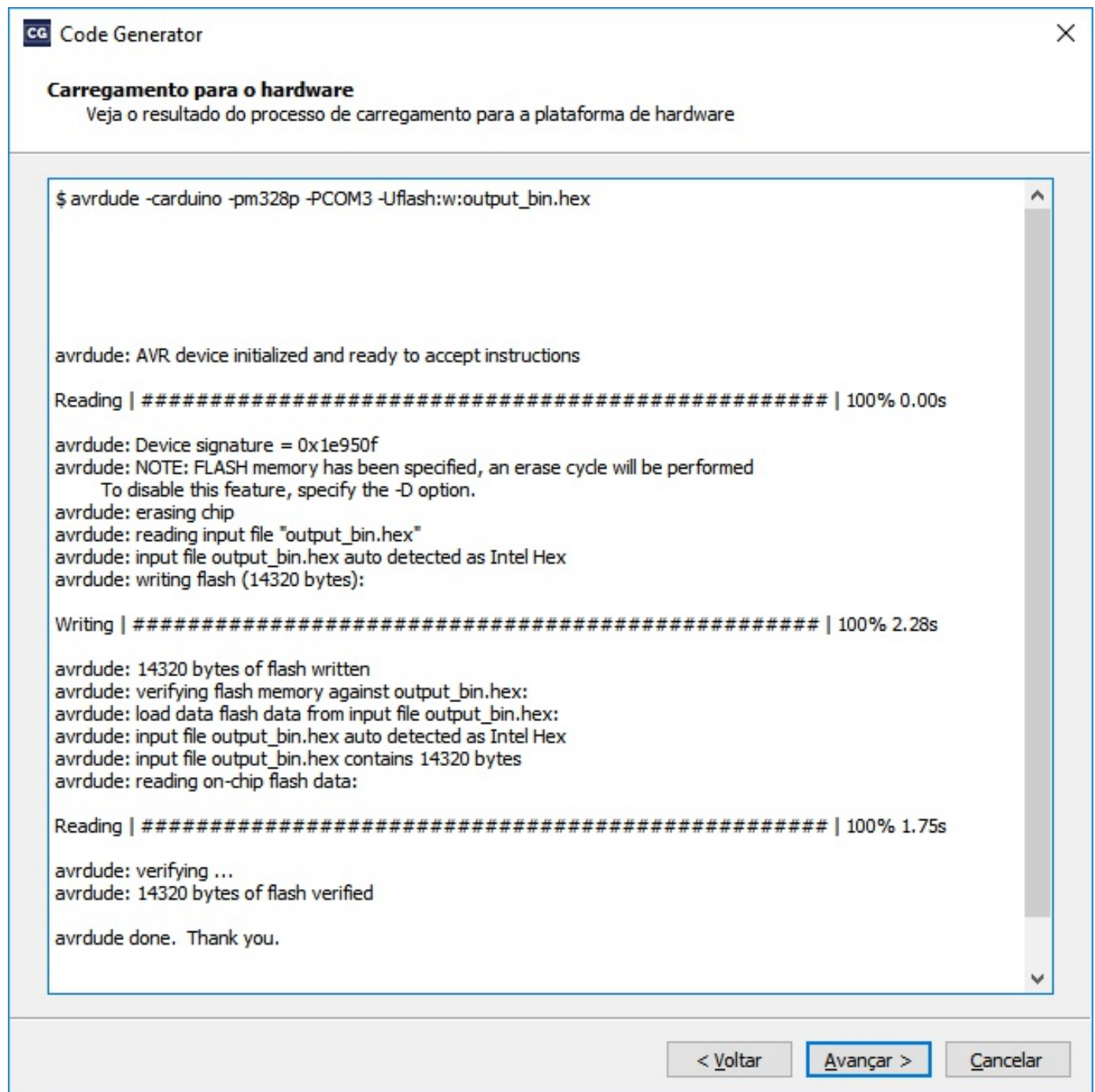
COM3

< 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.



```
Code Generator

Carregamento para o hardware
Veja o resultado do processo de carregamento para a plataforma de hardware

$ avrdude -carduino -pm328p -PCOM3 -Uflash:w:output_bin.hex

avrdude: AVR device initialized and ready to accept instructions

Reading | ##### | 100% 0.00s

avrdude: Device signature = 0x1e950f
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed
        To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "output_bin.hex"
avrdude: input file output_bin.hex auto detected as Intel Hex
avrdude: writing flash (14320 bytes):

Writing | ##### | 100% 2.28s

avrdude: 14320 bytes of flash written
avrdude: verifying flash memory against output_bin.hex:
avrdude: load data flash data from input file output_bin.hex:
avrdude: input file output_bin.hex auto detected as Intel Hex
avrdude: input file output_bin.hex contains 14320 bytes
avrdude: reading on-chip flash data:

Reading | ##### | 100% 1.75s

avrdude: verifying ...
avrdude: 14320 bytes of flash verified

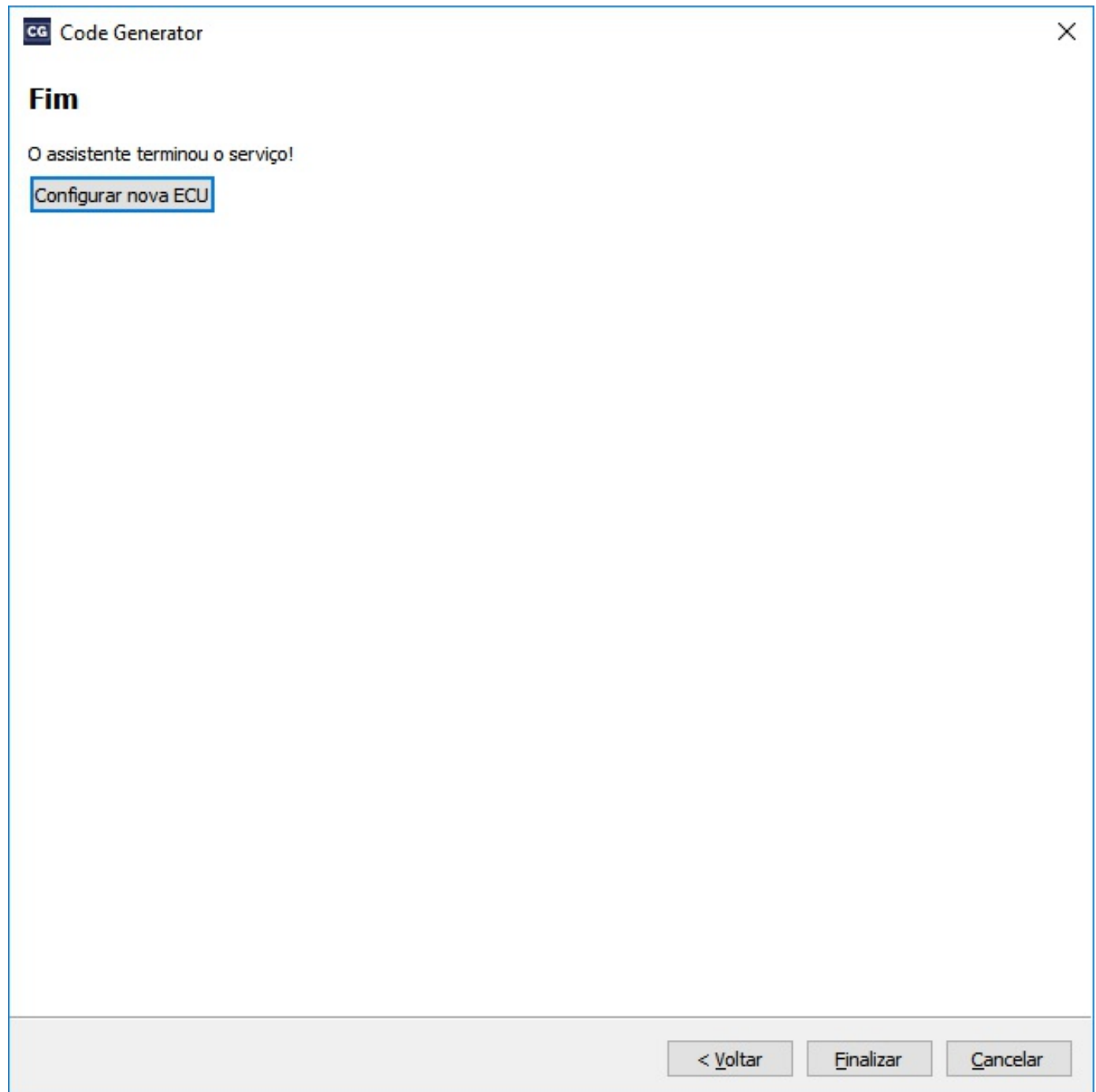
avrdude done. Thank you.

< Voltar  Avançar >  Cancelar
```

- 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.

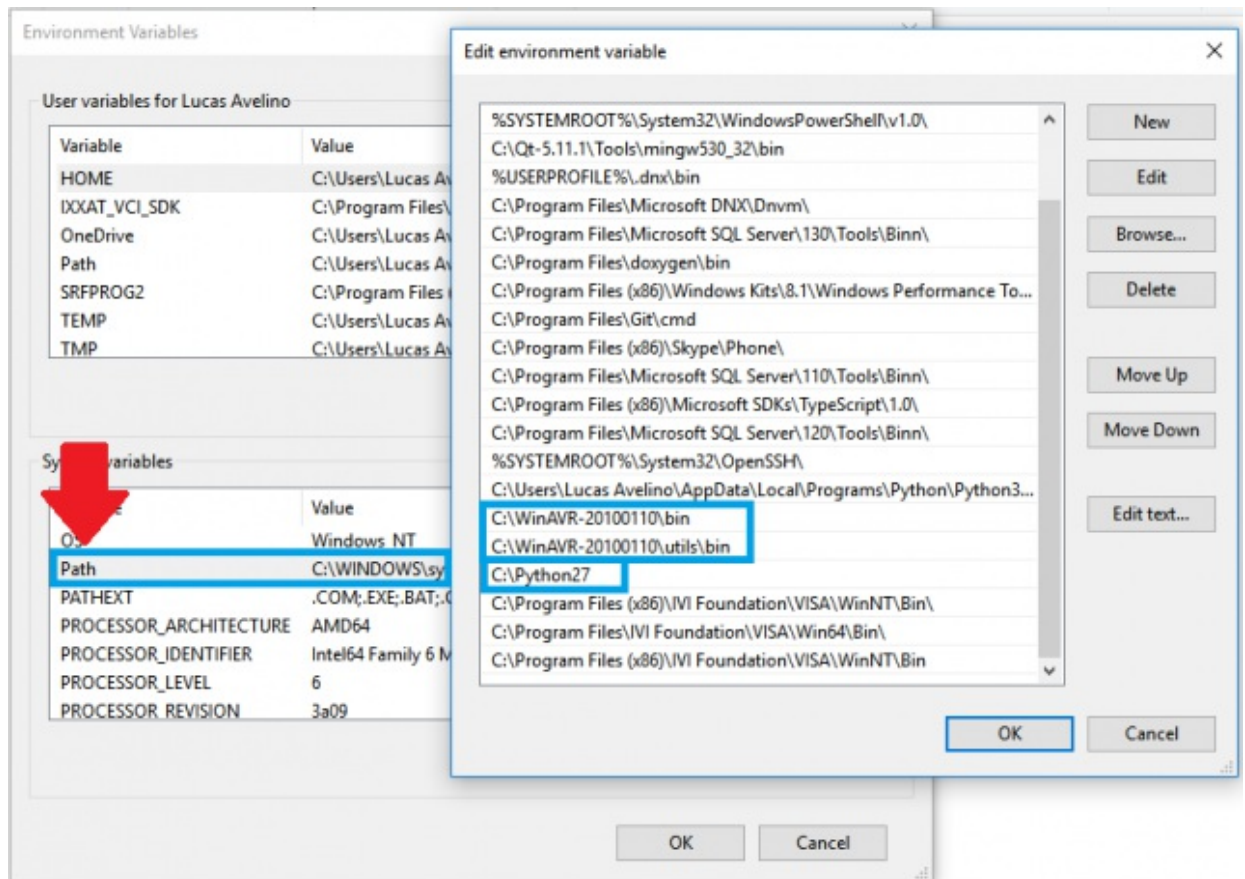


Dependências

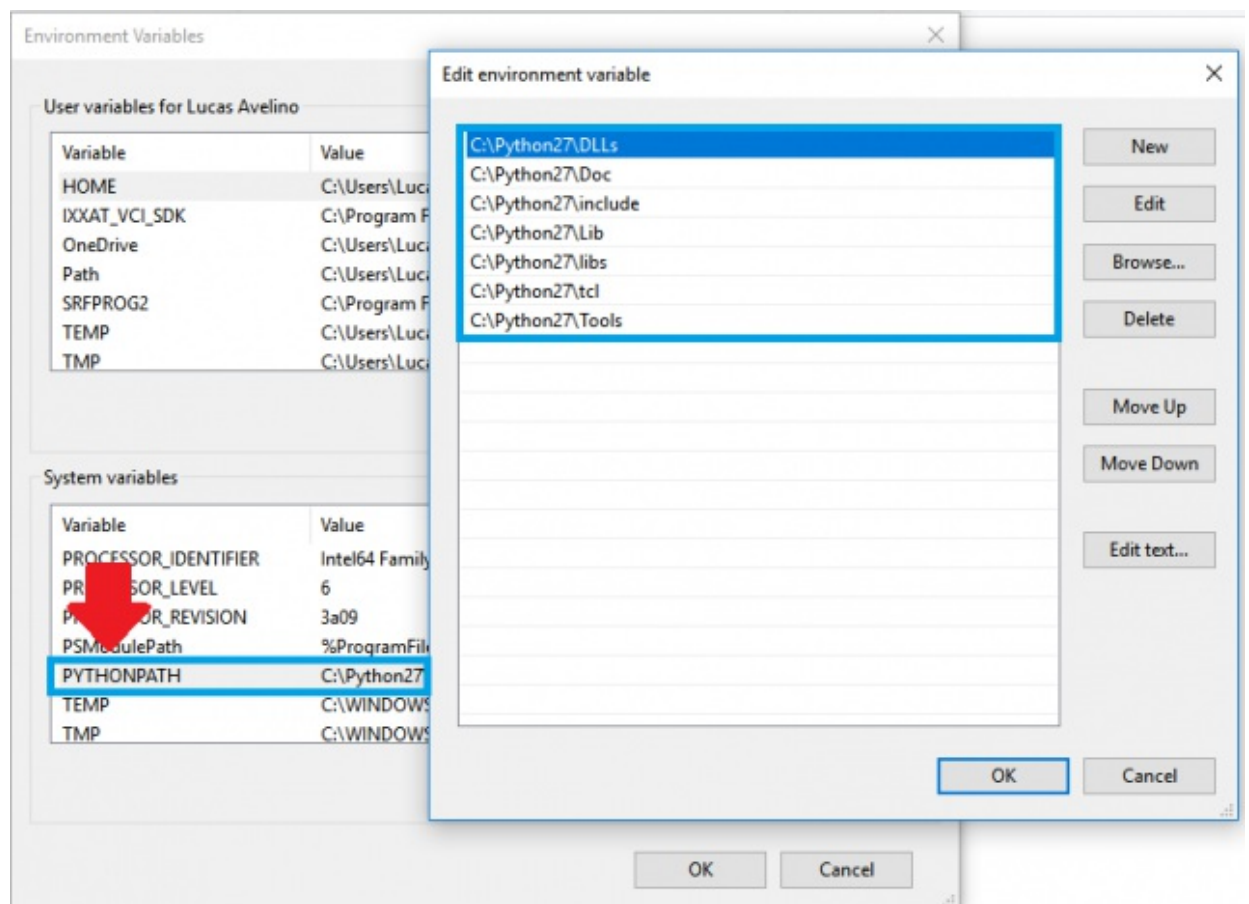
O assistente Code Generator faz uso do **TrampolineRTOS**, **Busmaster**,

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";  
};
```