

# **Construindo Um Projeto Com Processador Nios II**

## **Introdução**

Esse tutorial apresenta a construção de um projeto Nios II que contém componentes como processador, memória, interface input/output e interface serial JTag Uart. O Nios II é um processador softcore, ou seja, uma implementação de um processador descrito em linguagem de hardware, que pode ser customizado e sintetizado em um FPGA ou ASIC. A vantagem de ser softcore é a flexibilidade: fácil conexão com outros periféricos, alteração do conjunto de instruções e alteração das estruturas internas do processador [1].

Características do Nios II:

- Processador RISC com pipeline;
- 32 registradores de propósito geral;
- 3 formatos de instrução;
- Instruções de 32bits;
- Endereçamento de 32bits;
- Cache de dados e de instruções separados com tamanhos configuráveis;
- Memória on-chip;
- Branch Prediction;
- 32 interrupções prioritizáveis;
- On-chip hardware (Multiplicações, shift, rotate...);
- Memory Management Unit (MMU);
- Memory Protection Unit (MPU);
- Instruções customizadas em hardware;
- Debug utilizando JTAG;

Para desenvolver esse tutorial, assume-se que o usuário tem acesso a placa Cyclone IV da Macnica conectada ao computador que tem o software Quartus® II versão 16.1 e Software Monitor da Altera 16.1.

## Criando a plataforma do seu projeto

1. Abra o Quartus e inicie um novo projeto em *New Project Wizard*.

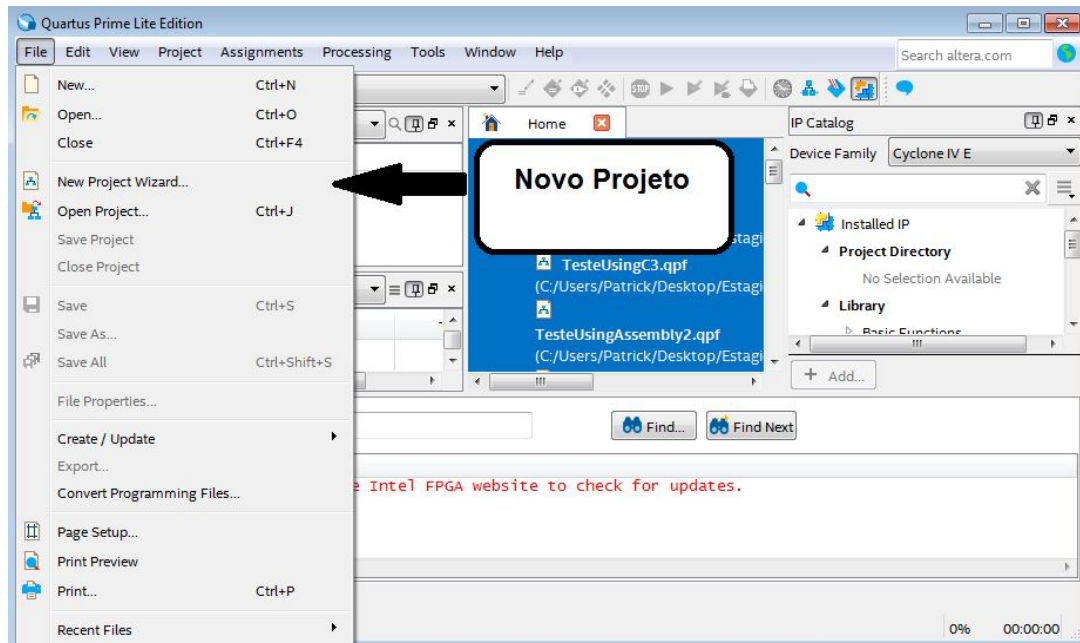


Figura 1: Novo projeto no quartus II

2. Defina o nome e diretório do projeto e avance a etapa utilizando o *Next*.

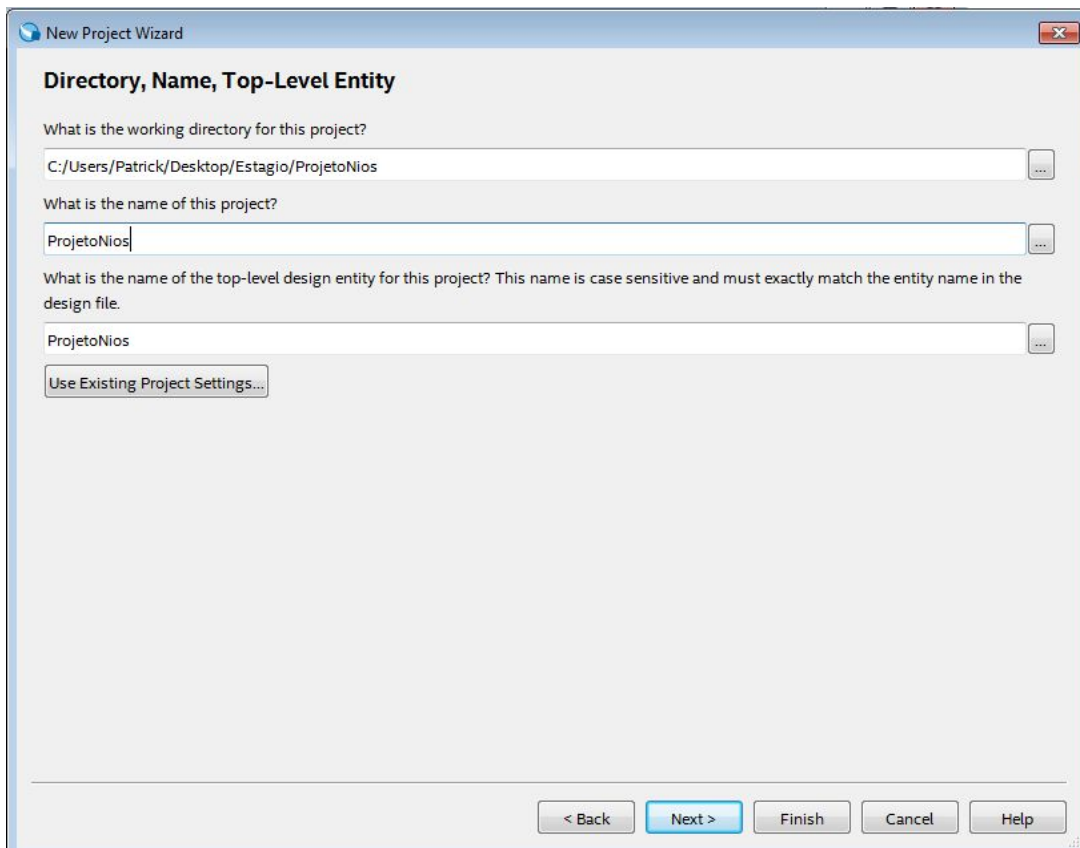


Figura 2: Definição de local e nome do projeto.

3. Selecione a placa cyclone IV E. Selecione também o número de série do FPGA **EP4CE30F23C7**. Avance tudo *Next* e utilize o *Finish* para finalizar a criação do projeto.

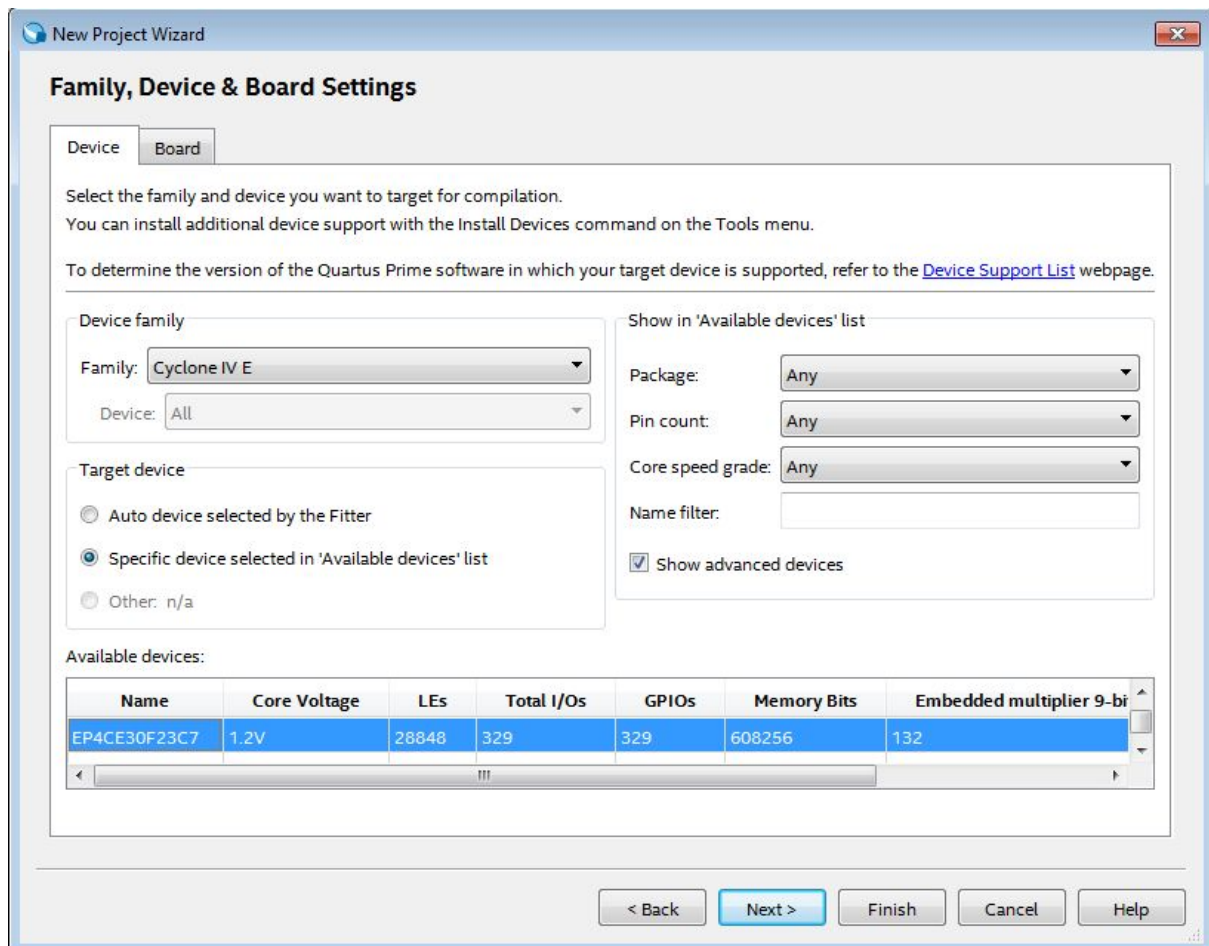


Figura 3: Definição da placa utilizada no projeto.

## Ferramenta Qsys da Altera

Depois de completar a criação do projeto, na janela principal do Quartus II selecione **Tools > Qsys**, conforme a figura 4.

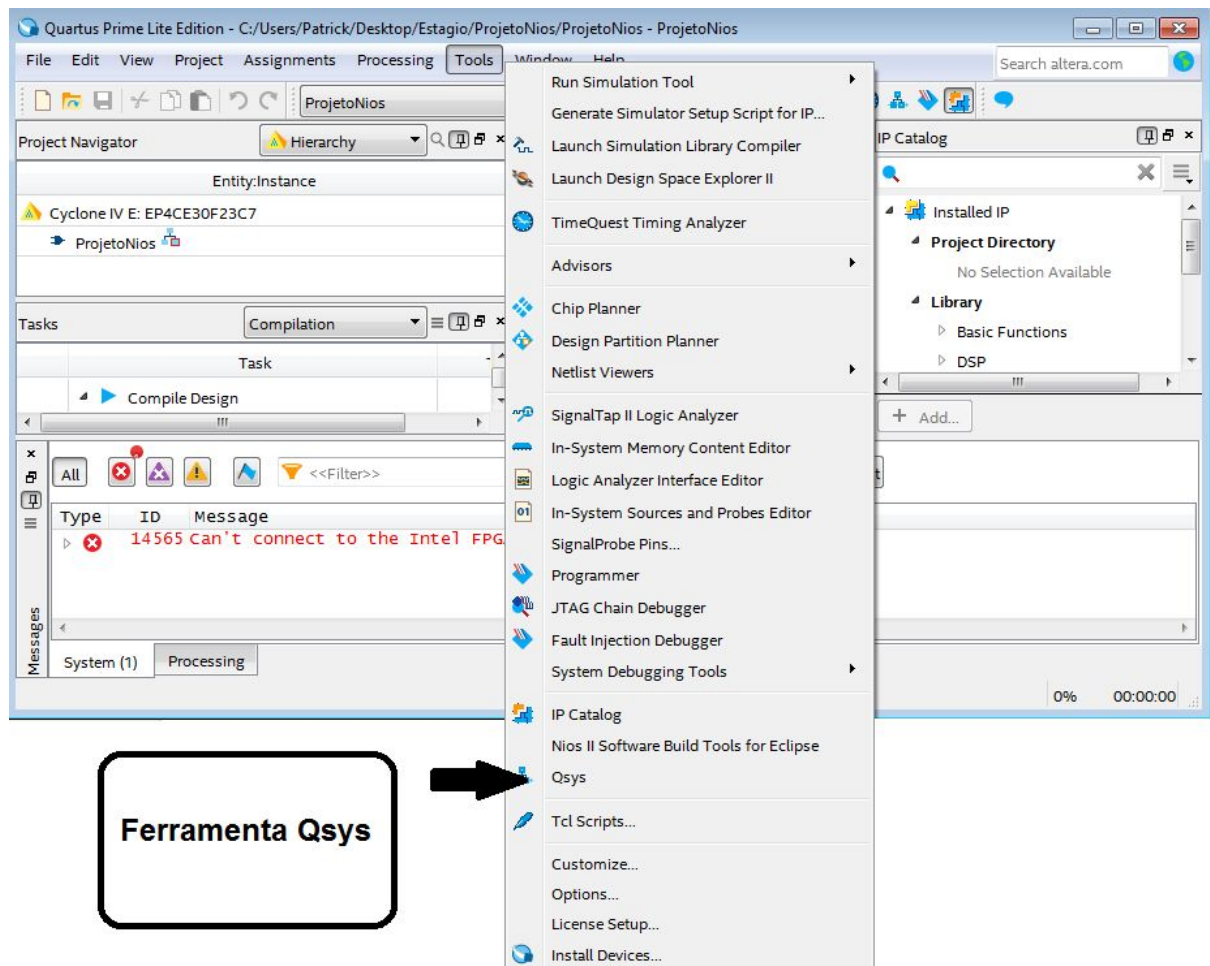


Figura 4: Como abrir o Qsys

O Qsys é usado para adicionar componentes ao projeto e configurá-los conforme os requisitos do projeto. Os componentes disponíveis são listados do lado esquerdo da janela do Qsys.

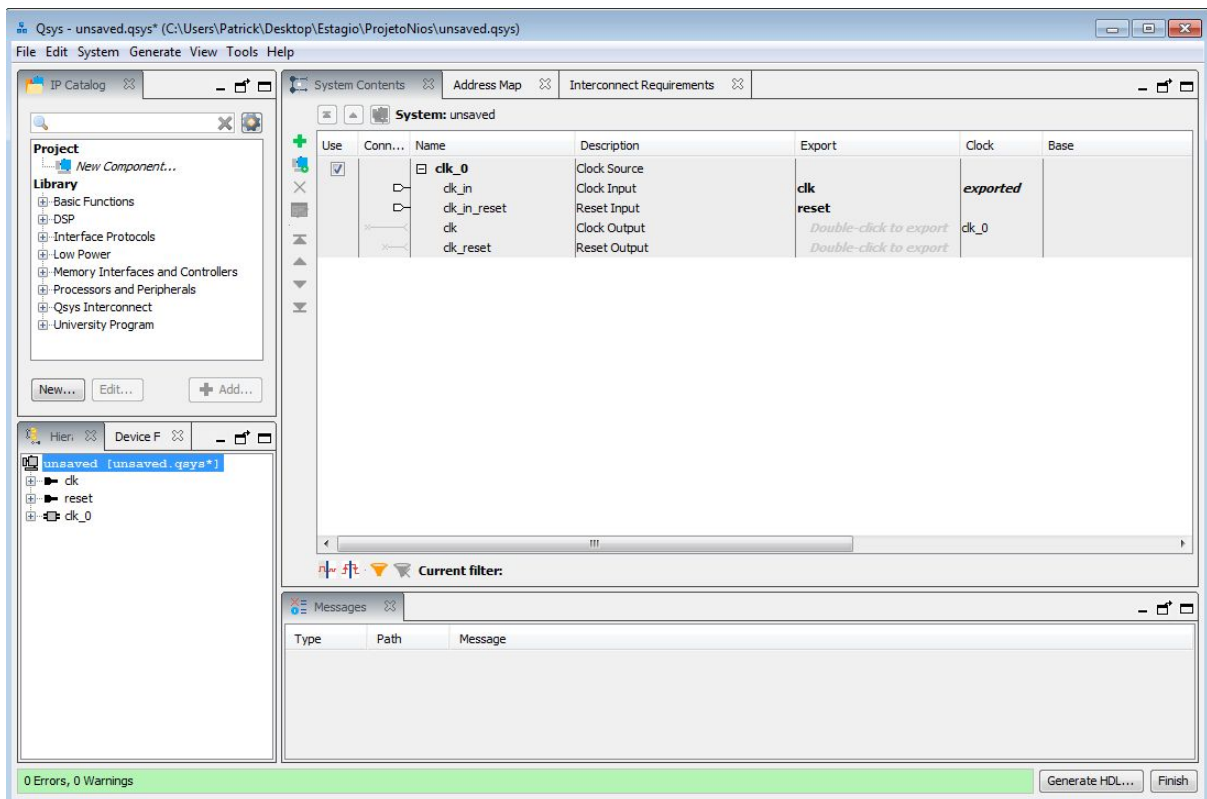


Figura 5: Ferramenta Qsys com sua barra lateral de componentes.

Segue o Passo a Passo para construção do projeto na ferramenta Qsys:

1. Especifique o processador como a seguir:
  - Sob o lado esquerdo da janela do Qsys abra **Processors and Peripherals**, selecione **Embedded Processor > Nios II Processor** e clique em **add**.

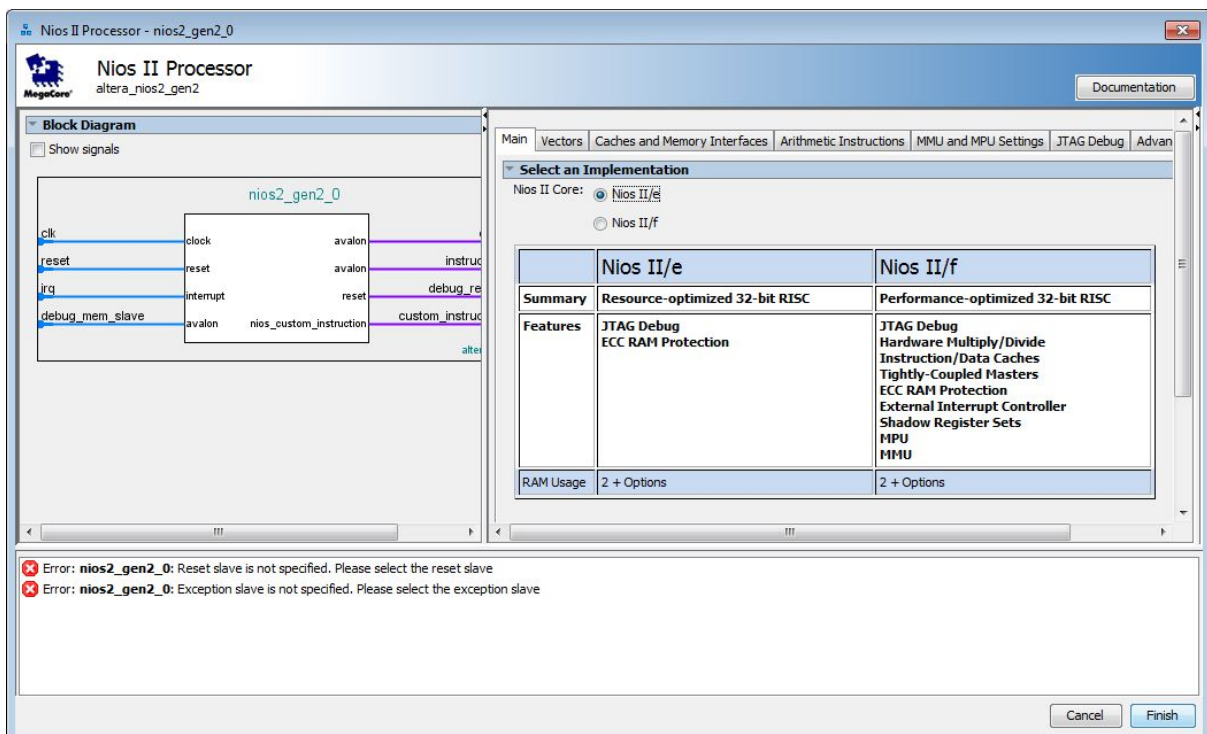


Figura 6: Configurações do processador Nios II.

- Escolha Nios II/e que representa a versão econômica do processador. Essa versão é disponibilizada para uso sem licença comercial, Figura 6. O processador Nios II têm pinos de entrada para reset e interrupção. Quando essas entradas são ativadas, o processador começa a executar as instruções em um endereço de memória específico chamado de vetor de reset e vetor de interrupção, respectivamente. Entretanto, nós não temos ainda um componente de memória inserido no nosso projeto. Como consequência, a ferramenta Qsys apresentará mensagens de erro correspondentes a esses vetores. Clique *Finish* para retornar para a janela principal que apresentará o processador Nios II escolhido, ilustrado na Figura 7.

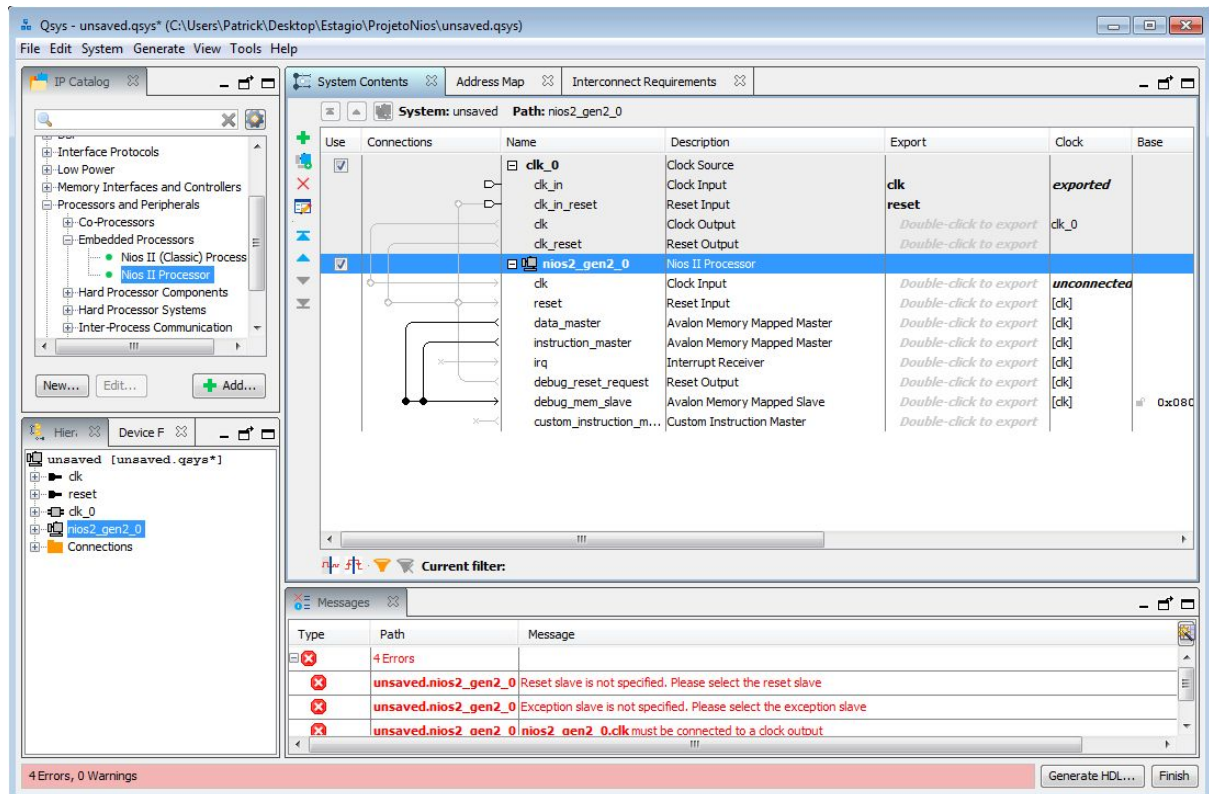


Figura 7: Nios II inserido nos componentes do projeto.

## 2. Para especificar a memória on-chip faça:

- Clique na categoria *Basic Functions*, e então expanda para selecionar **On-Chip Memory > On-Chip Memory (RAM or ROM)**, e clique em **Add**.
- Na janela de configuração da Memória On-Chip, exibida na Figura 8, garanta que o tamanho da Data seja de 32 bits e o total de tamanho da memória 4K bytes (4096 bytes).
- Não mude as outras configurações padrões.
- Clique em *Finish*, retornando assim a tela principal do Qsys como indicado na Figura 8.



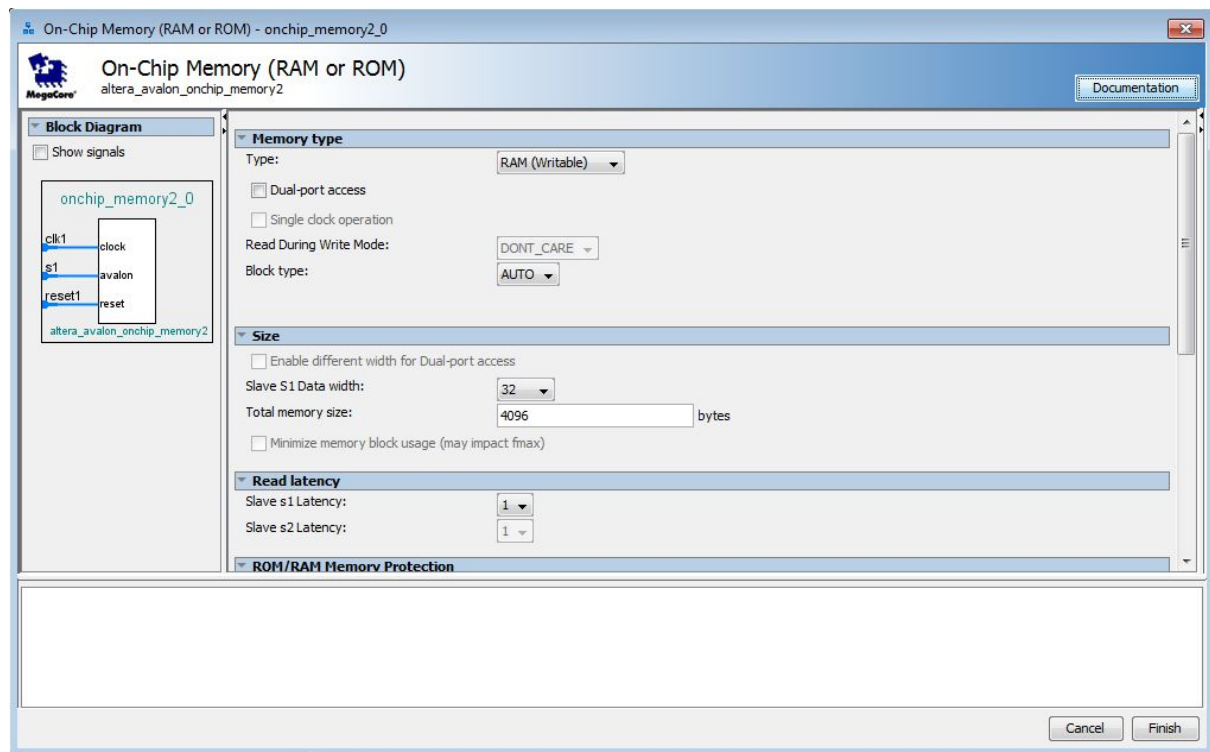


Figura 8: Configurações da memória on-chip.

3. Observe que quando o processador Nios II e a memória on-chip são inseridas no projeto, não existem conexões entre os componentes. Para especificar essas conexões, examine a área de conexões na Figura 9. As conexões são indicadas por círculos cheios e as outras possíveis conexões por círculos vazios, como indicado na Figura 10. Clicando sobre círculo vazio gera uma conexão. Clicando sobre círculo cheio remove a conexão. Faça as seguintes conexões:

- Entradas de clock do processador e da memória para saídas de clock do componente de clock.
- Entrada de reset do processador e da memória para ambas as saídas dos componentes reset e clock e a saída *Jtag\_debug\_module\_reset*.
- A entrada s1 da memória para ambos as saídas do processador *data\_master* e *instruction\_master*.

O resultado das conexões são mostrados na Figura 11.

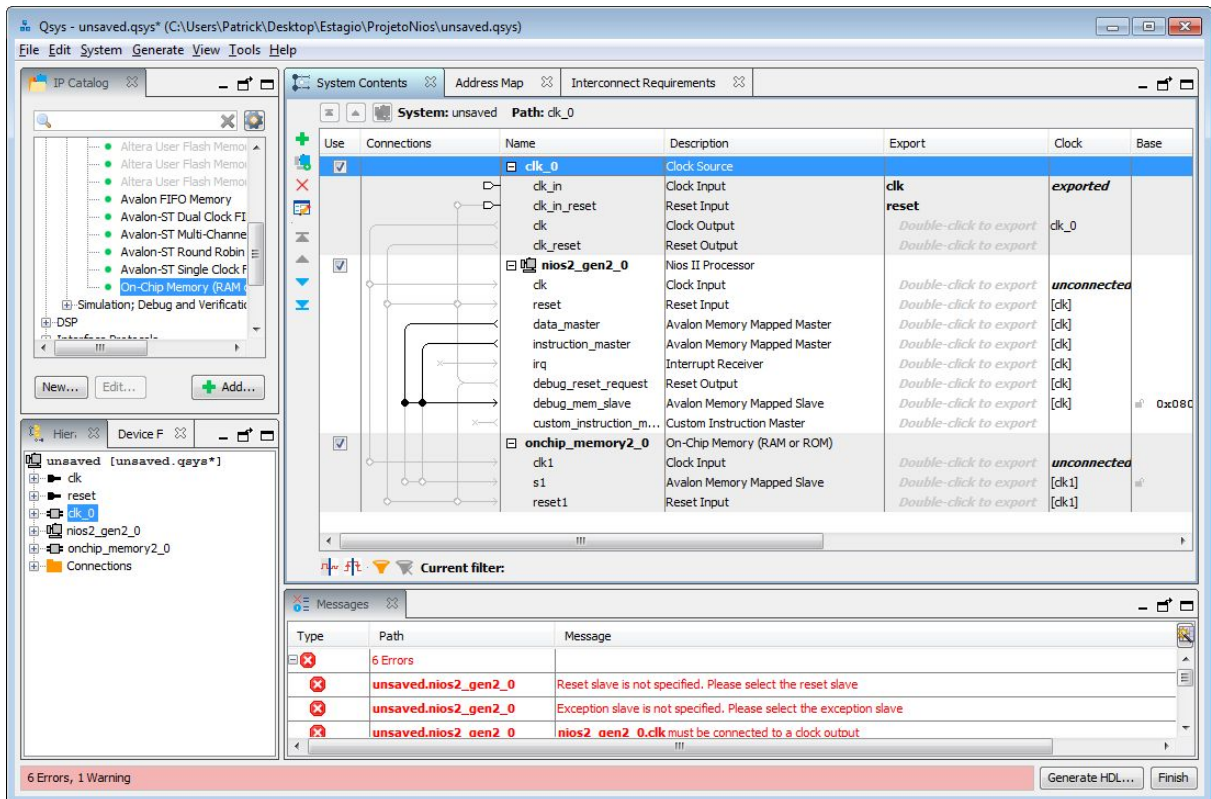


Figura 9: Painel de componentes com memória e processador.

Use	Connections	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source			
		clk_in	Clock Input	clk	exported	
		clk_in_reset	Reset Input	reset		
		clk	Clock Output	Double-click to export	clk_0	
		clk_reset	Reset Output	Double-click to export		
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor			
		clk	Clock Input	Double-click to export	unconnected	
		reset	Reset Input	Double-click to export	[clk]	
		data_master	Avalon Memory Mapped Master	Double-click to export	[clk]	
		instruction_master	Avalon Memory Mapped Master	Double-click to export	[clk]	
		irq	Interrupt Receiver	Double-click to export	[clk]	
		debug_reset_request	Reset Output	Double-click to export	[clk]	
		debug_mem_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x080
		custom_instruction_m...	Custom Instruction Master	Double-click to export		
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)			
		clk1	Clock Input	Double-click to export	unconnected	
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk1]	
		reset1	Reset Input	Double-click to export	[clk1]	



Figura 10: Componentes sem conexões

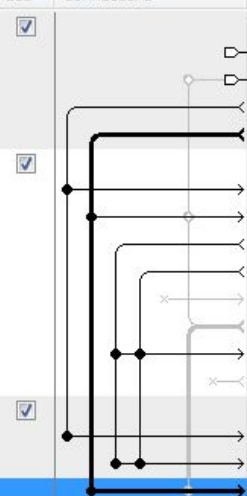
Use	Connections	Name	Description	Export	Clock	Base
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source			
		clk_in	Clock Input	<b>clk</b>	<b>exported</b>	
		clk_in_reset	Reset Input	<b>reset</b>		
		clk	Clock Output	<i>Double-click to export</i>	clk_0	
		clk_reset	Reset Output	<i>Double-click to export</i>		
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor			
		clk	Clock Input	<i>Double-click to export</i>	clk_0	
		reset	Reset Input	<i>Double-click to export</i>	[clk]	
		data_master	Avalon Memory Mapped Master	<i>Double-click to export</i>	[clk]	
		instruction_master	Avalon Memory Mapped Master	<i>Double-click to export</i>	[clk]	
		irq	Interrupt Receiver	<i>Double-click to export</i>	[clk]	
		debug_reset_request	Reset Output	<i>Double-click to export</i>	[clk]	
		debug_mem_slave	Avalon Memory Mapped Slave	<i>Double-click to export</i>	[clk]	0x080
		custom_instruction_m...	Custom Instruction Master	<i>Double-click to export</i>		
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)			
		clk1	Clock Input	<i>Double-click to export</i>	clk_0	
		s1	Avalon Memory Mapped Slave	<i>Double-click to export</i>	[clk1]	0x000
		reset1	Reset Input	<i>Double-click to export</i>	[clk1]	

Figura 11: Componentes com conexões.

4. Para especificar a interface de entrada *parallel I/O* faça:

- Selecione **Processors and Peripherals > Peripherals > PIO (Parallel I/O)** e clique **Add** para abrir as configurações PIO na Figura 12.
- Especifique o tamanho da porta para 8 bits e escolha a direção para *input*, como exibido na figura.
- Clique *Finish*.

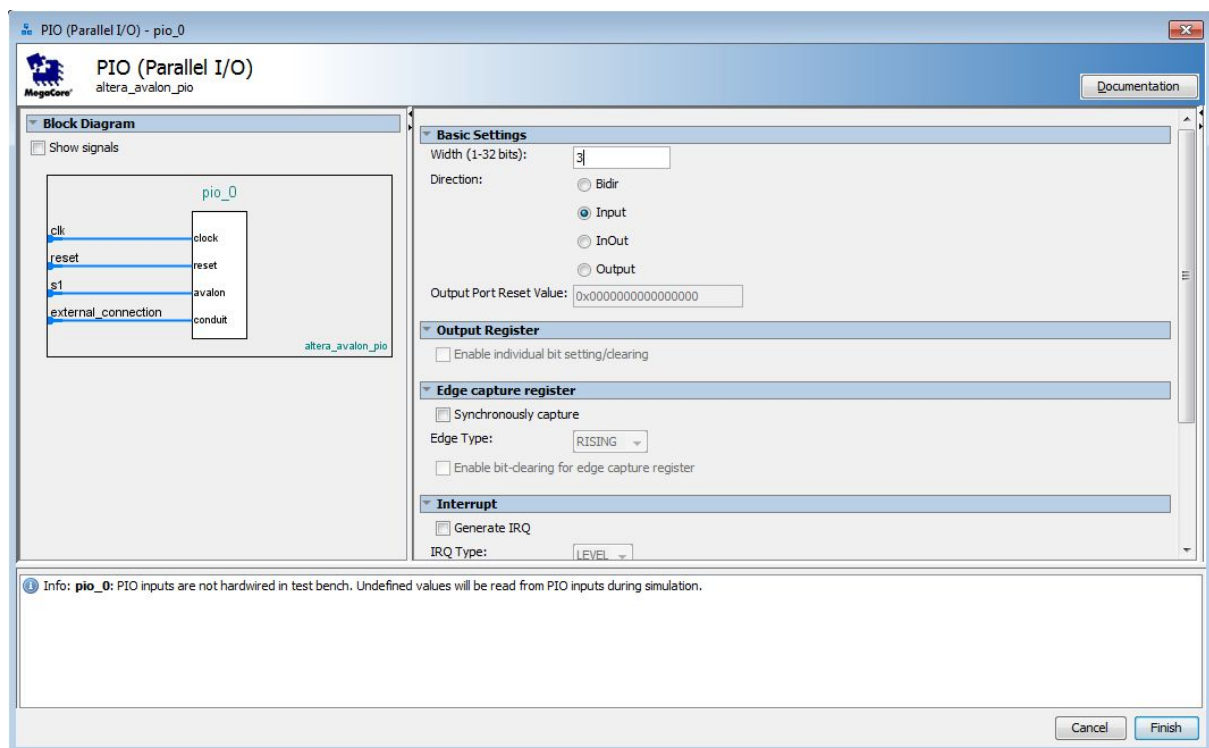


Figura 12: Configurações de PIO.

4. Seguindo a mesma ideia, especifique a interface de saída *parallel I/O*:

- Selecione **Peripherals > Microcontroller Peripherals > PIO (Parallel I/O)** e clique **Add** para aparecer a janela de configuração PIO novamente.

- Especifique o tamanho da porta para 8 bits e escolha a direção da porta para *output*.
- Clique em *Finish* para retornar para o Qsys.

5. Especifique as conexões necessárias para os dois PIOs:

- Entrada do clock do PIO para saída do componente de clock.
- Entrada de reset do PIO para saída reset do componente de clock e a saída do *jtag\_debug\_module\_reset*.
- A entrada s1 do PIO para a saída do processador *data\_master*.

O projeto resultante é exibido na Figura 13.

Use	Connections	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source				
		clk_in	Clock Input	clk	exported		
		clk_in_reset	Reset Input	reset			
		clk	Clock Output	Double-click to export	clk_0		
		clk_reset	Reset Output	Double-click to export			
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor				
		clk	Clock Input	Double-click to export	clk_0		
		reset	Reset Input	Double-click to export	[clk]		
		data_master	Avalon Memory Mapped Master	Double-click to export	[clk]		
		instruction_master	Avalon Memory Mapped Master	Double-click to export	[clk]		
		irq	Interrupt Receiver	Double-click to export	[clk]		IRQ 0
		debug_reset_request	Reset Output	Double-click to export	[clk]		IRQ 31
		debug_mem_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0800	0x0fff
		custom_instruction_m...	Custom Instruction Master	Double-click to export			
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)				
		clk1	Clock Input	Double-click to export	clk_0		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk1]		
		reset1	Reset Input	Double-click to export	[clk1]	0x0000	0x0fff
<input checked="" type="checkbox"/>		<b>pio_0</b>	PIO (Parallel I/O)				
		clk	Clock Input	Double-click to export	clk_0		
		reset	Reset Input	Double-click to export	[clk]		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0000	0x000f
		external_connection	Conduit	Double-click to export			
<input checked="" type="checkbox"/>		<b>pio_1</b>	PIO (Parallel I/O)				
		clk	Clock Input	Double-click to export	clk_0		
		reset	Reset Input	Double-click to export	[clk]		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0000	0x000f
		external_connection	Conduit	Double-click to export			

Figura 13: Resultado das conexões de PIO com o projeto.

6. Em nosso projeto, desejamos que haja uma comunicação entre o computador e o sistema Nios II. Isso pode ser associado instanciando a componente JTAG UART.

- Selecione **Interface Protocols > Serial > JTAG UART** e clique **Add** para abrir as configurações JTAG UART. Conforme a Figura 14.

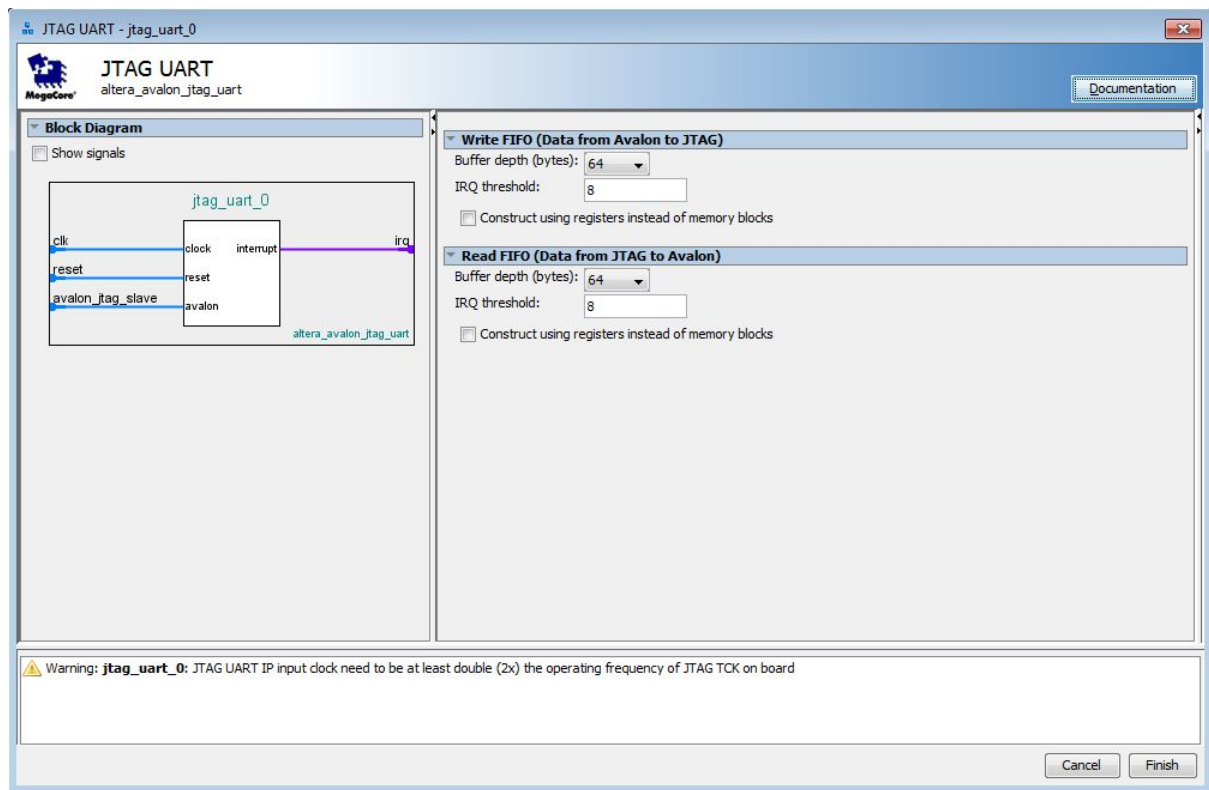


Figura 14: Configurações UART.

- Não mude as configurações padrões.
- Clique em *Finish* para retornar para o Qsys.

Conecte a requisição de interrupção (IRQ) do JTAG UART para o processador Nios II, conforme a Figura 15. Uma vez que a conexão esteja feita, uma caixa com o número 0 aparecerá sobre a conexão. O processador Nios II tem 32 portas de interrupções de 0 a 31, e o número nessa caixa representa qual porta será utilizada para essa IRQ. Clique sobre a caixa e mude para usar a porta 5. Tenha certeza que a porta IRQ da JTAG UART está automaticamente conectada para a porta `d_irq` do processador Nios II.

Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source	<b>clk</b> <b>reset</b> <i>Double-click to export</i> <i>Double-click to export</i>	<i>exported</i> <b>clk_0</b>			
		clk_in	Clock Input					
		clk_in_reset	Reset Input					
		clk	Clock Output					
		clk_reset	Reset Output					
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	<b>clk_0</b> [clk] [clk] [clk] [clk] [clk] [clk]			
		clk	Clock Input					
		reset	Reset Input					
		data_master	Avalon Memory Mapped Master					
		instruction_master	Avalon Memory Mapped Master					
		irq	Interrupt Receiver					IRQ 0
		debug_reset_request	Reset Output					
		debug_mem_slave	Avalon Memory Mapped Slave					
		custom_instruction_master	Custom Instruction Master					0x0800
					0x0fff			
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	<b>clk_0</b> [clk1] [clk1]			
		clk1	Clock Input					
		s1	Avalon Memory Mapped Slave					0x0000
		reset1	Reset Input					0x0fff
<input checked="" type="checkbox"/>		<b>pio_0</b>	PIO (Parallel I/O)	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	<b>clk_0</b> [clk] [clk]			
		clk	Clock Input					
		reset	Reset Input					
		s1	Avalon Memory Mapped Slave					0x0000
		external_connection	Conduit			0x000f		
<input checked="" type="checkbox"/>		<b>pio_1</b>	PIO (Parallel I/O)	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	<b>clk_0</b> [clk] [clk]			
		clk	Clock Input					
		reset	Reset Input					
		s1	Avalon Memory Mapped Slave					0x0000
		external_connection	Conduit			0x000f		
<input checked="" type="checkbox"/>		<b>jtag_uart_0</b>	JTAG UART	<i>Double-click to export</i> <i>Double-click to export</i> <i>Double-click to export</i>	<b>clk_0</b> [clk] [clk]			
		clk	Clock Input					
		reset	Reset Input					
		avalon_jtag_slave	Avalon Memory Mapped Slave					0x0000
		irq	Interrupt Sender	<i>Double-click to export</i>	[clk]		0x0007	IRQ 31

Figura 15: Conexão de interrupção representada no lado direito da Figura.

7. Note que a ferramenta Qsys automaticamente escolhe nomes para vários componentes. Os nomes não são necessariamente descritivos o suficiente para ser facilmente associados aos designs, mas eles podem ser mudados. Na figura 16, nós usamos os nomes Switches e Leds para as *parallel I/O*. Clique com o botão direito sobre o nome pio\_0 e selecione Rename. Mude o nome para Switches. Similarmente, mude pio\_1 para Leds. A figura 16 mostra o sistema com os nomes mudados.

Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source					
		clk_in	Clock Input	clk	exported			
		clk_in_reset	Reset Input	reset				
		clk	Clock Output	Double-click to export	clk_0			
		clk_reset	Reset Output	Double-click to export				
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		data_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
		instruction_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
		irq	Interrupt Receiver	Double-click to export	[clk]		IRQ 0	IRQ 31
		debug_reset_request	Reset Output	Double-click to export	[clk]			
		debug_mem_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0800	0x0fff	
		custom_instruction_m...	Custom Instruction Master	Double-click to export				
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)					
		clk1	Clock Input	Double-click to export	clk_0			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk1]	0x0000	0x0fff	
		reset1	Reset Input	Double-click to export	[clk1]			
<input checked="" type="checkbox"/>		<b>switches</b>	PIO (Parallel I/O)					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0000	0x000f	
		external_connection	Conduit	Double-click to export				
<input checked="" type="checkbox"/>		<b>leds</b>	PIO (Parallel I/O)					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0000	0x000f	
		external_connection	Conduit	Double-click to export				
<input checked="" type="checkbox"/>		<b>jtag_uart_0</b>	JTAG UART					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		avalon_jtag_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0000	0x0007	
		irq	Interrupt Sender	Double-click to export	[clk]			

Figura 16: Módulos Renomeados.

8. Observe que a base e o fim dos endereços de vários componentes no projeto estão se sobrepondo. Esses endereços podem ser redefinidos pelo usuário e, também, podem ser automaticamente redefinidos pela ferramenta Qsys. Nós escolhemos a última possibilidade. Entretanto, queremos ter certeza que o endereço base de memória on-chip inicie com zero. Dê clique duplo no endereço base da memória on-chip na janela Qsys e entre com o endereço 0x00000000. Então, feche o cadeado desse endereço clicando sobre ele. Agora, permita que o Qsys associe os endereços correspondentes selecionando **System > Assign Base Addresses** (No topo da janela), que produzirá um conjunto de endereços similar à exibida na Figura 17.

Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source					
		clk_in	Clock Input	clk	exported			
		clk_in_reset	Reset Input	reset				
		clk	Clock Output	Double-click to export	clk_0			
		clk_reset	Reset Output	Double-click to export				
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		data_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
		instruction_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
		irq	Interrupt Receiver	Double-click to export	[clk]		IRQ 0	IRQ 31
		debug_reset_request	Reset Output	Double-click to export	[clk]			
		debug_mem_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x1800	0x1fff	
		custom_instruction_m...	Custom Instruction Master	Double-click to export				
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)					
		clk1	Clock Input	Double-click to export	clk_0			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk1]	0x0000	0x0fff	
		reset1	Reset Input	Double-click to export	[clk1]			
<input checked="" type="checkbox"/>		<b>switches</b>	PIO (Parallel I/O)					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x2030	0x203f	
		external_connection	Conduit	Double-click to export				
<input checked="" type="checkbox"/>		<b>leds</b>	PIO (Parallel I/O)					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x2020	0x202f	
		external_connection	Conduit	Double-click to export				
<input checked="" type="checkbox"/>		<b>jtag_uart_0</b>	JTAG UART					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		avalon_jtag_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x2048	0x204f	
		irq	Interrupt Sender	Double-click to export	[clk]			

Figura 17: Endereços dos módulos redefinidos com a memória iniciando em 0.



O comportamento do processador Nios II quando é resetado é definido por um vetor de reset. Ele é localizado no dispositivo de memória do qual o processador obtém a próxima instrução que ele deve executar quando uma interrupção surge. Para especificar esses dois parâmetros, faça o seguinte:

- Clique com o direito sobre o componente processador nios2\_gen2\_0 na janela e então selecione editar para abrir a janela na Figura 18.
- Selecione *onchip\_memory.s1* para ser o dispositivo de memória para ambos os vetores de reset e interrupção (*exception*), conforme Figura 18.
- Não mude as configurações padrões para offsets.
- Observe que as mensagens de erro agora desapareceram.
- Clique *Finish* para retornar ao Qsys.

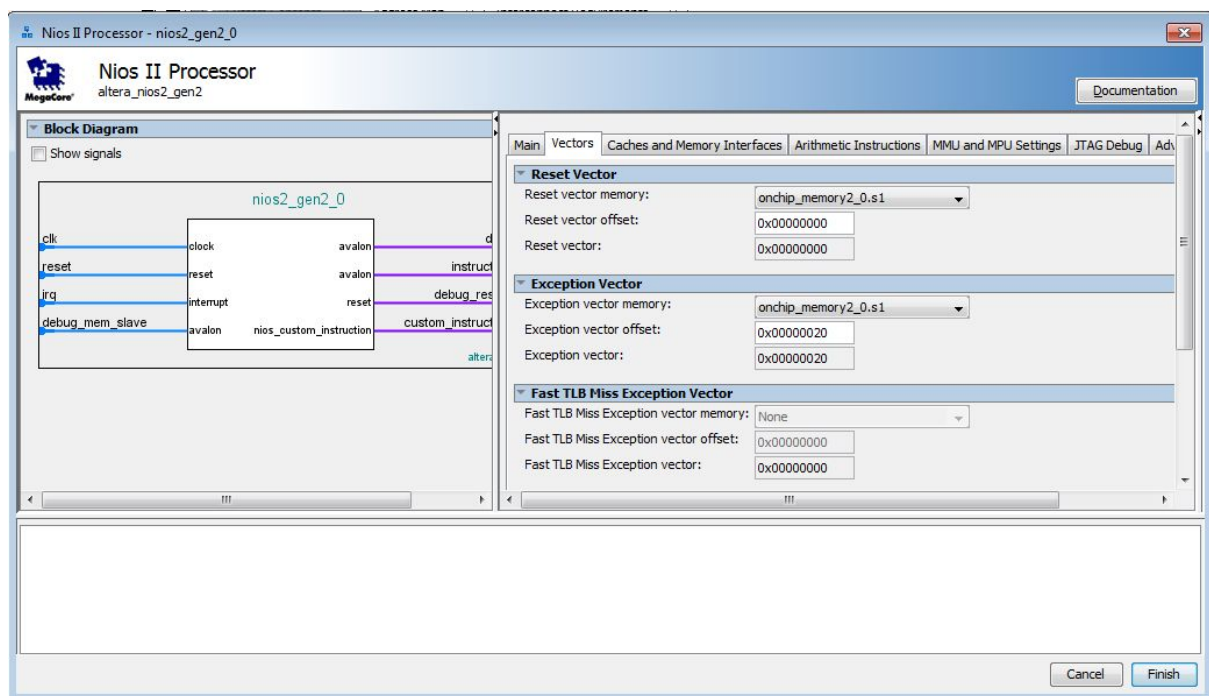


Figura 19: Mudança dos vetores para a memória.

9. Agora que nós temos especificado todas as conexões dentro do nosso projeto Nios. É necessário também especificar as conexões para componentes externos, switches e Leds no nosso caso. Para isso, clique duas vezes em *Double-click to export* para *external\_connection* do switches PIO, e mude para o nome switches. Similarmente, mude conexão externa dos leds para o nome leds. Essa mudança completa a especificação para o projeto do Nios, conforme Figura 20.



Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		<b>clk_0</b>	Clock Source		<b>exported</b>			
		clk_in	Clock Input	clk				
		clk_in_reset	Reset Input	reset				
		clk	Clock Output	Double-click to export	clk_0			
		clk_reset	Reset Output	Double-click to export				
<input checked="" type="checkbox"/>		<b>nios2_gen2_0</b>	Nios II Processor					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		data_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
		instruction_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
		irq	Interrupt Receiver	Double-click to export	[clk]		IRQ 0	IRQ 31
		debug_reset_request	Reset Output	Double-click to export	[clk]			
		debug_mem_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	# 0x1800	0x1fff	
		custom_instruction_m...	Custom Instruction Master	Double-click to export				
<input checked="" type="checkbox"/>		<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)					
		clk1	Clock Input	Double-click to export	clk_0			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk1]	# 0x0000	0x0fff	
		reset1	Reset Input	Double-click to export	[clk1]			
<input checked="" type="checkbox"/>		<b>switches</b>	PIO (Parallel I/O)					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	# 0x2030	0x203f	
		external_connection	Conduit	switches				
<input checked="" type="checkbox"/>		<b>leds</b>	PIO (Parallel I/O)					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	# 0x2020	0x202f	
		external_connection	Conduit	leds				
<input checked="" type="checkbox"/>		<b>jtag_uart_0</b>	JTAG UART					
		clk	Clock Input	Double-click to export	clk_0			
		reset	Reset Input	Double-click to export	[clk]			
		avalon_jtag_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	# 0x2048	0x204f	
		irq	Interrupt Sender	Double-click to export	[clk]			

Figura 20: Mudança dos nomes de exportação das entradas e saídas.

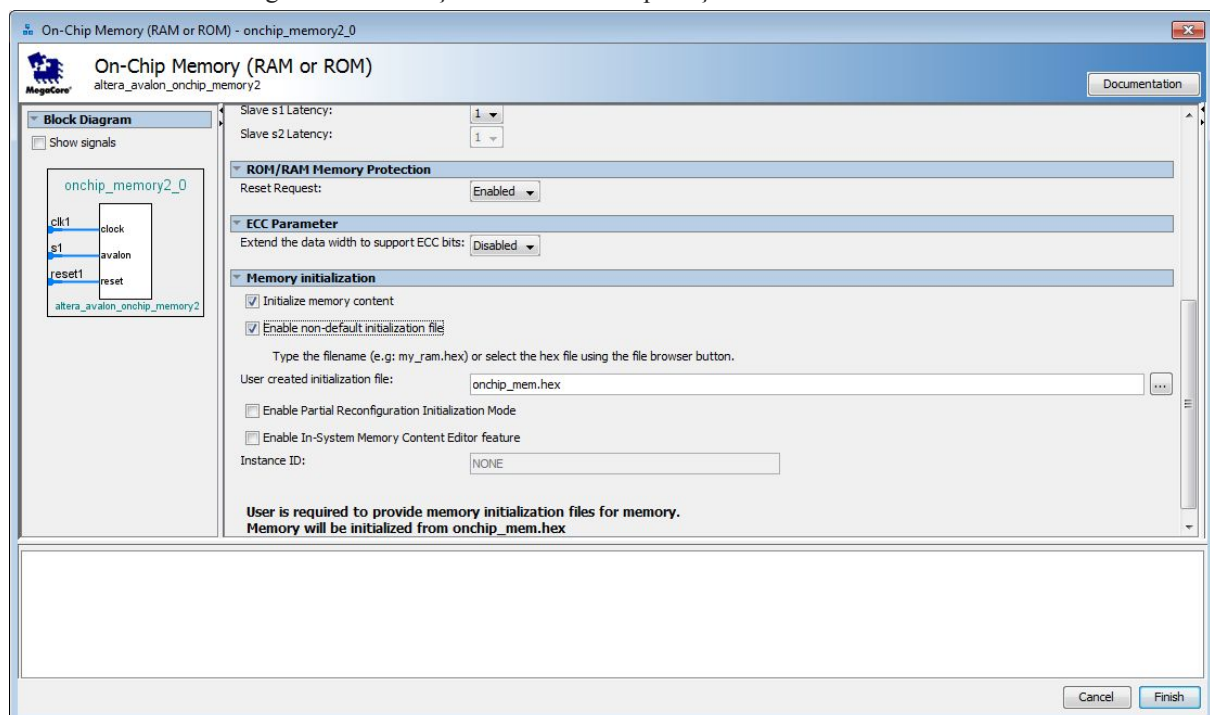


Figura 21: Mudança na memória on-chip para por memória inicializável.

9. Selecione o *Edit* na memória on-chip e marque o check-box **Enable non-default initialization file**. Dentro do diretório do projeto construa um arquivo *.hex*. Selecione esse arquivo *.hex* na opção **User created initialization file** na memória on-chip, Figura 21.

10. Depois de especificar todos os componentes necessários para a implementação do projeto, podemos agora gerar o HDL. Salve o projeto Qsys especificado; Nós usaremos o nome de ProjetoNiosQsys. Então, selecione **Generate > Generate HDL**, conforme a Figura 22.

Selecione *None* para as opções de **Simulation > Create simulation model** e **Testbench System > Create testbench Qsys system**, porque nesse tutorial nós não trabalhamos com simulação de hardware. Clique em *Generate HDL* no botão da janela. Quando completar a geração, aparecerá a mensagem “*Generate Completed*”. Saia da ferramenta Qsys para retornar para janela principal do Quartus II.

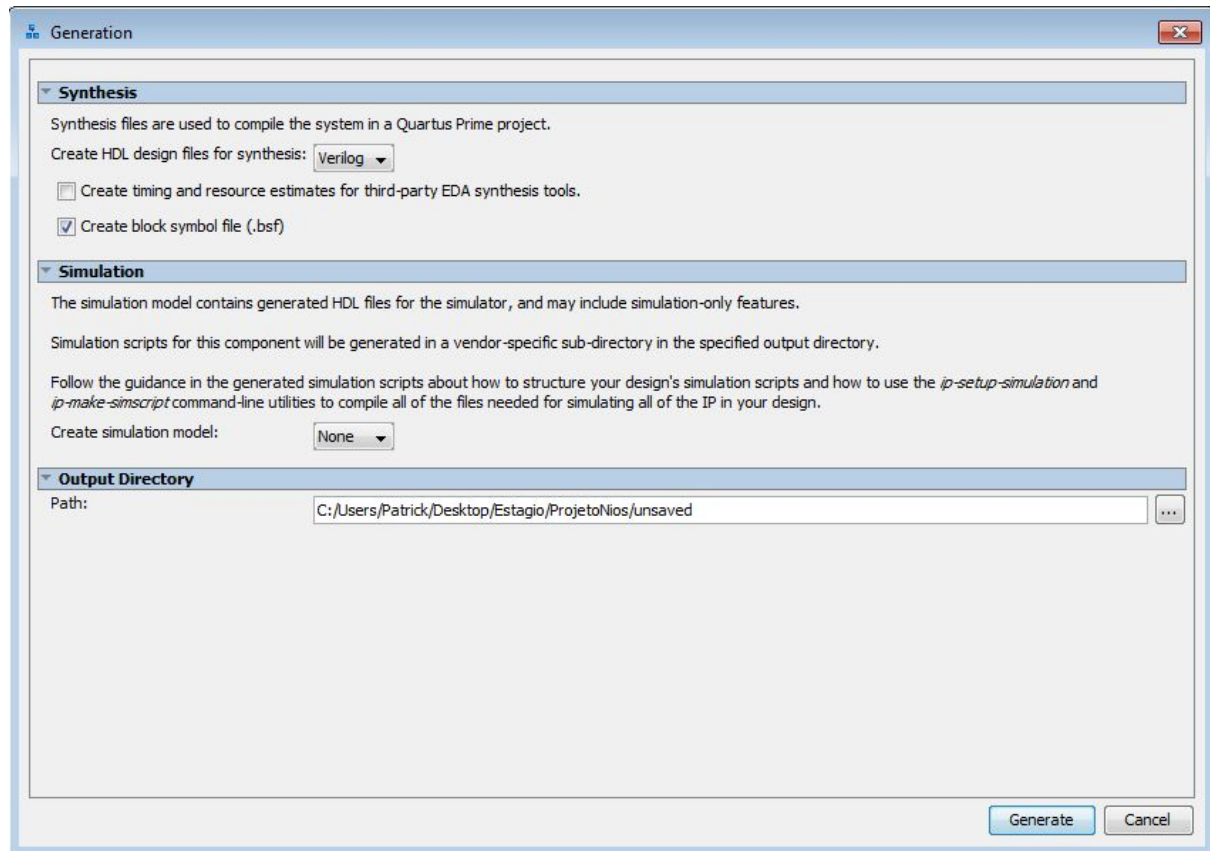


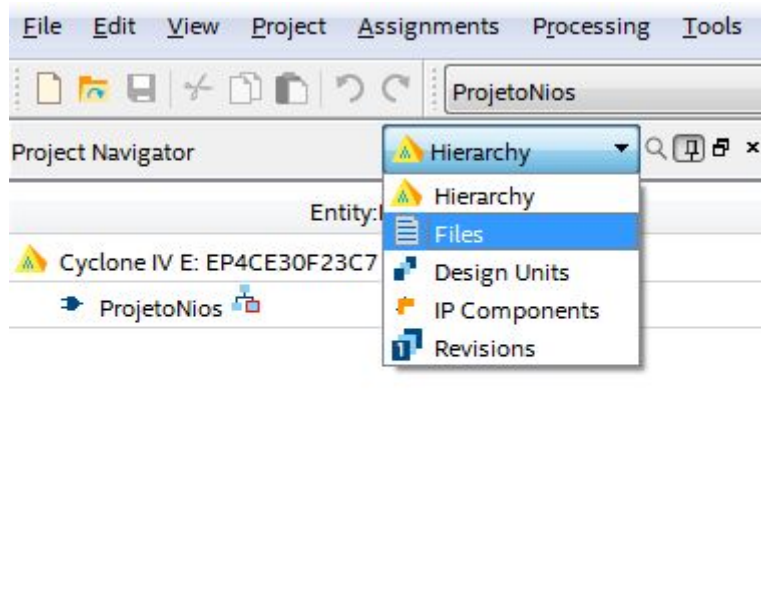
Figura 22: Geração do HDL.

Mudanças no projeto são simples de implementar, basta apenas re-abrir a ferramenta Qsys. Alguns componentes na ferramenta Qsys podem ser selecionados e editados ou deletados, ou um novo componente pode ser adicionado ao projeto.

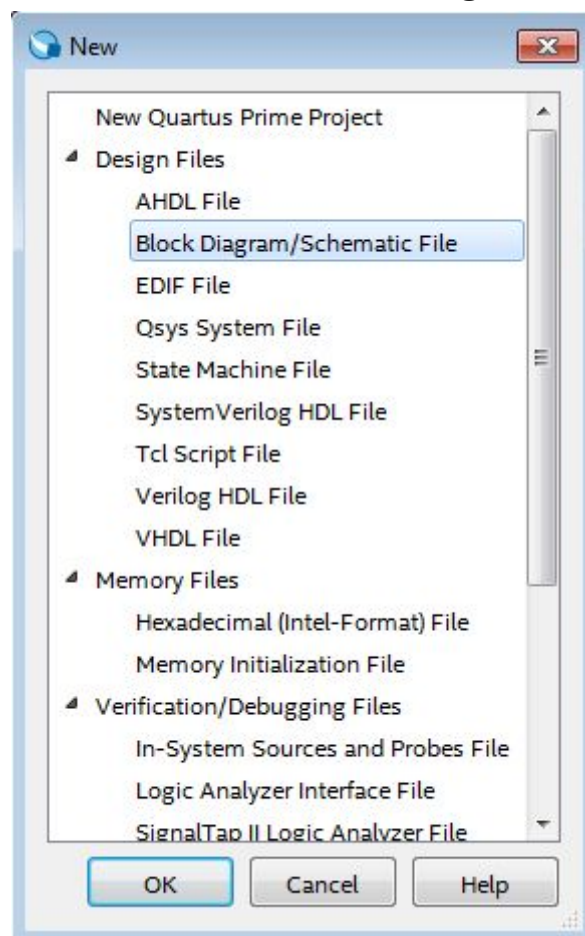
## Definindo Entidade Top Level do Projeto e compilando

1. Depois que o projeto foi construído utilizando a ferramenta Qsys, selecione no *Project Navigator* por *File* (Figura 23).

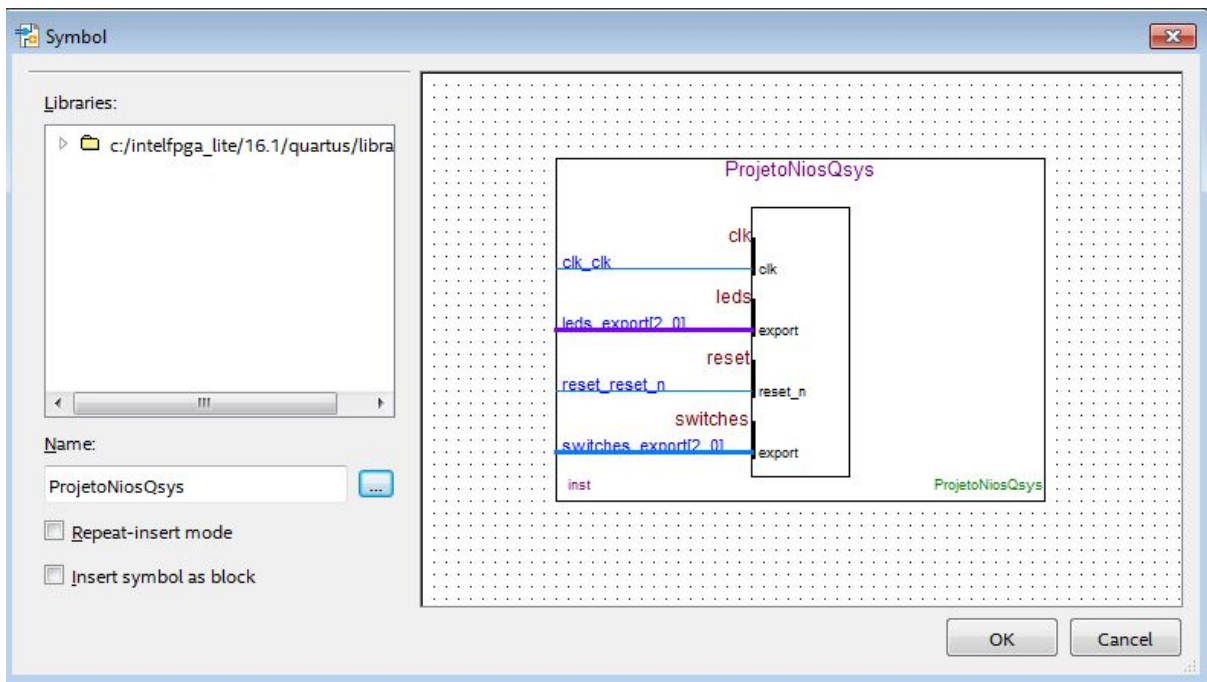
- Clique com o direito em Files dentro do Project Navigator e selecione **Add/Remove Files in Project..**
- Selecione o arquivo com extensão .qip dentro da pasta de synthesis e adicione ao projeto.



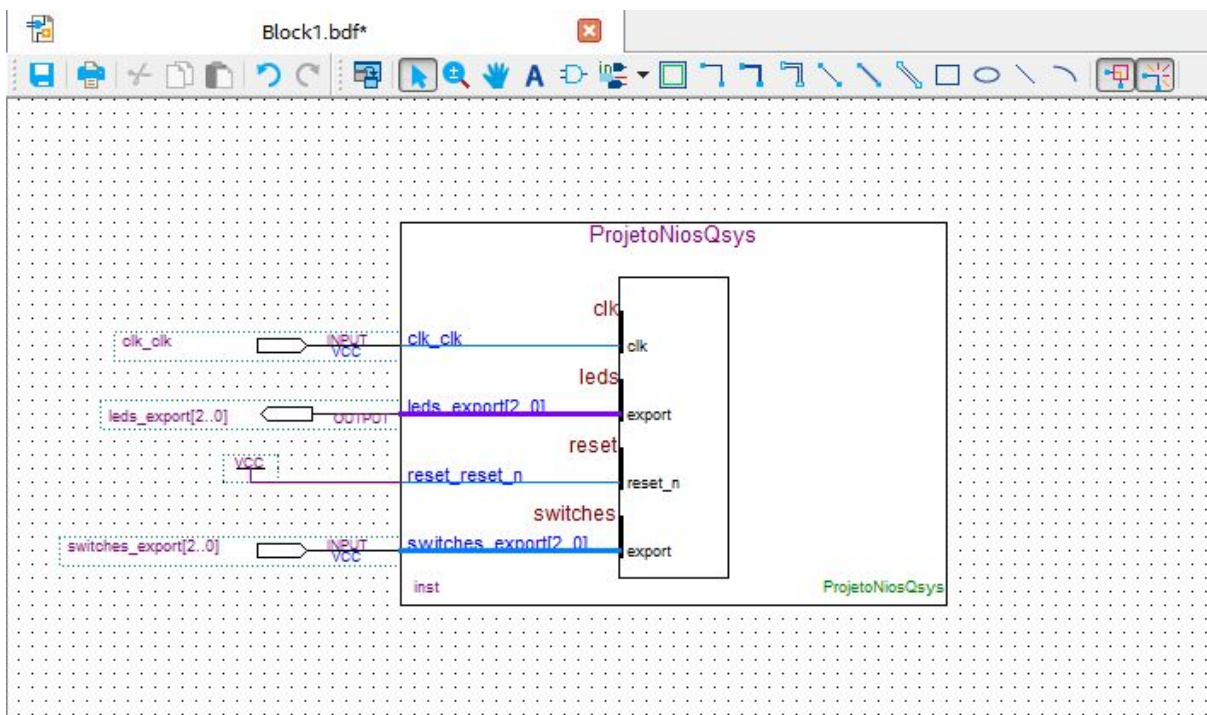
2. Agora crie um diagrama de blocos em **New > Block Diagram/Schematic File**.



3. Clique duas vezes no bloco e busque pelo diretório do projeto para selecionar o arquivo de extensão .bsf.



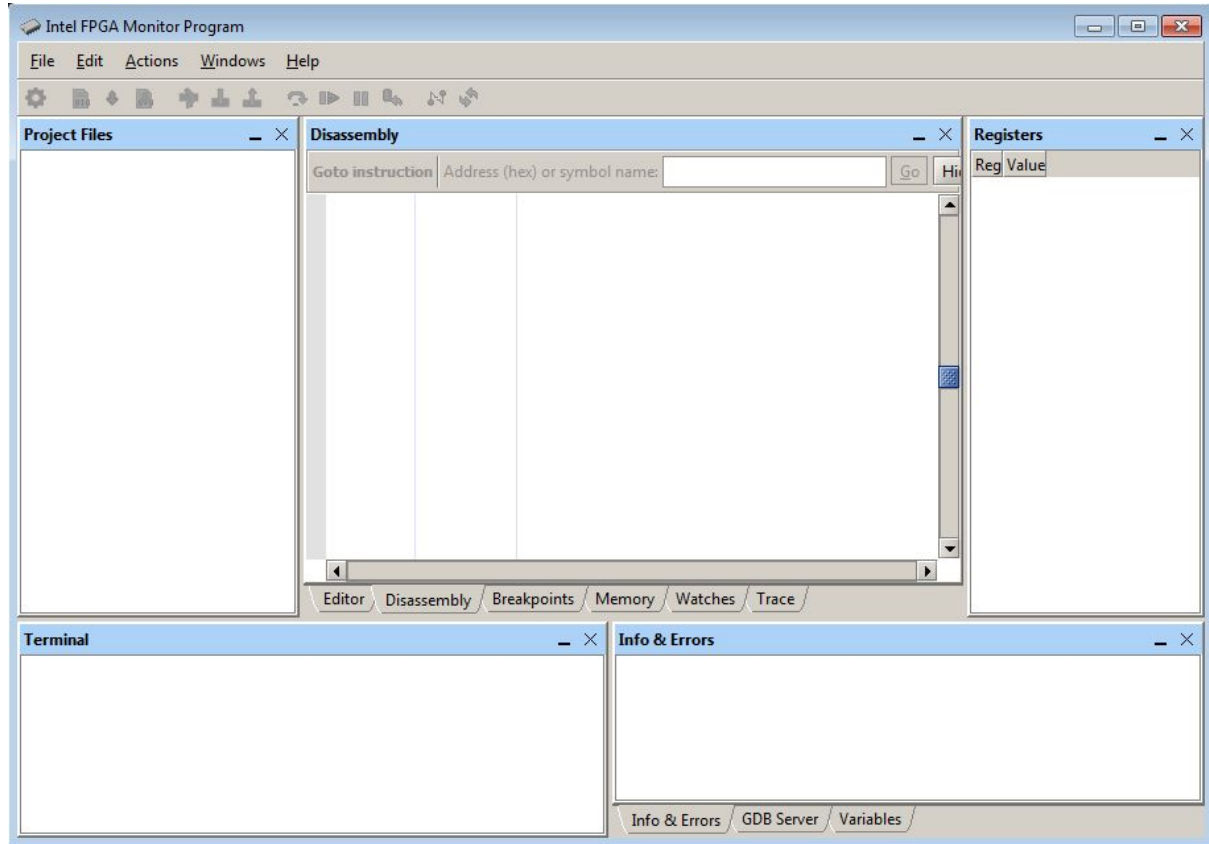
4. Insira a pinagem de entrada e saída ao bloco.



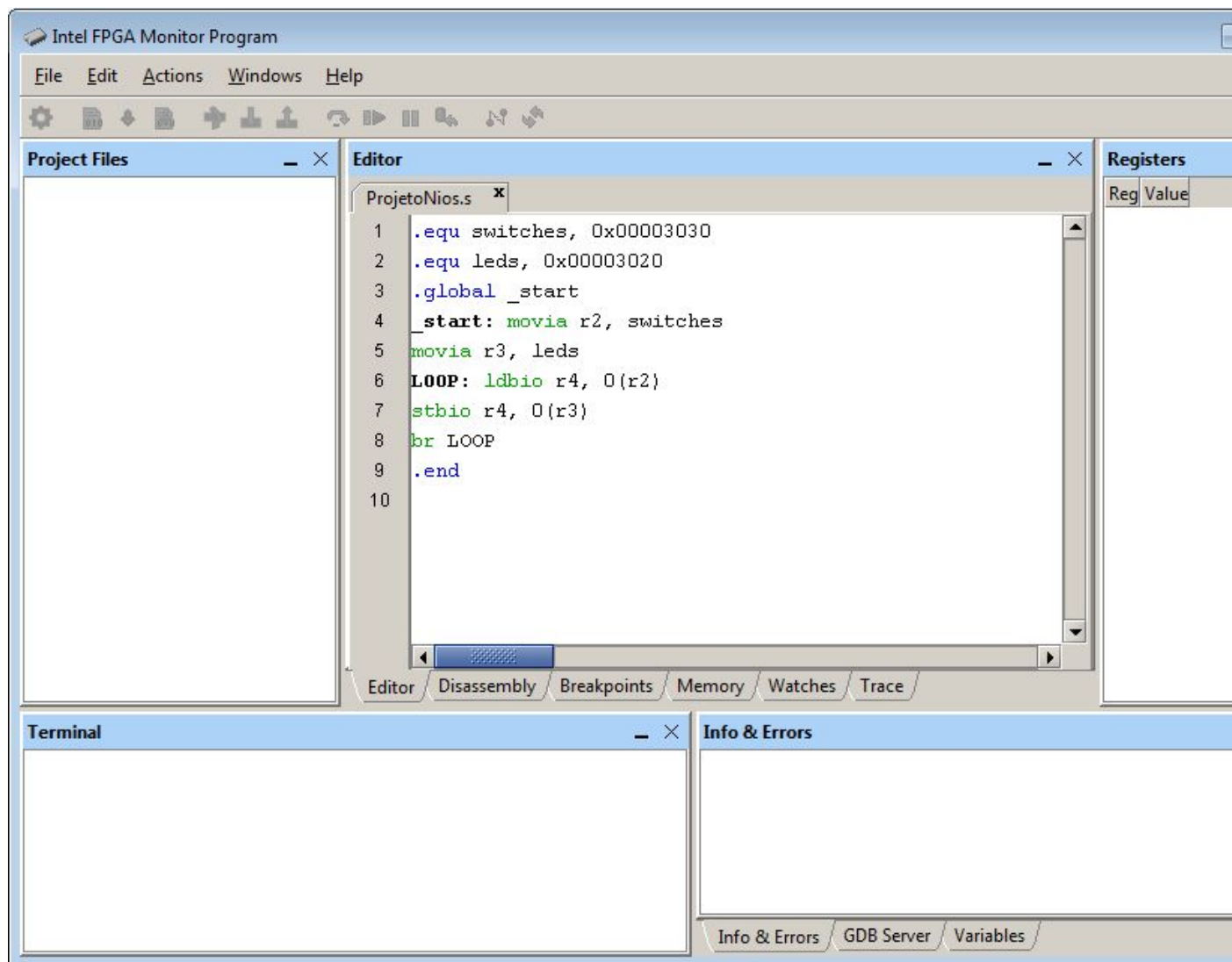
5. Por fim, compile o projeto.

## Gerando Hexadecimal do Assembly Com o Program Monitor

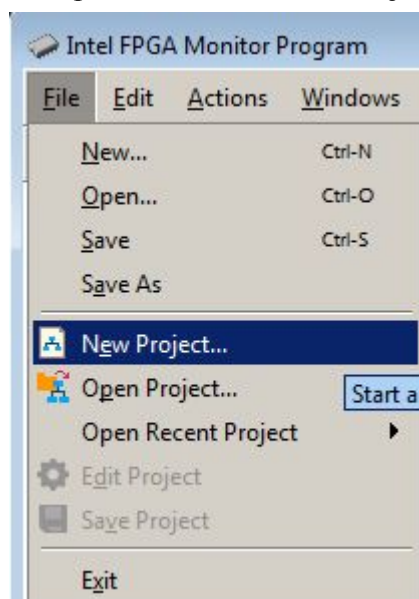
1. Abra o software Program Monitor.



2. Clique em **File > New...** e escreva o código da Figura X. Depois, salve com a extensão **.s** .



3. Depois, selecione **New Project...**, conforme Figura X.



4. Siga as seguintes configurações representadas pelas Figuras X, Z, Y.



New Project Wizard

File Settings | System Settings | Program Type | Program Settings | Connection Settings

### Specify a project name and directory

Project directory:  
C:\Users\Patrick\Desktop\Estagio\ProjetoNios\Monitor

Project name:  
ProjetoNios

Architecture: Nios II

New Project Wizard

File SettingsSystem SettingsProgram TypeProgram SettingsConnection Settings

# Specify a system

Select a system

< Custom System >Documentation

Specify a system by selecting a system description (SOPCInfo) file, and optional Quartus II programming (SOF) and Quartus II JTAG debugging information (JDI) files.

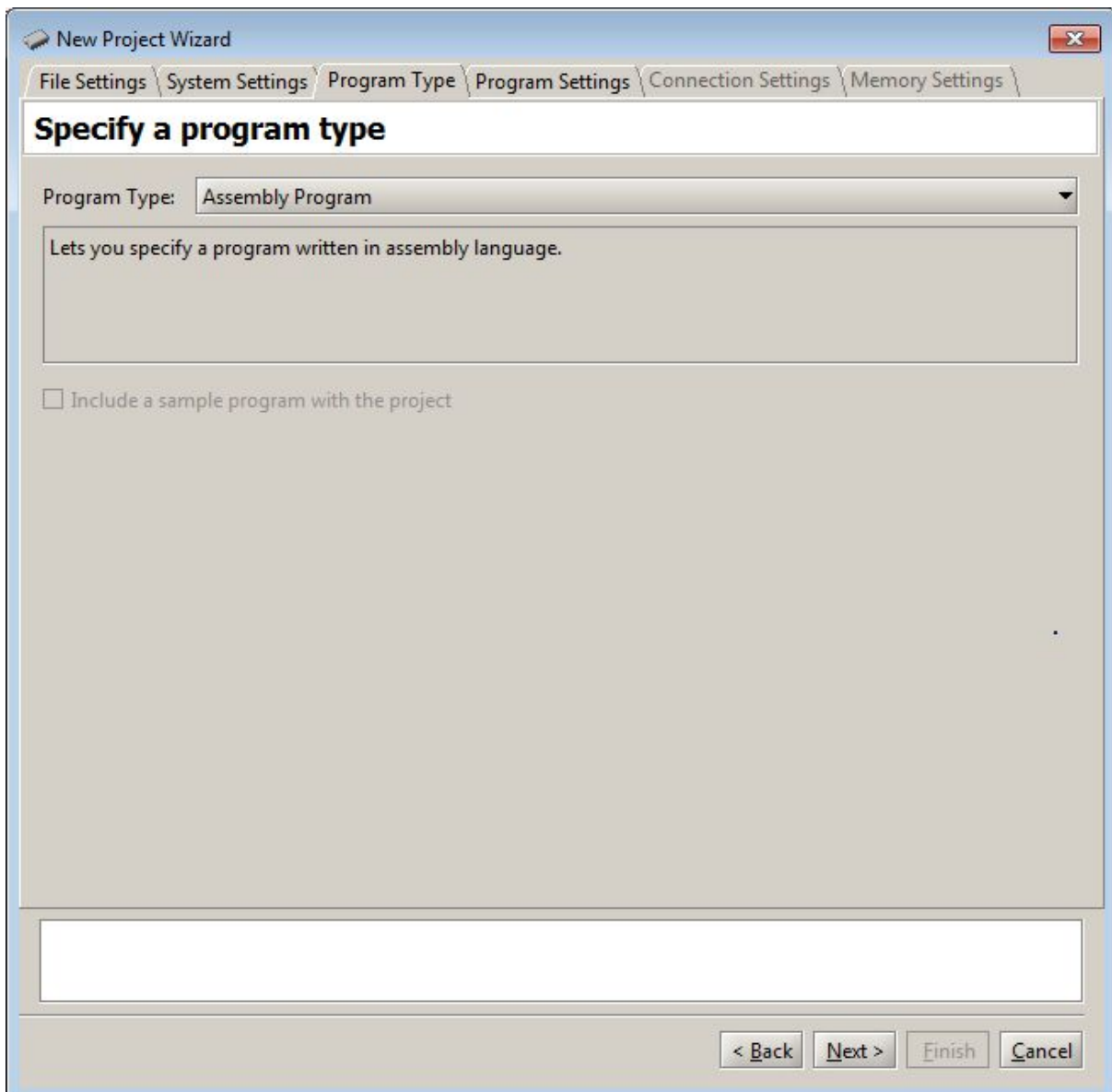
System details

System description file (SOPCInfo):  
C:/Users/Patrick/Desktop/Estagio/ProjetoNios/ProjetoNiosQsys.sopcinfoBrowse...

FPGA programming (SOF) file:  
C:/Users/Patrick/Desktop/Estagio/ProjetoNios/output\_files/ProjetoNios.sofBrowse...

The SOF file represents the FPGA programming file for the hardware system. If it is specified here, then the Monitor Program can be used to download this programming file onto the board. Otherwise, the system will need to be downloaded using some other method (for example, by using Quartus II).

< BackNext >FinishCancel



New Project Wizard

File SettingsSystem SettingsProgram TypeProgram SettingsConnection SettingsMemory Settings

Specify program details

Source files

First source file is used to determine the name of the binary program file.

C:/Users/Patrick/Desktop/Estagio/ProjetoNios/Monitor/ProjetoNios.s

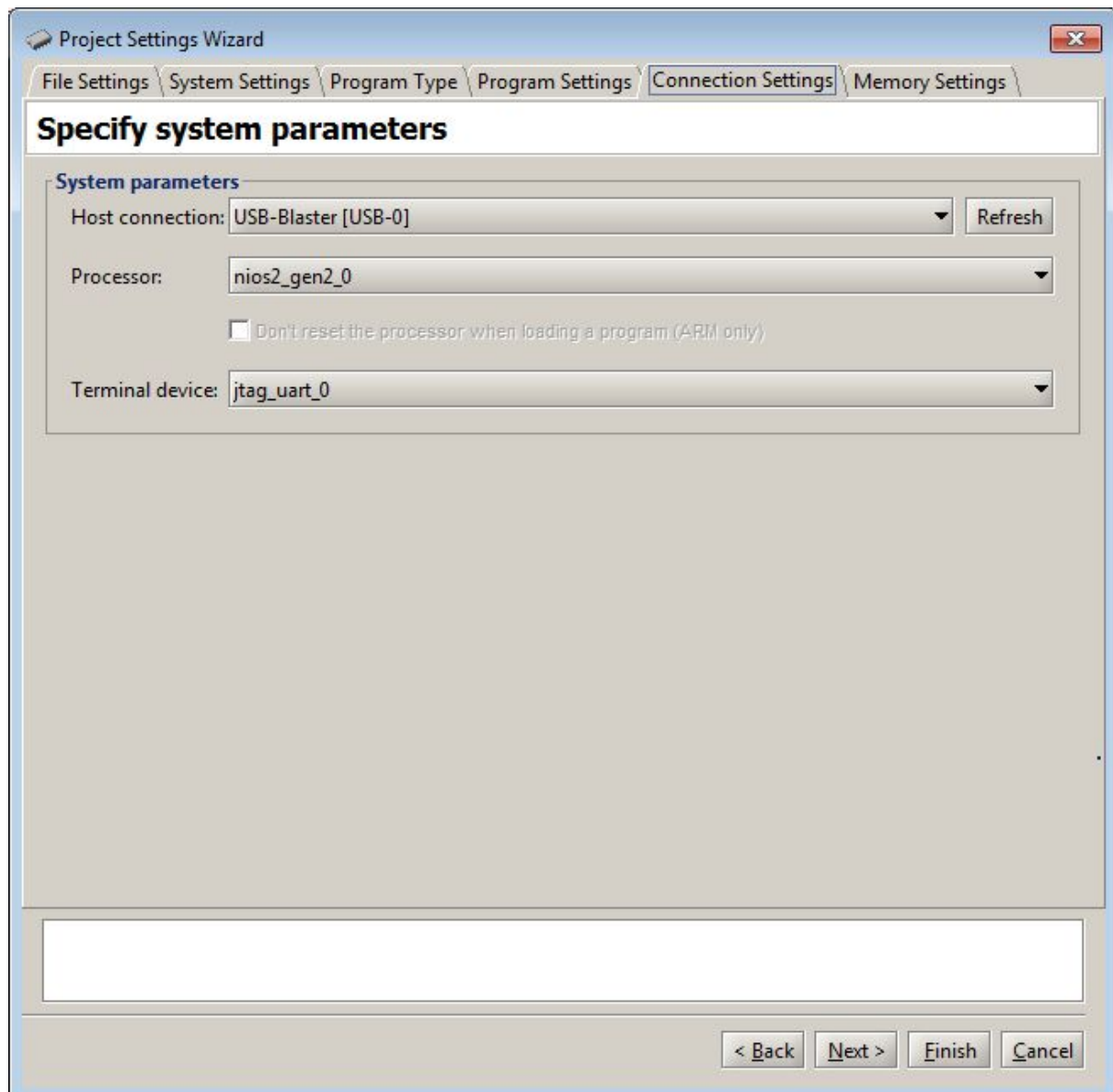
Add...RemoveUpDown

Program options

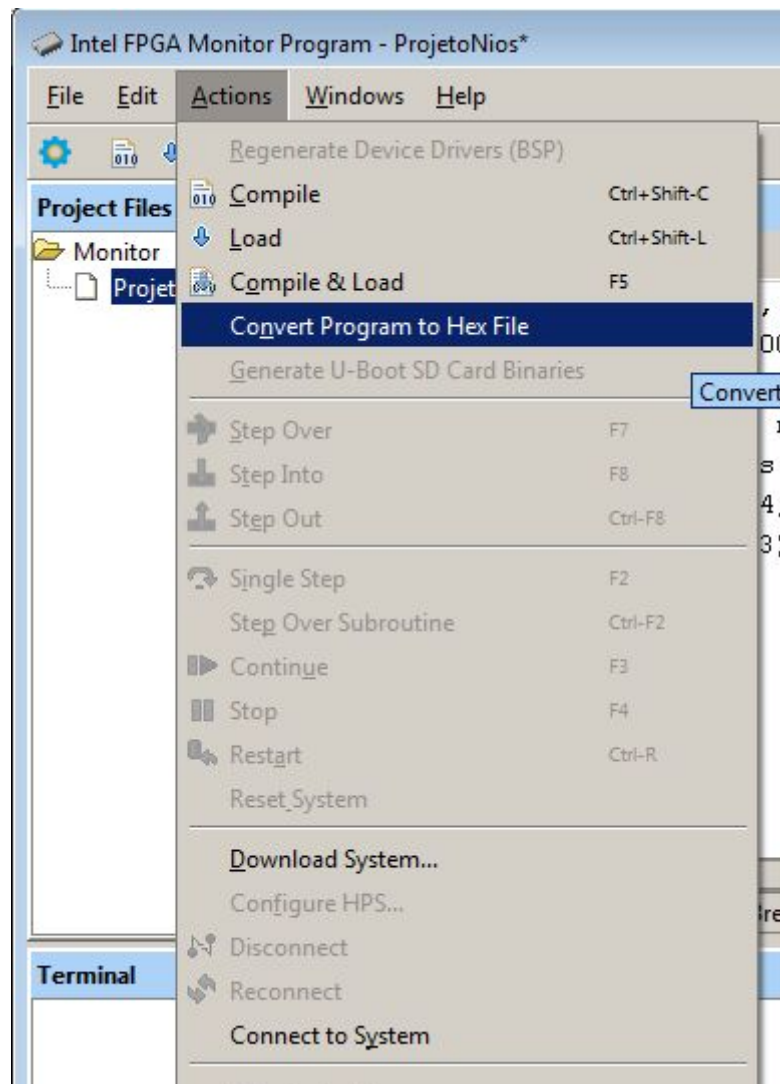
Start symbol: \_start

☐ Include system info directives file

< BackNext >FinishCancel



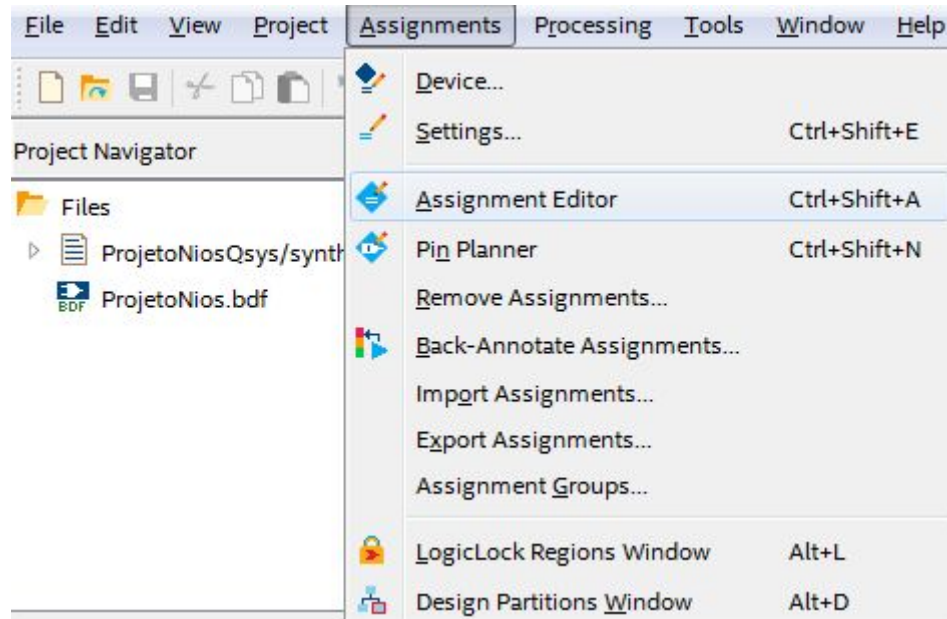
5. Compile o projeto no Program Monitor e selecione **Actions > Convert Program to Hex File**. Pronto, nesta etapa o programa em assembly foi convertido em hexadecimal. Esse arquivo .hex deve substituir o arquivo que foi setado no diretório de inicialização de memória.





## Finalizando Projeto

1. Volte para o projeto no quartus II e faça a pinagem na placa. Figura X.



2. Compile e utilize a função Programmer do quartus para transferir o projeto para a placa, finalizando o projeto.

## Projeto com componente RS232

A comunicação RS232 era muito utilizada para comunicação entre computadores e periféricos e é a forma mais popular de comunicação entre um CLP (controlador lógico programável) e um dispositivo externo. Nesse sentido, o projeto a seguir envia o caractere “p” para o computador via RS232.

## Construir o Projeto

Essa etapa deve

## Referencias

<https://www.embarcados.com.br/altera-nios-ii/>

[https://people.ece.cornell.edu/land/courses/ece5760/DE1\\_SOC/Introduction\\_to\\_the\\_Altera\\_Qsys\\_Tool.pdf](https://people.ece.cornell.edu/land/courses/ece5760/DE1_SOC/Introduction_to_the_Altera_Qsys_Tool.pdf)