

Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Prática em Sistemas Digitais (SSC-0108)

Relatório de Construção da CPU

Arthur Ernesto de Carvalho 14559479

Luiz Felipe Diniz Costa 13782032

Thiago Zero Araujo - 11814183

São Carlos
2023

Índice

Projeto Completo no Github	3
Introdução	3
Recursos Utilizados	4
Componentes da CPU	5
CLK (Clock)	5
Contador	5
ROM (Read-Only Memory)	5
Unidade de Controle	5
ULA (Unidade Lógica e Aritmética)	5
Registrador 1 e Registrador 2	5
Registradores	7
ULA (Unidade Lógica e Aritmética)	12
ROM (Read-Only Memory)	18
Contador	18
Unidade de Controle	22
Projeto Final	25

Projeto Completo no Github



Processador

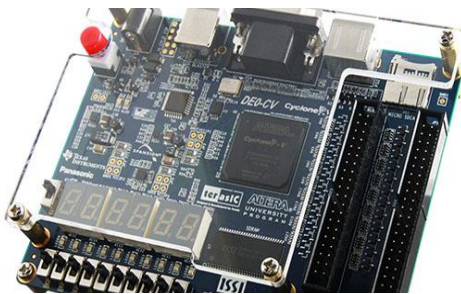
Desenvolvimento de uma Unidade Central de Processamento (CPU) com o auxílio da plataforma Quartus

Introdução

No campo da Ciência da Computação, especificamente na área de sistemas digitais, a construção de um processador é uma tarefa fundamental e desafiadora que requer conhecimentos interdisciplinares em eletrônica, lógica digital e arquitetura de computadores. Este relatório apresenta o processo de desenvolvimento de um processador por nós alunos do Instituto de Ciências Matemáticas e de Computação (ICMC) da Universidade de São Paulo. O processador projetado incorpora uma variedade de componentes essenciais que trabalham em conjunto para realizar operações lógicas e aritméticas, formando o núcleo funcional de um sistema digital.

Recursos Utilizados

FPGA Cyclone V DE0-CV => Código da placa Cyclone 5: 5CEBA4F23C7 Confira o [Manual da FPGA](#)



FPGA Cyclone V DE0-CV

Código da placa Cyclone 5: 5CEBA4F23C7

Confira o Manual da FPGA



Quartus Prime

O Quartus Prime é uma plataforma de design digital versátil para criação e otimização de circuitos integrados

Componentes da CPU

Nosso processador é composto por uma variedade de componentes, que incluem:

CLK (Clock)

O sinal de clock é a pulsação que sincroniza todos os componentes da CPU, garantindo a execução ordenada das operações.

Contador

O contador é um elemento vital na CPU, responsável por gerar uma sequência ordenada de valores que auxiliam na coordenação das operações. Ele é utilizado para acessar comandos na memória ROM

ROM (Read-Only Memory)

A memória ROM contém as instruções do programa em formato binário. Essas instruções são buscadas para serem executadas pela CPU.

Unidade de Controle

A unidade de controle é responsável por interpretar as instruções buscadas na ROM e coordenar a execução das operações correspondentes. Ela emite os sinais apropriados para os demais componentes da CPU.

ULA (Unidade Lógica e Aritmética)

A ULA é o componente da CPU que executa operações lógicas (como AND, OR, NOT) e aritméticas (como adição e subtração) sobre os dados presentes nos registradores. O seu controle é feito pela Unidade de Controle.

Registrador 1 e Registrador 2

Os registradores são elementos de armazenamento temporário, são utilizados para a CPU armazenar valores para uso

posterior.

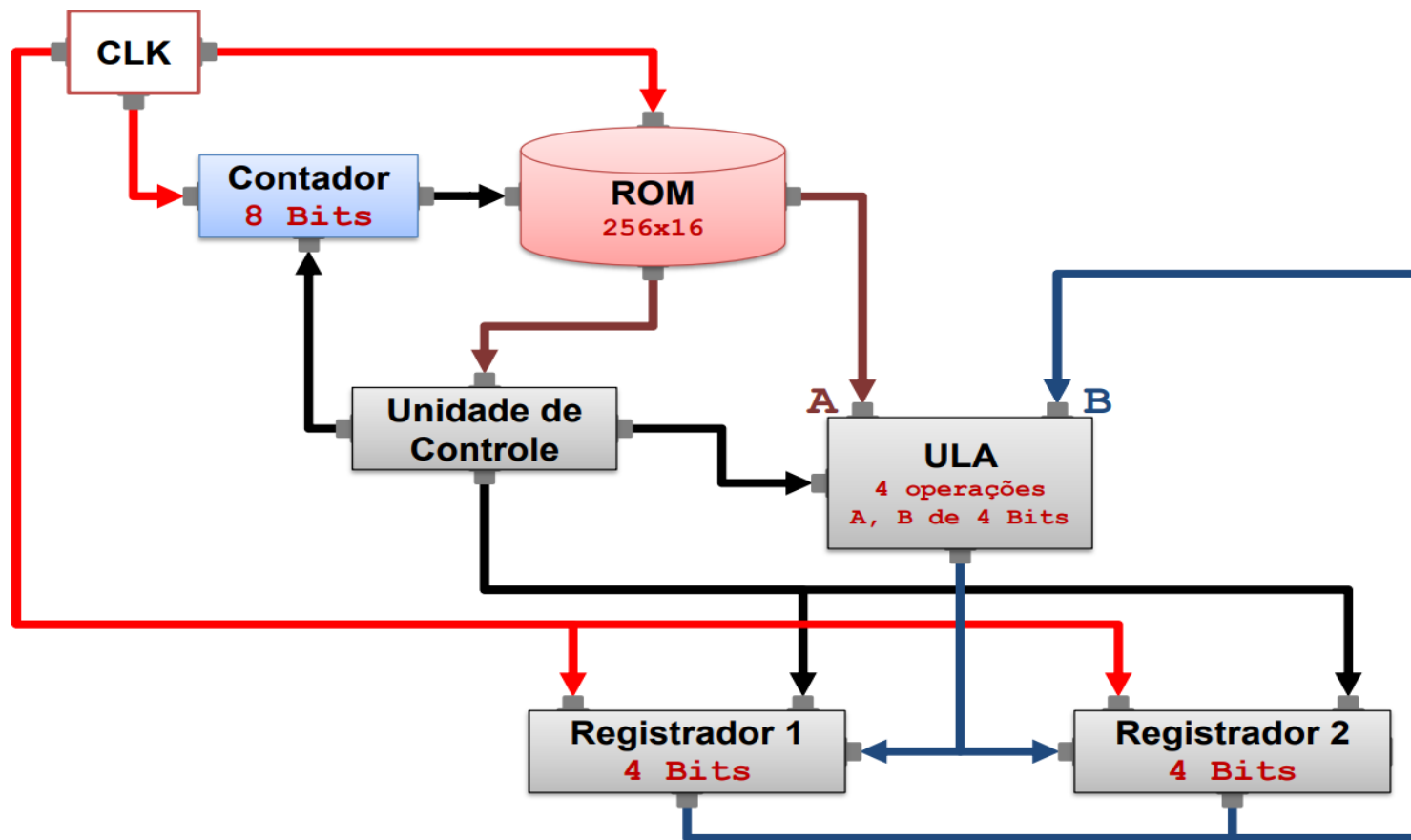
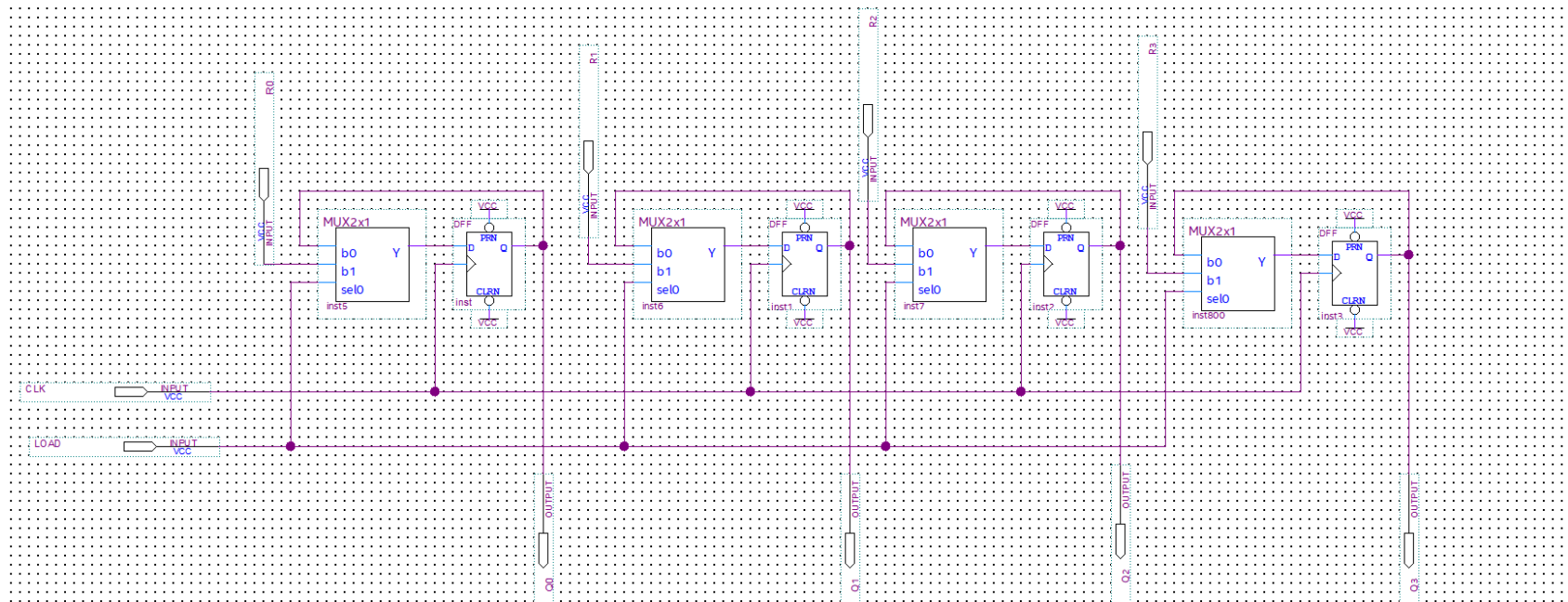


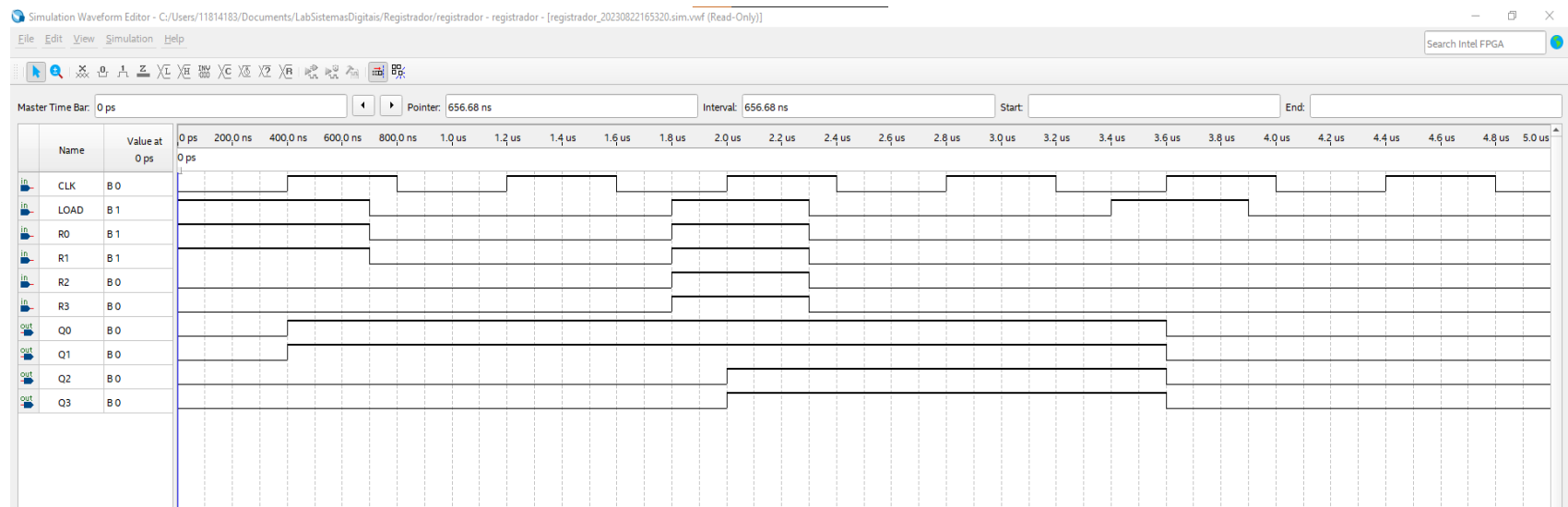
Ilustração do funcionamento do Processador

Registradores

Para termos memória, construímos um registrador de 4 bits. Esse registrador é composto por um conjunto de 4 Flip-Flops do tipo D, cada qual armazena um bit. O registrador recebe 4 bits e o sinal de clock e a sua saída são os 4 bits armazenados. Transformamos esse registrador em um block para uso posterior no projeto final. Abaixo seguem imagens do circuito e de sua simulação



Circuito do Registrador feito para simulações (sem o *Debouncer*)



Simulação Funcional do comportamento de um registrador de 4 bits de carga paralela.

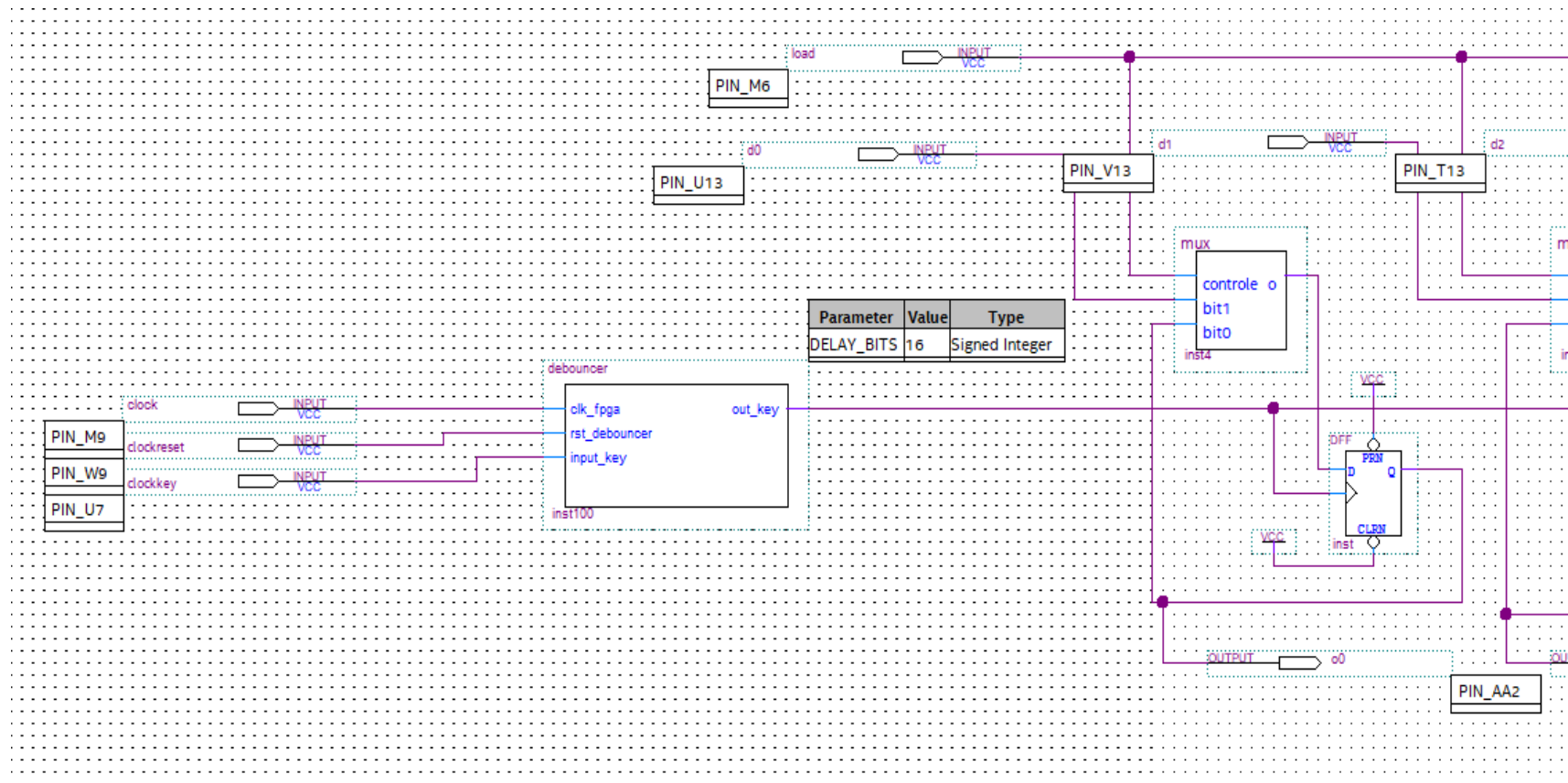


Imagem 1 ⇒ Circuito

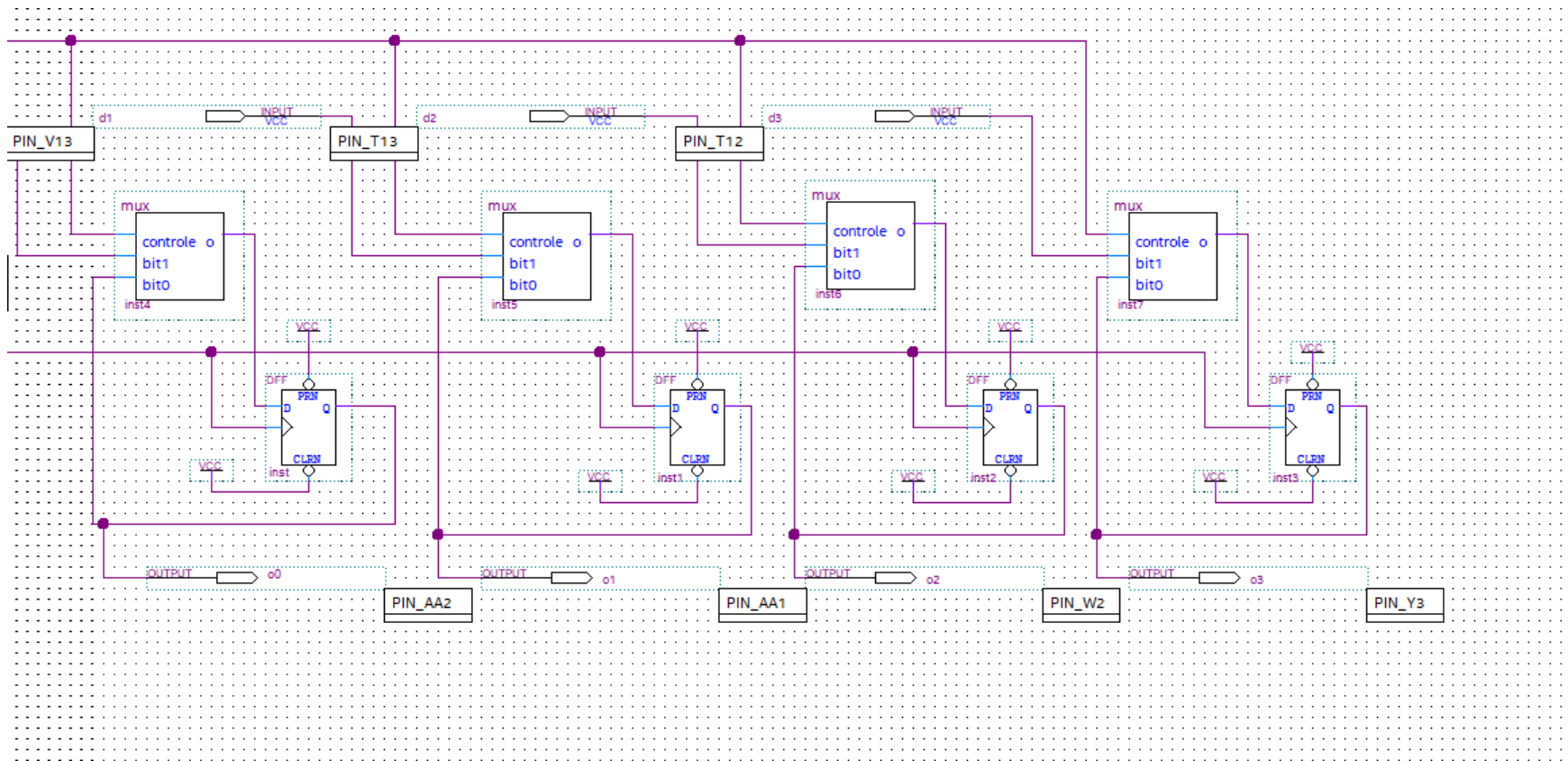












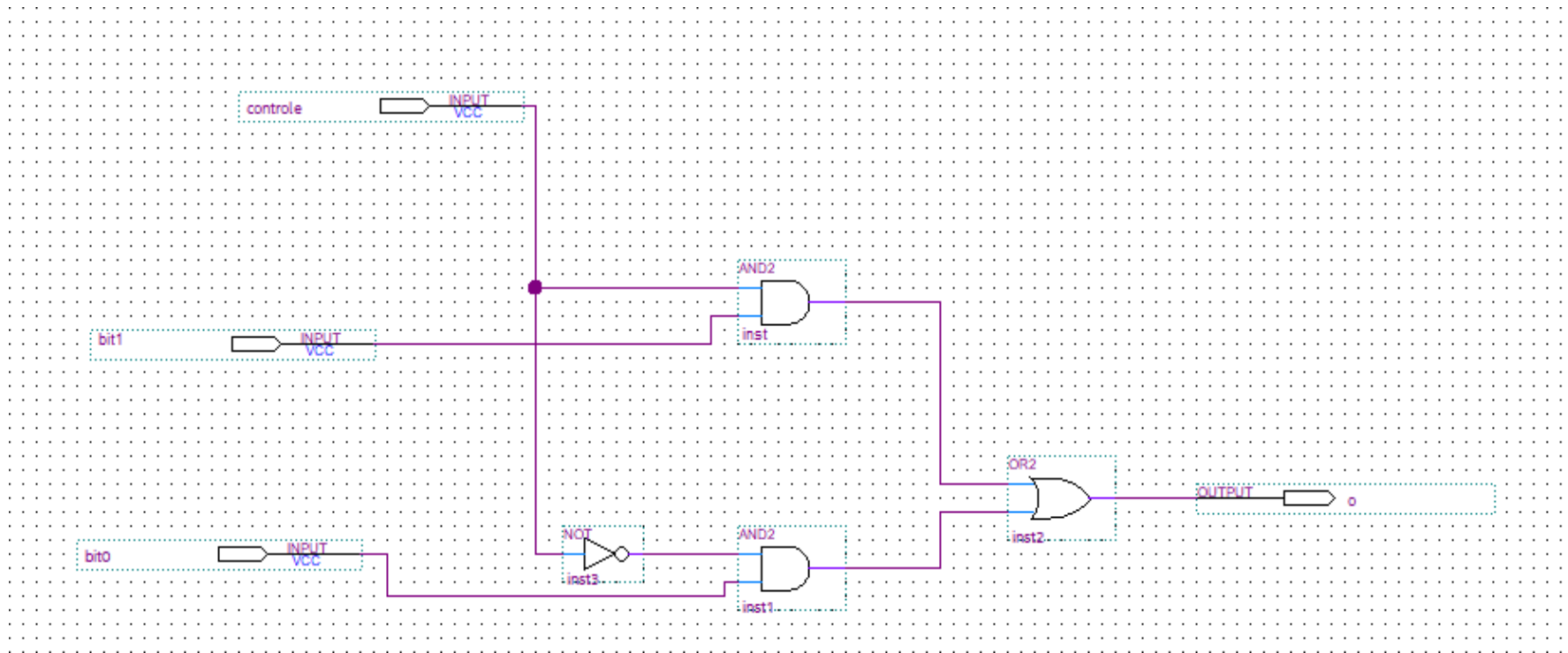


Imagem 2 ⇒ Circuito

tu	From	To	Assignment Name	Value	Enabled	Entity	Comment	Tag
▶		 clockkey	Location	PIN_U7	Yes			
▶		 clockreset	Location	PIN_W9	Yes			
▶		 d0	Location	PIN_U13	Yes			
▶		 d1	Location	PIN_V13	Yes			
▶		 d2	Location	PIN_T13	Yes			
▶		 d3	Location	PIN_T12	Yes			
▶		 o0	Location	PIN_AA2	Yes			
▶		 o1	Location	PIN_AA1	Yes			
▶		 o2	Location	PIN_W2	Yes			
▶		 o3	Location	PIN_Y3	Yes			
▶		 load	Location	PIN_M6	Yes			
▶		 clock	Location	PIN_M9	Yes			
	<<new>>	<<new>>	<<new>>					

Configuração dos Pinos na FPGA



Implementação do Mux

ULA (Unidade Lógica e Aritmética)

Aqui, exibimos com a representação da Unidade Lógica e Aritmética (ULA) do nosso processador, integrada no software do Quartus.

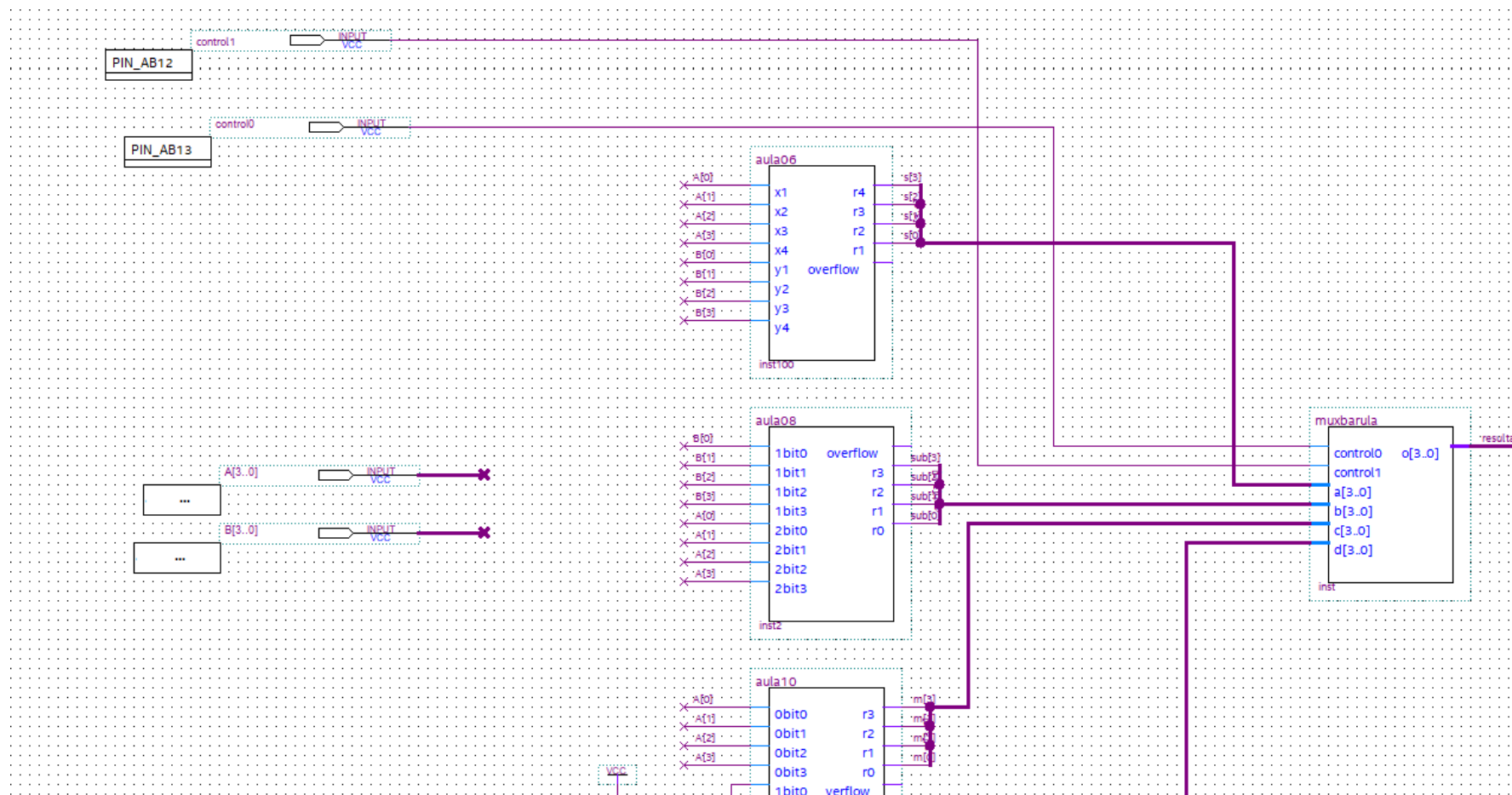


Imagem 1 ⇒ ULA

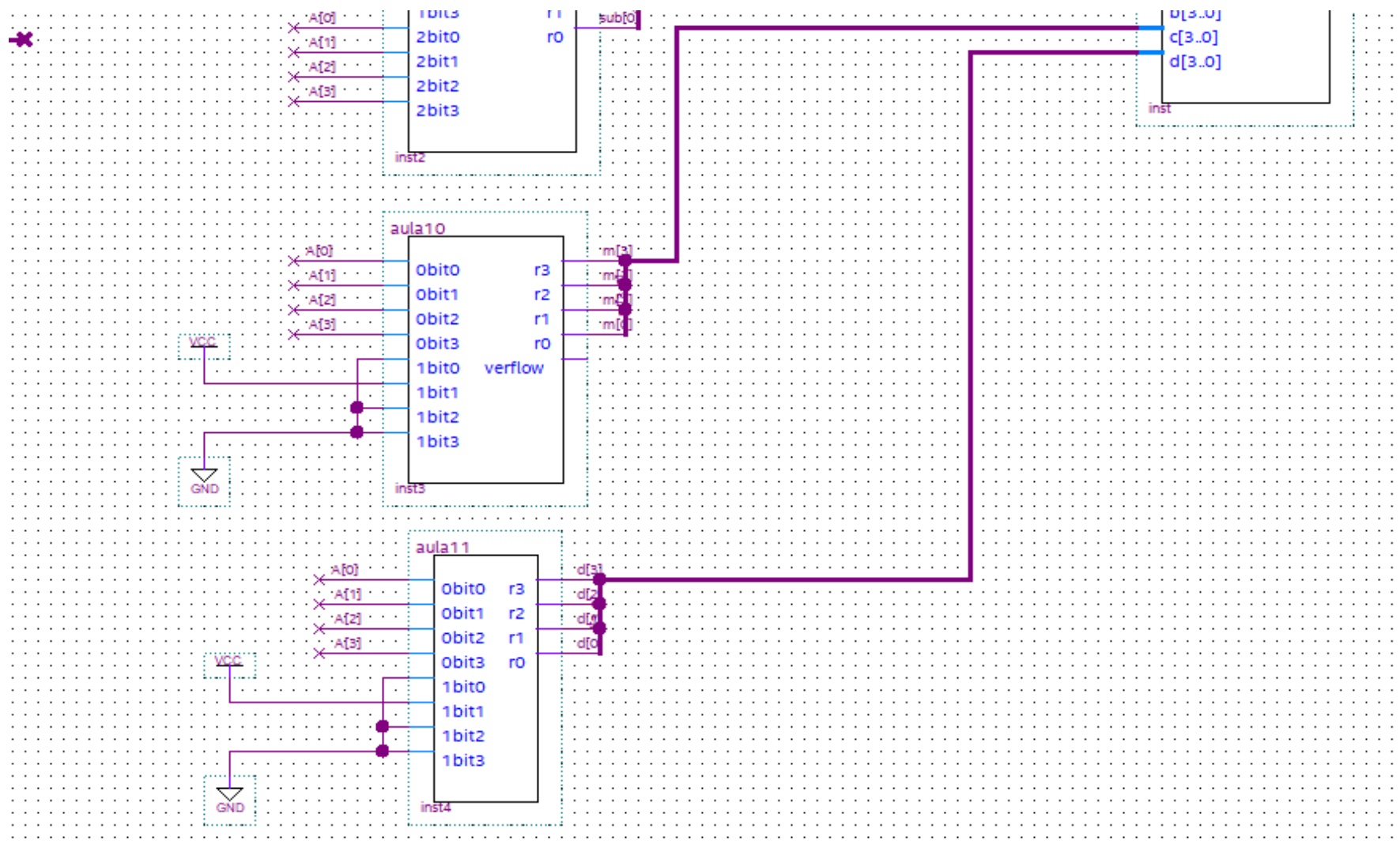


Imagem 2 ⇒ ULA

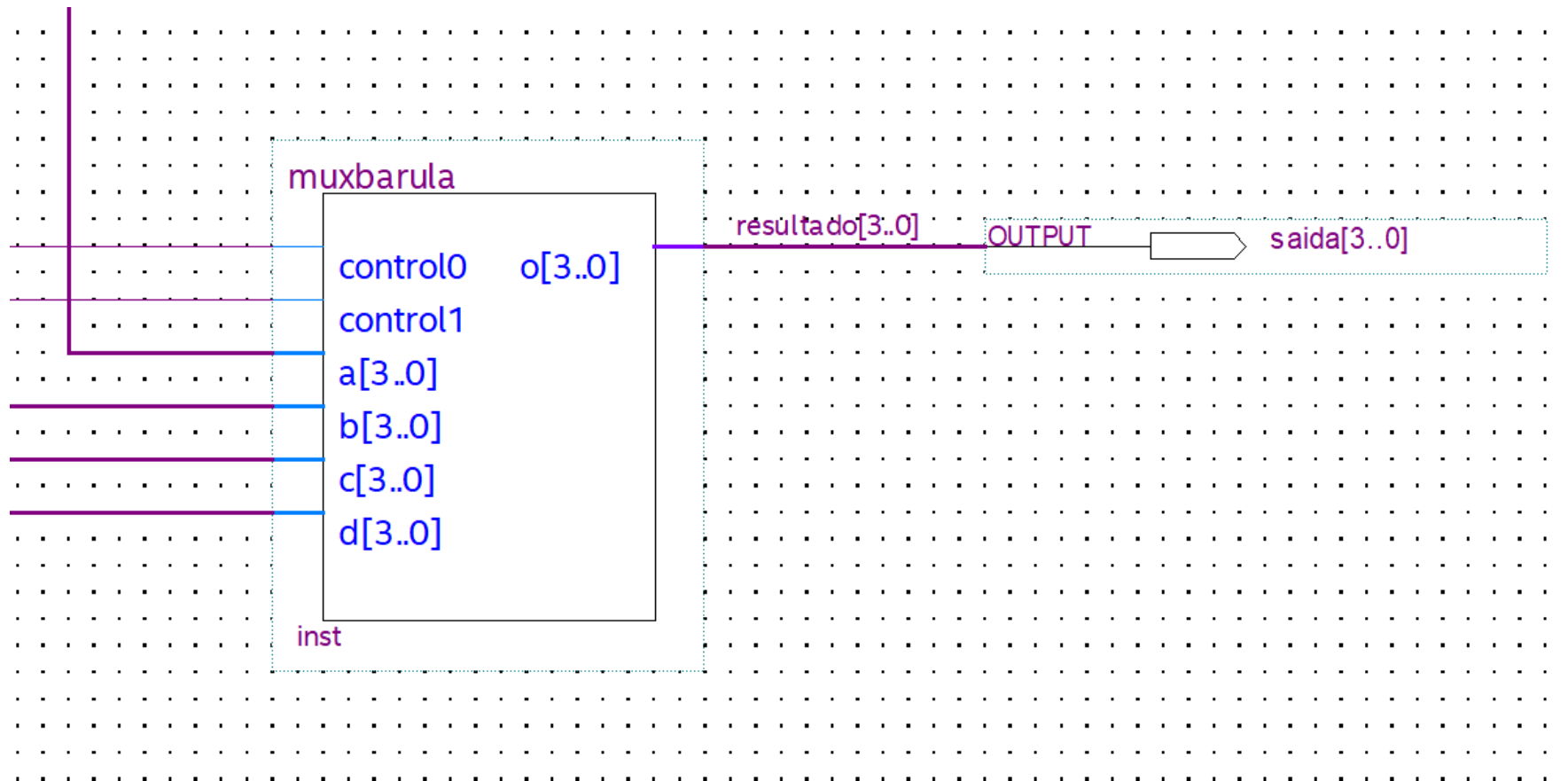
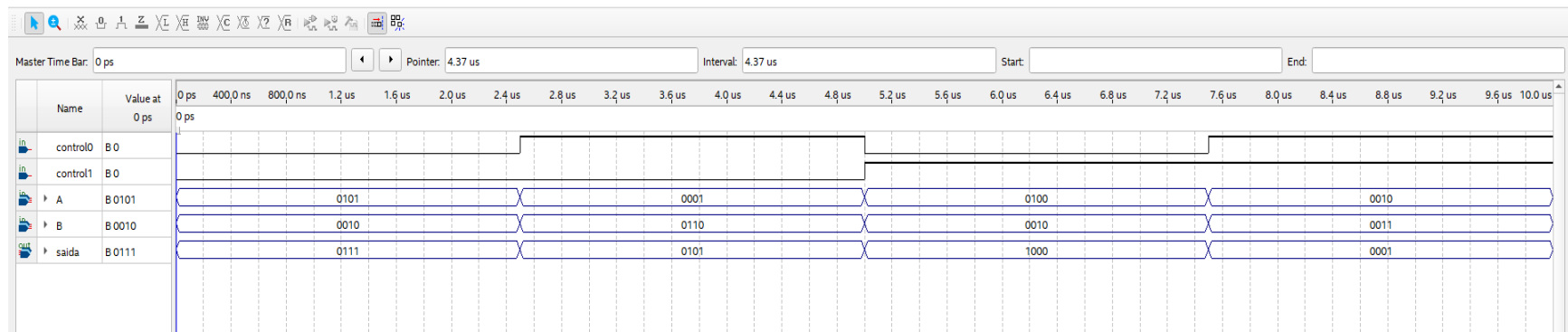


Imagem 3 ⇒ ULA

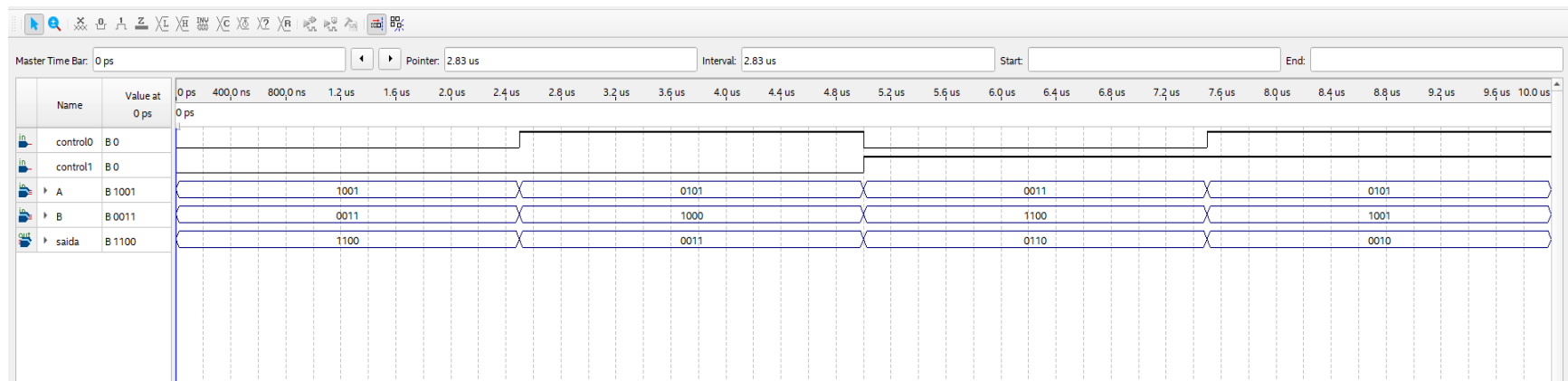
	status	From	To	Assignment Name	Value	Enabled	Entity
1	✓		control1	Location	PIN_AB12	Yes	
2	✓		control0	Location	PIN_AB13	Yes	
3	✓		toptop	Location	PIN_U21	Yes	
4	✓		topright	Location	PIN_V21	Yes	
5	✓		botright	Location	PIN_W22	Yes	
6	✓		botbot	Location	PIN_W21	Yes	
7	✓		botleft	Location	PIN_Y22	Yes	
8	✓		topleft	Location	PIN_Y21	Yes	
9	✓		mid	Location	PIN_AA22	Yes	
10	✓		A[0]	Location	PIN_U13	Yes	
11	✓		A[1]	Location	PIN_V13	Yes	
12	✓		A[2]	Location	PIN_T13	Yes	
13	✓		A[3]	Location	PIN_T12	Yes	
14	✓		B[0]	Location	PIN_AA15	Yes	
15	✓		B[1]	Location	PIN_AB15	Yes	
16	✓		B[2]	Location	PIN_AA14	Yes	
17	✓		B[3]	Location	PIN_AA13	Yes	

Configuração dos Pinos na FPGA

Abaixo está a simulação da nossa ULA, a qual submetemos a testes utilizando diferentes valores, representados nas imagens abaixo. Neste contexto, podemos observar que a ULA está apresentando um desempenho satisfatório.



Simulação com valor x



Simulação com valor y

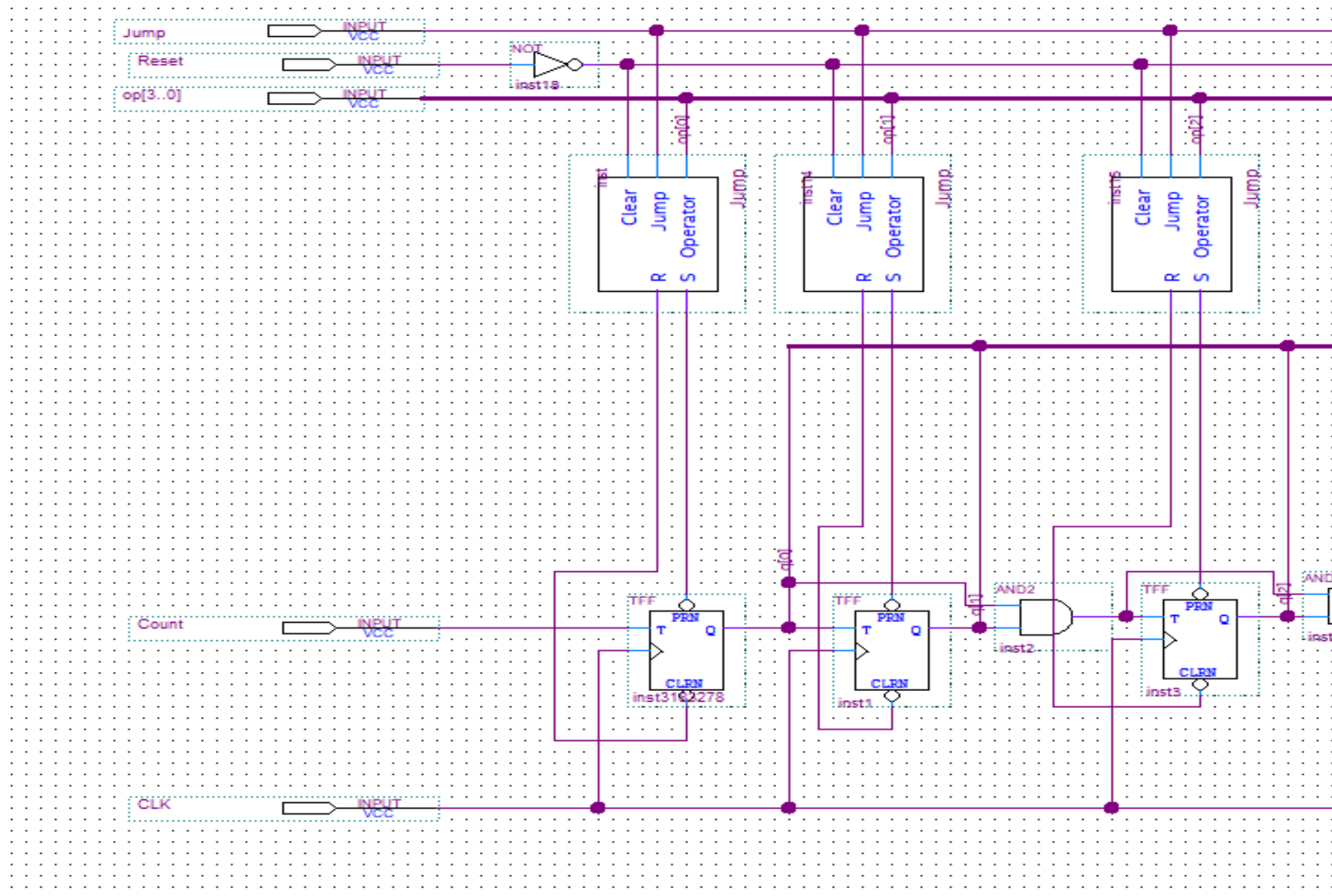
ROM (Read-Only Memory)

Para armazenar os comandos a serem executados pela CPU, construímos a memória ROM. A memória ROM é um componente que armazena dados permanentes, os quais no nosso caso será os dados necessários para os comandos.

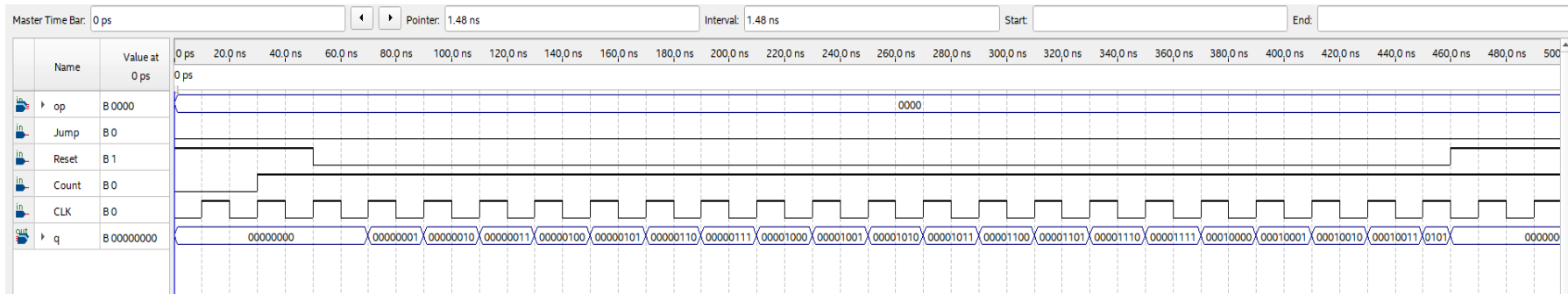
No contexto atual, o professor já nos forneceu a memória ROM, apenas sendo necessário criar o bloco dentro do Quartus. Quando tivermos o contador construído poderemos acessar os comandos da ROM de forma sequencial.

Contador

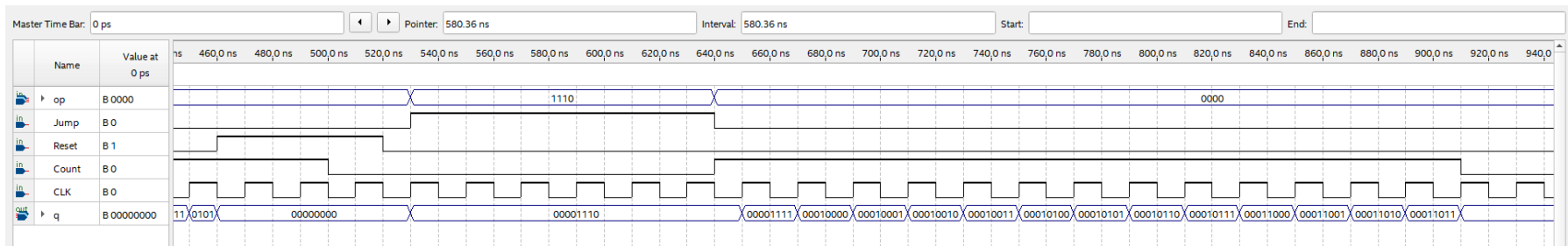
Neste projeto, estruturamos um Contador Síncrono de 8 bits, que é capaz de representar 256 estados distintos. Transformamos o contador em um bloco para uso no projeto final. No projeto do contador, empregamos instruções de salto (JUMPs), com operandos de 4 bits.



Contador usando Jumper ⇒ Imagem 1



Simulação Funcional do comportamento do contador ⇒ Parte 1

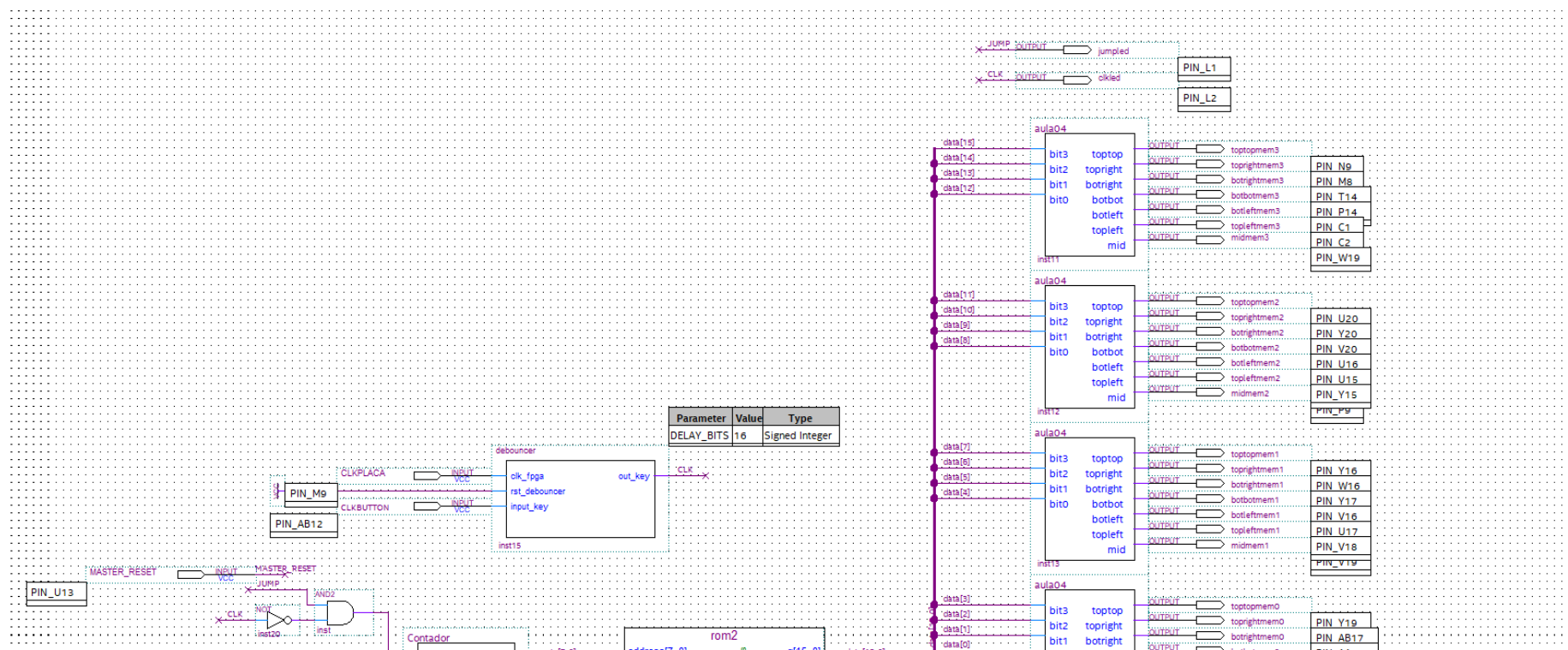


Simulação Funcional do comportamento do contador ⇒ Parte 2

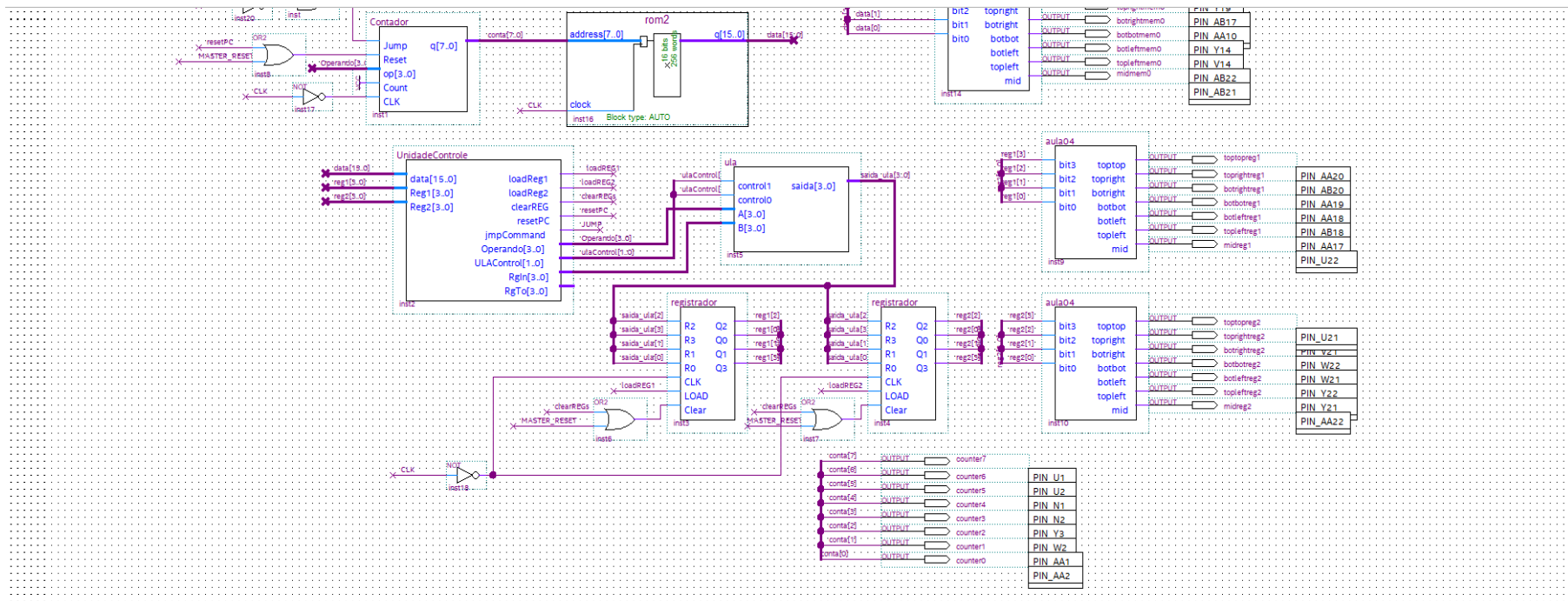
Unidade de Controle

Nessa etapa construímos a Unidade de Controle, a qual será utilizada para interpretar os comandos da memória ROM, e enviar sinais para o contador em caso de jump, para a ULA em caso de operações aritméticas e para os registradores receberem informações.

A seguir, apresentam-se as simulações realizadas com a Unidade de Controle.



Esquemático do circuito da unidade de controle ⇒ Parte 1



Esquemático do circuito da unidade de controle - Parte 2

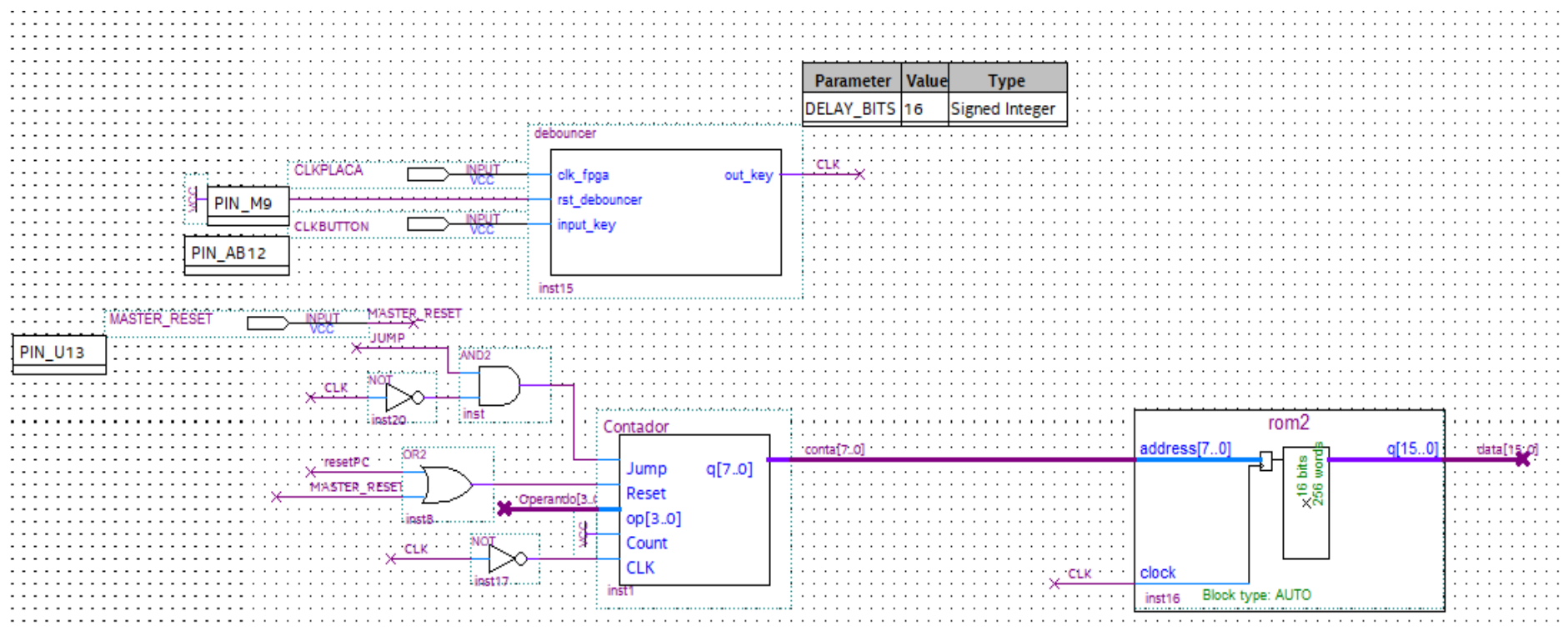
Projeto Final

Por fim, integramos tudo que havíamos construído para formar o nosso processador. A cada subida de clock, a memória ROM utiliza o endereço armazenado no contador para levar o comando desejado à Unidade de Controle, assim a unidade de controle processa o comando e disponibiliza os sinais necessários para o contador, ULA e os dois registradores. Já na descida do clock, o contador avança em sua contagem e realiza um jump se indicado pela Unidade de Controle e os registradores salvam a informação disponibilizada na saída da ULA se indicado pela Unidade de Controle.

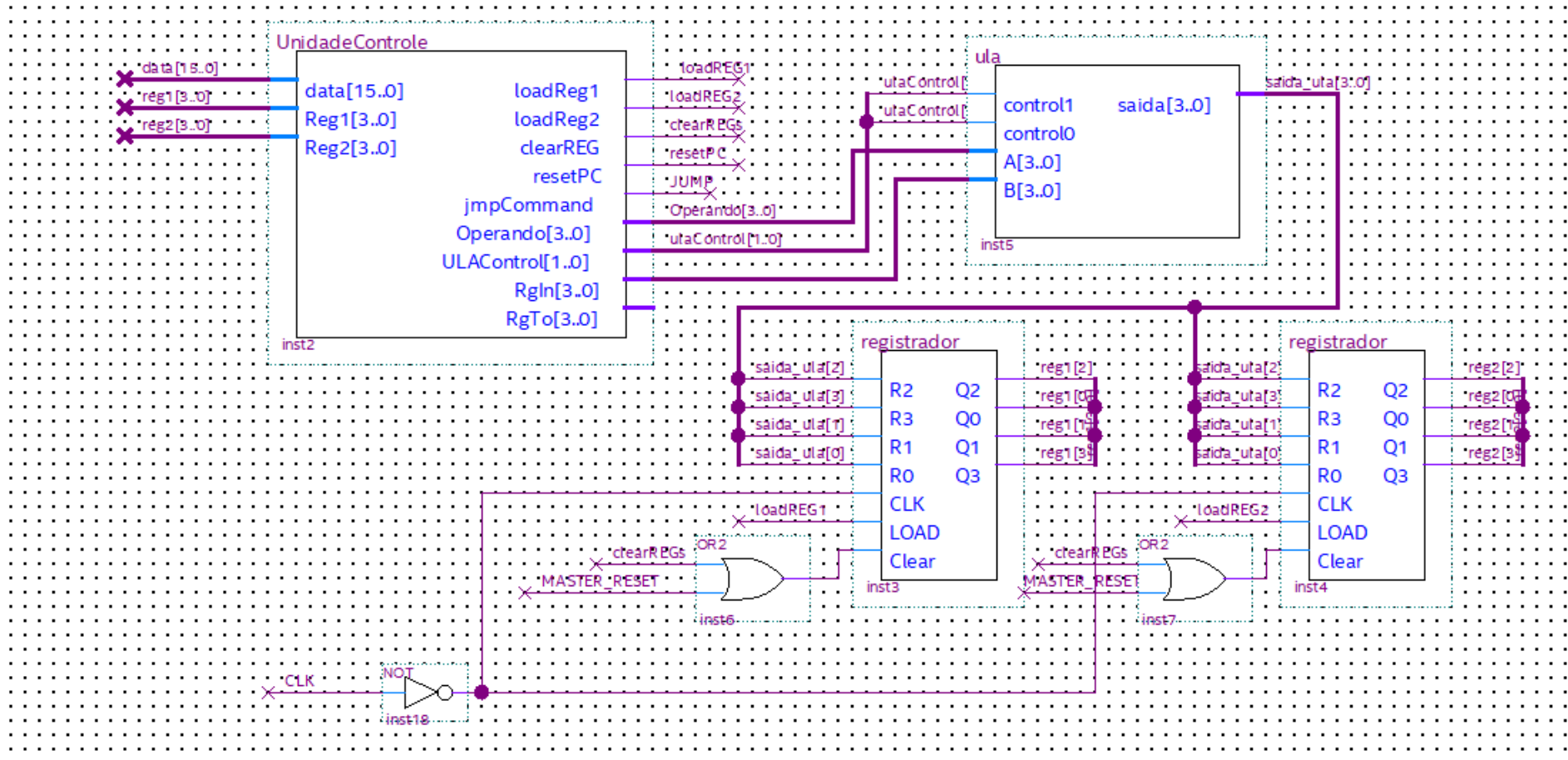
Dessa forma a nossa CPU é composta pelos seguintes componentes:

- Contador
- Memória ROM
- Unidade de Controle
- ULA
- 2 Registradores

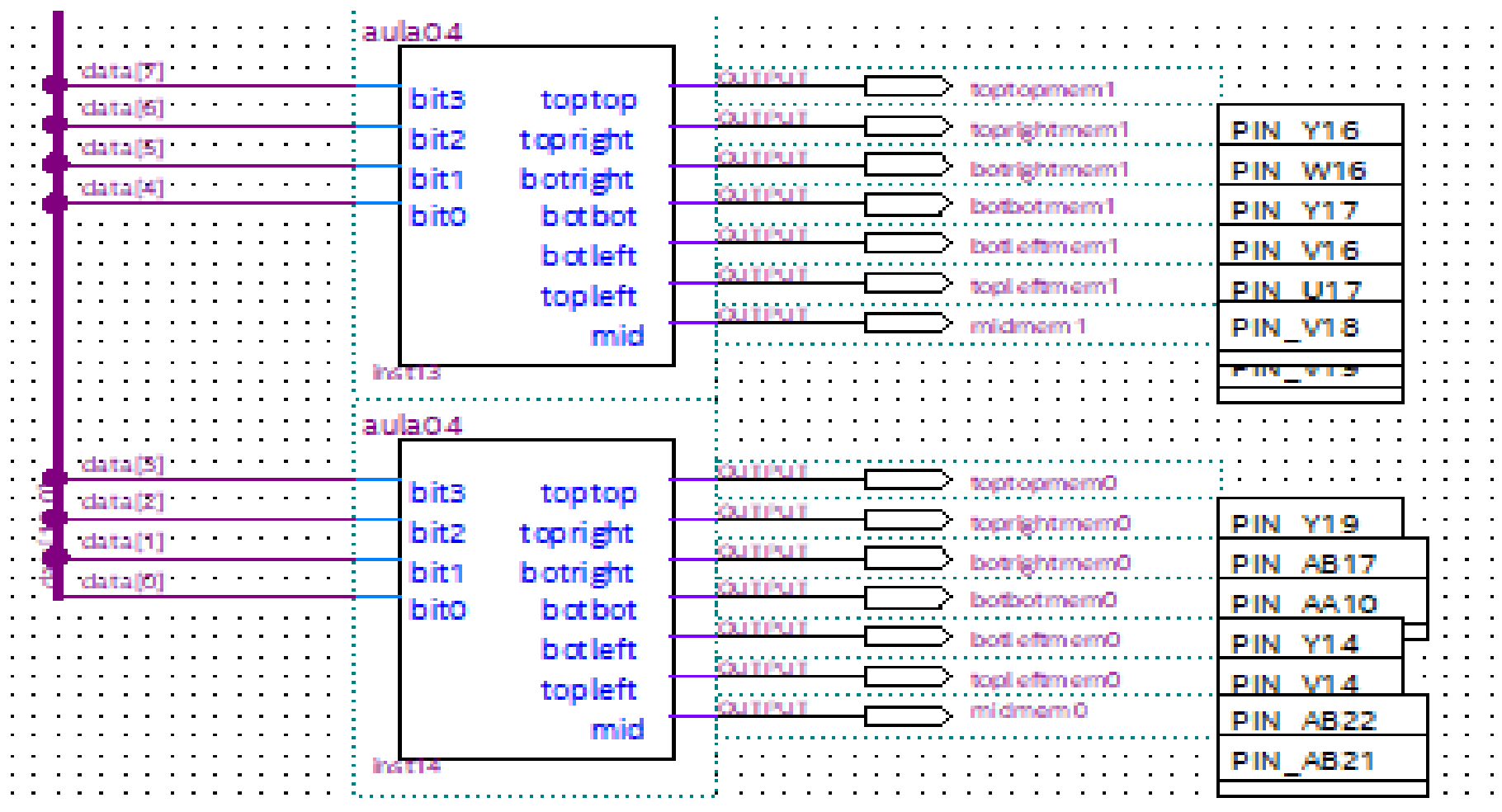
Abaixo seguem imagens do processador finalizado.



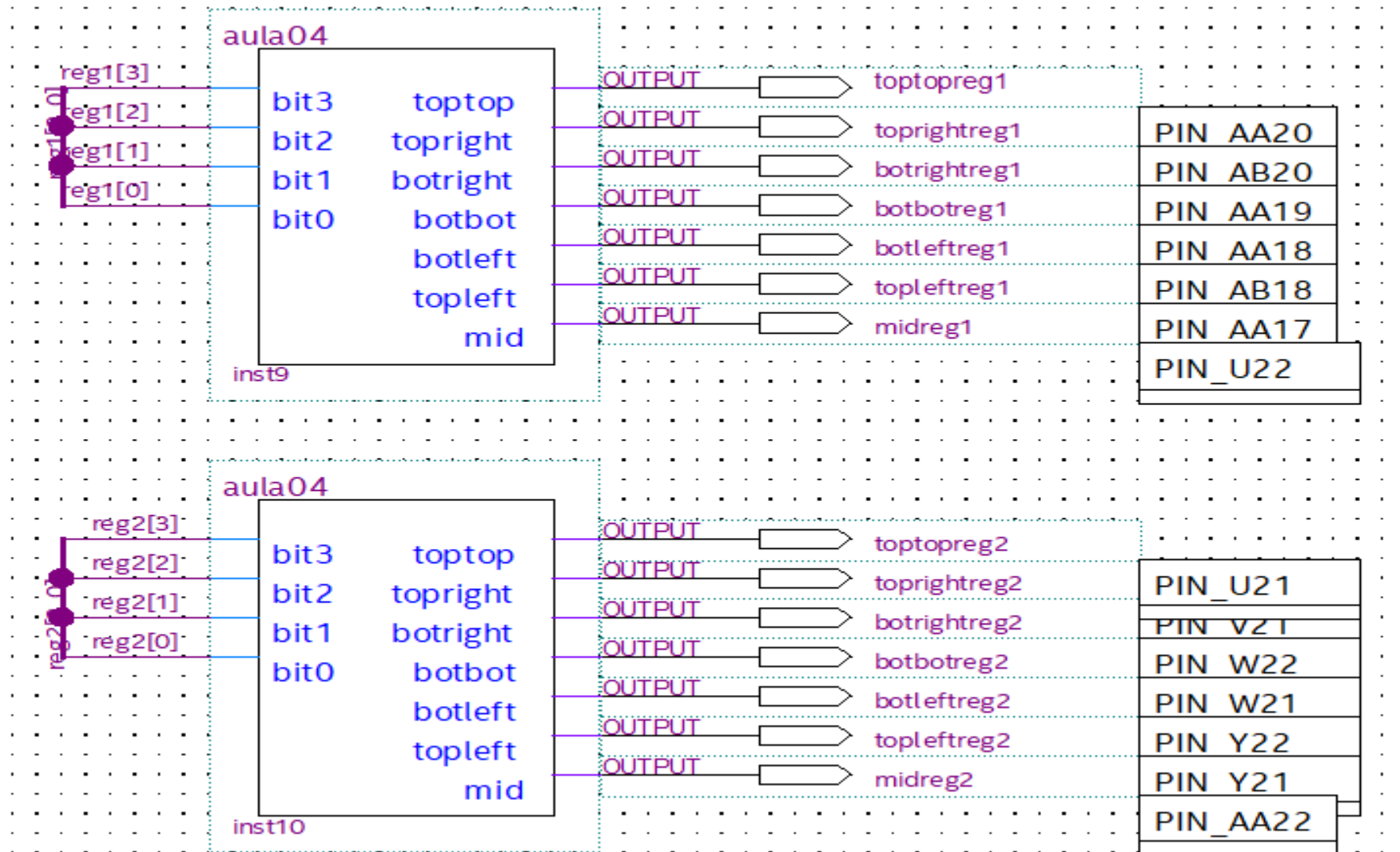
Clock, Contador e ROM



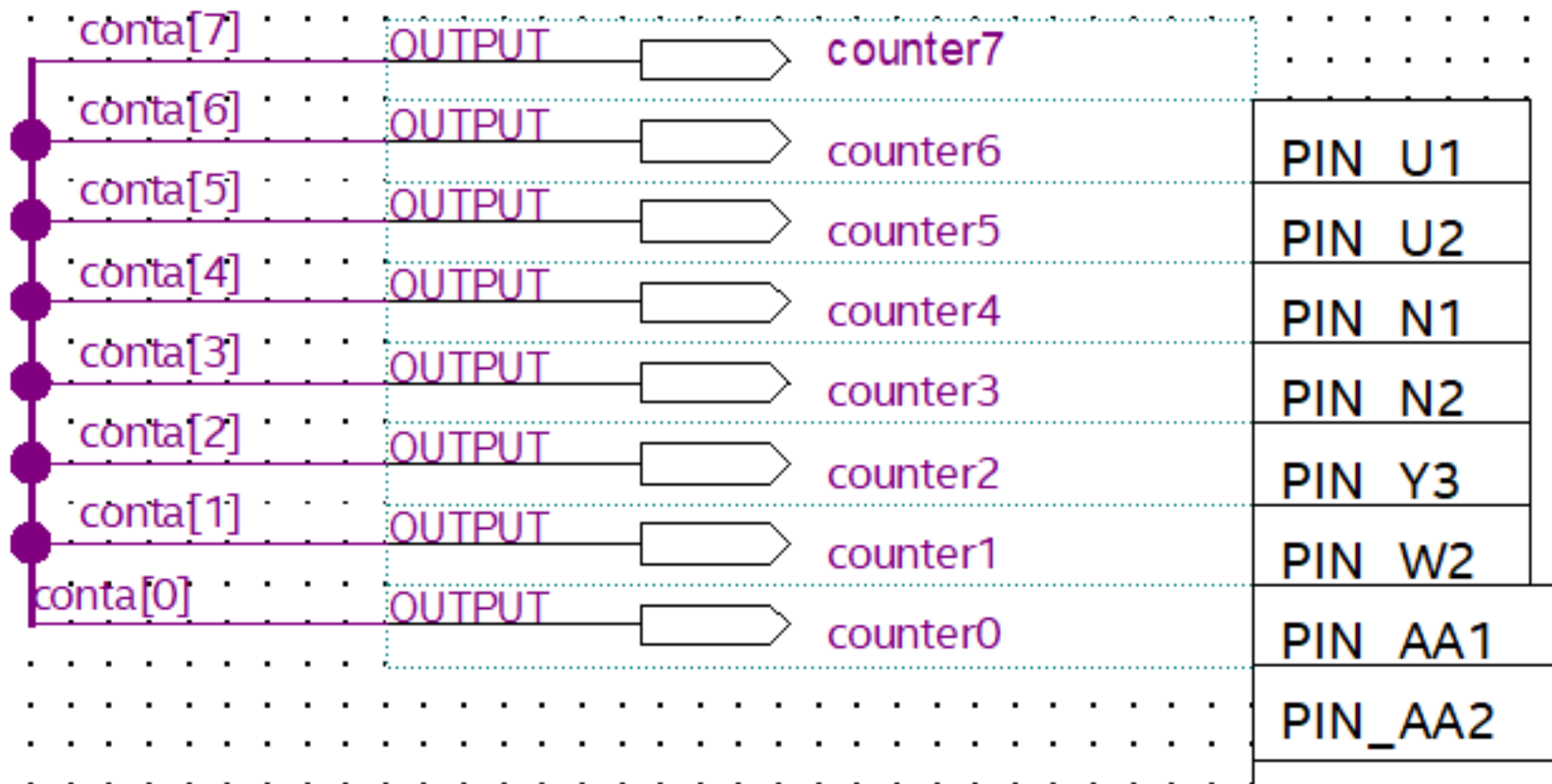
Unidade de Controle, ULA, dois registradores



Displays ROM



Diplays Registradores



Leds Contador