UNIVERSIDADE FEDERAL DA BAHIA
ESCOLA POLITÉCNICA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
ENGG56 – PROJETO DE CIRCUITOS INTEGRADOS DIGITAIS

Leon Santos

Projeto 01: Protocolo ACK/REQ

Salvador - BA - Brasil
2022

SUMÁRIO

1. Introdução

O presente trabalho consiste na elaboração em linguagem Verilog de módulos *Sender* e *Receiver* que utilizem o protocolo *Request/Acknowledgement*. Para tal propósito, foram utilizados os softwares Intel Quartus Prime 20.1 e ModelSim - Intel FPGA Starter Edition 2020.1.

O modelo de comunicação proposto é mostrado no diagrama da Figura 01. O módulo sender é composto por 3 sub-módulos: Sender, Sender Control e Sender Memory. De modo semelhante, o módulo sender é composto por 3 sub-módulos: Receiver, Receiver Control e Receiver Memory.

Figura 01: Componentes do Ecossistema ROS.

Sender_Control deverá enviar sinais de controle Clock, Reset e Transmit para Sender, recebendo deste o sinal Ready. Já o Receiver_Control deverá enviar sinais de controle Clock e Reset para Receiver, recebendo deste o sinal Ready.

O Sender_Control, ao ativar o sinal Transmit por 3 pulsos de clock, solicita que o módulo Sender envie os dados para o receptor. Quando detecta a ativação, o Sender envia todo o conteúdo de Sender_Memory para o Receptor, obedecendo o protocolo Rec/Ack. Após a transferência, o Sender ativa o sinal Ready por 2 pulsos de clock.

O Receiver recebe dados do Sender. Cada dado é escrito consecutivamente na memória de Receiver_Memory. Quando sua memória é totalmente preenchida, o ponteiro indicador da posição de memória retorna ao início e o sinal Ready é ativado por 2 pulsos de clock. Isso sinaliza para o módulo Receiver_Control que os dados foram recebidos.

Sender_Memory e Receiver_Memory são memórias SRAM, com profundidade de 16 endereços e largura de 16 bits. Possuem portas de entrada DataIN, Address, ReadEnable e WriteEnable, e porta de saída DataOut.

Para os módulos de memória foram utilizados IPs fornecidos no próprio programa Quartus. Durante o desenvolvimento do projeto foi importante a atenção para as temporizações inerentes à leitura e escrita em memórias voláteis SRAM.

O código-fonte dos projetos está disponível em https://github.com/leon-ufba/PCID. Os arquivos de simulação, inclusive os com extensão *.do (wave.do), estão disponíveis nos respectivos diretórios ./ModelSim de cada projeto.

2. Diagrama de blocos

A seguir são mostrados os diagramas de blocos de cada módulo.

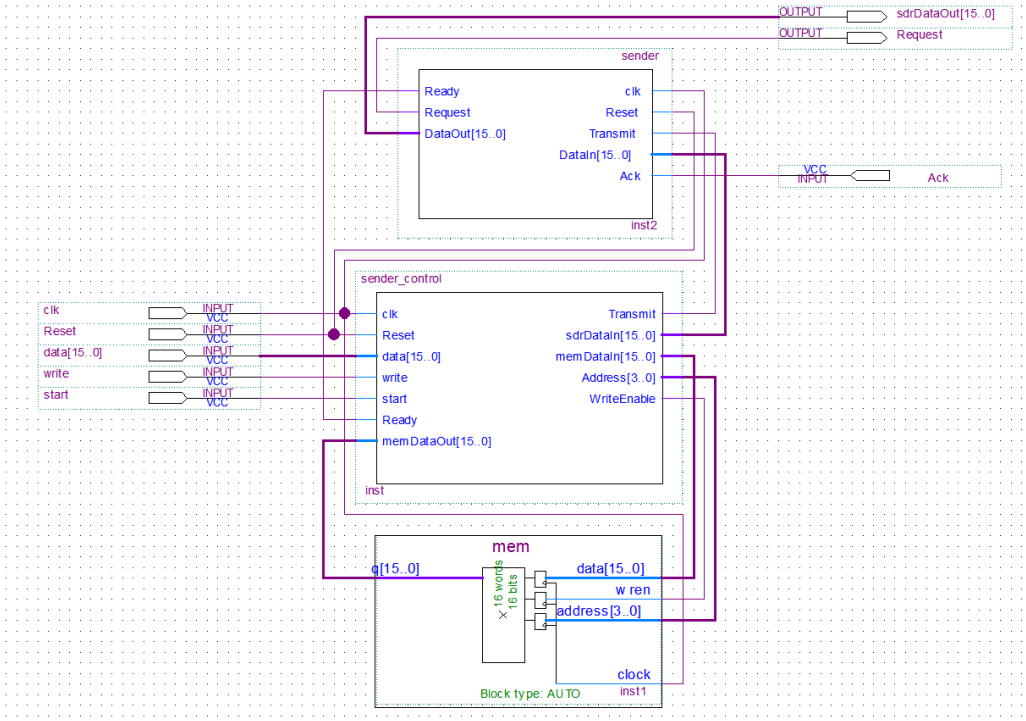

Figura 05: Diagrama de Blocos do módulo Sender.

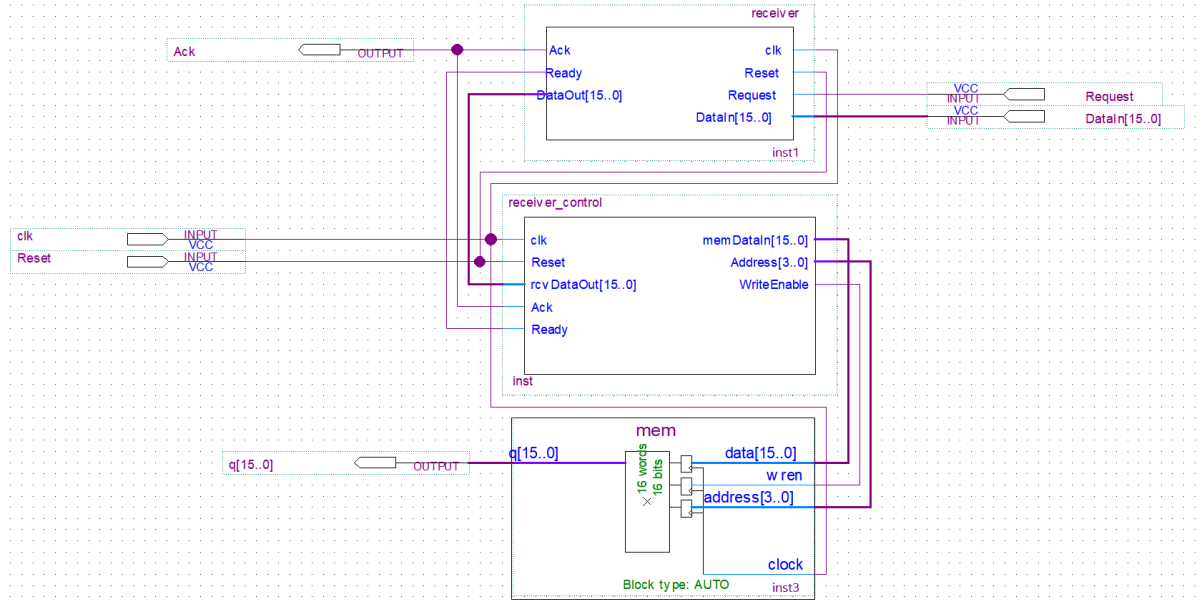

Figura 05: Diagrama de Blocos do módulo Receiver.

3. Máquinas de Estados Finitos

Para cada sub-módulo desenvolvido foram definidas as máquinas de estados finitos.

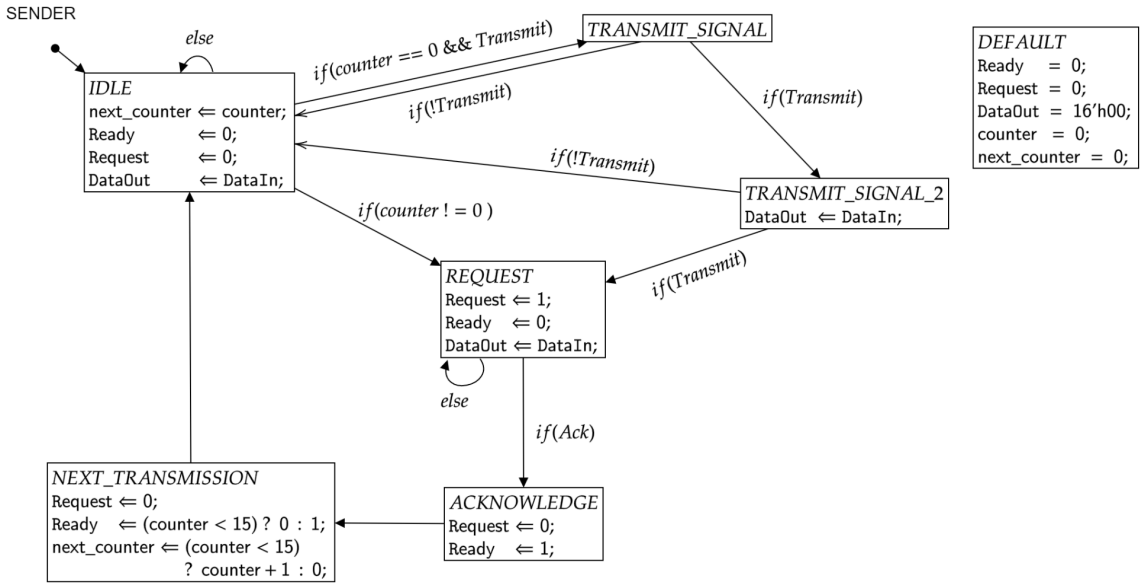

Figura 02: FSM do módulo Sender.

A Figura 02 mostra a FSM do módulo Sender. O módulo inicia no estado IDLE. Caso seja o início da transmissão, o módulo irá para a sequência de estágios: TRANSMIT_SIGNAL e TRASMIT_SIGNAL_2; sendo necessário que o sinal Transmit permaneça ativo por 3 pulsos de clock para que se chegue ao estado REQUEST. Das vezes subsequentes ao início da transmissão, isto é, a partir do segundo endereço de memória, passa-se diretamente do estado IDLE para o estado REQUEST. A máquina permanece no estado REQUEST até receber o sinal Ack, quando passa para o estado ACKNOWLEDGE. Após isso, ela chega ao estado NEXT_TRANSMISSION que compara o contador com o tamanho limite de memória para poder zerar o ponteiro em caso de estouro. Em seguida retorna ao estado IDLE.

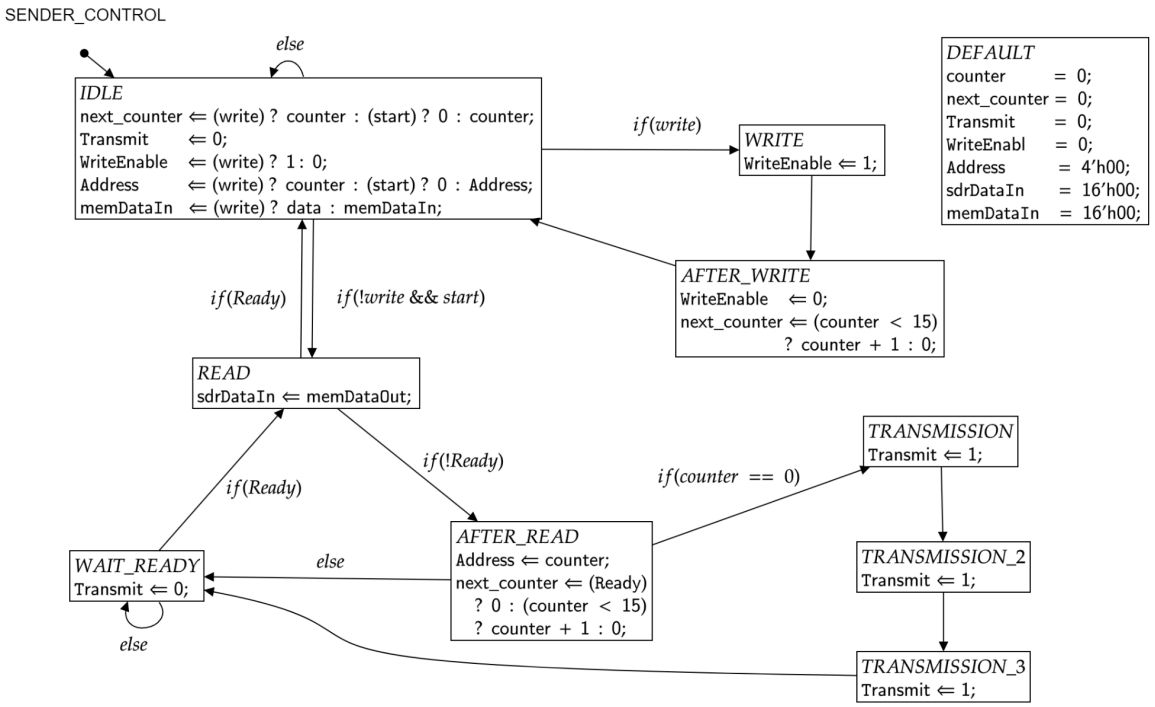

Figura 03: FSM do módulo Sender Control.

A Figura 03 mostra a FSM do módulo Sender Control. O módulo inicia no estado IDLE. Caso seja enviado o sinal de write, o módulo irá para o estágio WRITE que solicita que o módulo de memória guarde o dado no endereço, ambos já indicados no estágio IDLE. Essa opção se deu para garantir que o dado e o endereço estivessem corretos no clock anterior ao de salvamento. Em seguida a máquina vai para o estado AFTER_WRITE, que desativa o sinal writeEnable da memória. Após isso, o módulo retorna ao estado IDLE.

Alternativamente, caso não tenha recebido o sinal de write, mas sim o sinal de start, a máquina sai do estado IDLE para o estado READ, onde será realizada a leitura da memória no endereço já indicado no estágio IDLE. Depois o módulo segue para o estágio AFTER_READ que será responsável por incrementar o contador de endereços. Caso seja o início da transmissão, o Sender Control irá para a sequência de estados TRANSMISSION, TRANSMISSION_2 e TRANSMISSION_3; que permitem que o sinal Transmit seja enviado por 3 pulsos de clock para o sub-módulo Sender. Caso a transmissão já esteja iniciada parte-se do estado AFTER_READ para o WAIT_READY, que aguarda o sinal Ready. Então, retorna-se ao estado READ, que verificará se há sinal Ready pelo segundo pulso de clock, em caso negativo, continua-se a transmissão. Caso seja verdadeiro, a transmissão é encerrada e retorna-se ao estado IDLE.

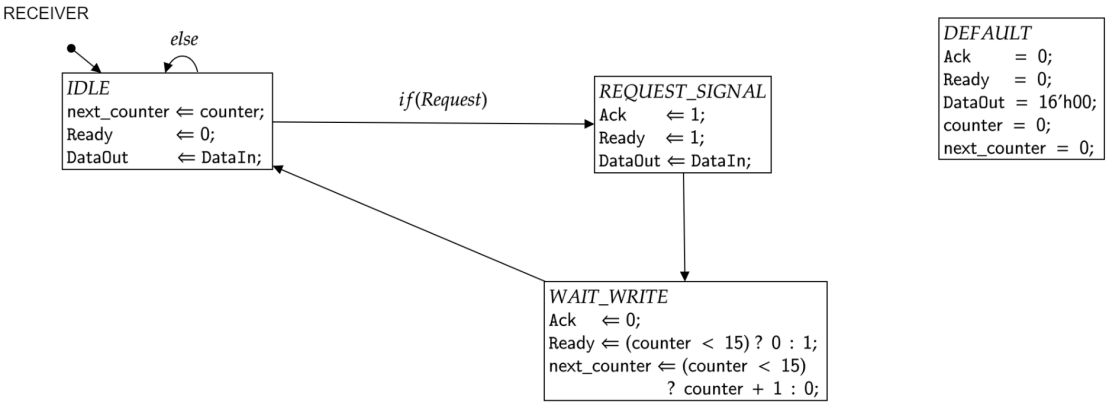

Figura 04: FSM do módulo Receiver.

A Figura 04 mostra a FSM do módulo Receiver. O módulo inicia no estado IDLE. Caso seja enviado o sinal de Request, o módulo irá para o estágio REQUEST_SIGNAL que captura os dados enviados pela Sender e indica ao Receiver Control que houve recebimento de 1 dado, ativando o sinal Ready. Em seguida a máquina vai para o estado WAIT_WRITE que espera um pulso de clock para a escrita na memória, incrementa o contador de endereços, desativa o sinal Ack e apenas mantém o sinal Ready ativo pelo segundo pulso de clock se for o final do processo de recebimento dos dados. Por fim, retorna-se ao estado IDLE.

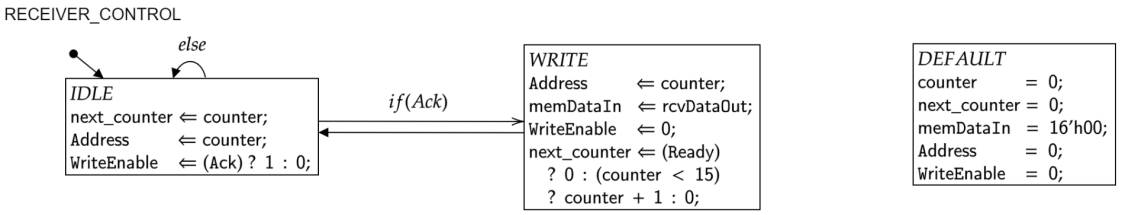

Figura 05: FSM do módulo Receiver Control.

A Figura 05 mostra a FSM do módulo Receiver Control. O módulo inicia no estado IDLE. Caso seja ativado o sinal Ack, o módulo irá para o estágio WRITE que irá operar a escrita na memória. Depois retorna-se ao estado IDLE.

4. Testbenches

As simulações para verificação do bom funcionamento dos módulos foram realizadas no ModelSim. Destaca-se que tanto o Sender, quanto o Receiver tiveram o comportamento desejado. Os arquivos de testbench são sender_ic_tb.v e receiver_ic_tb.v.

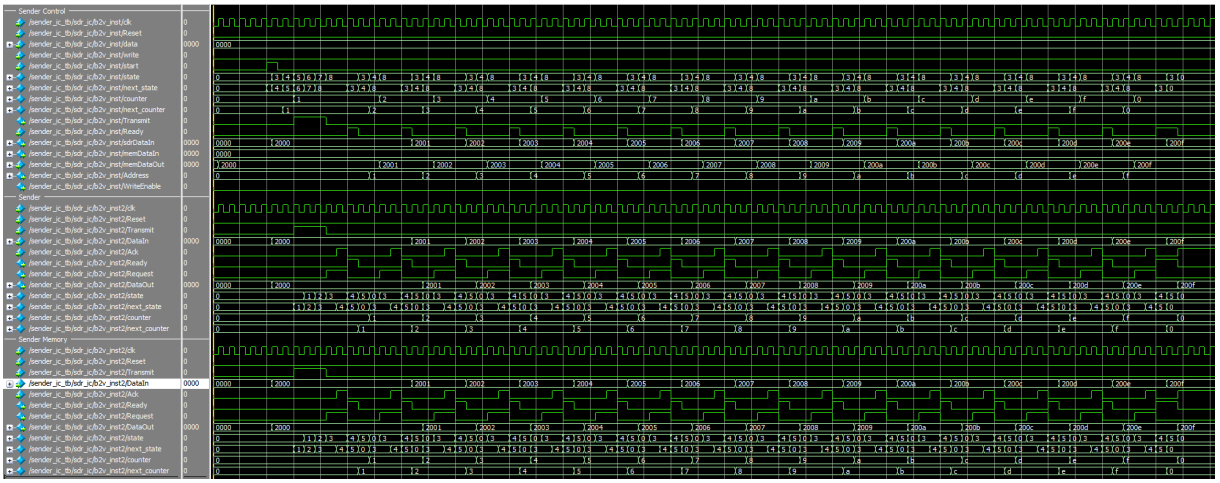

Figura 02: Waveform do módulo Sender.

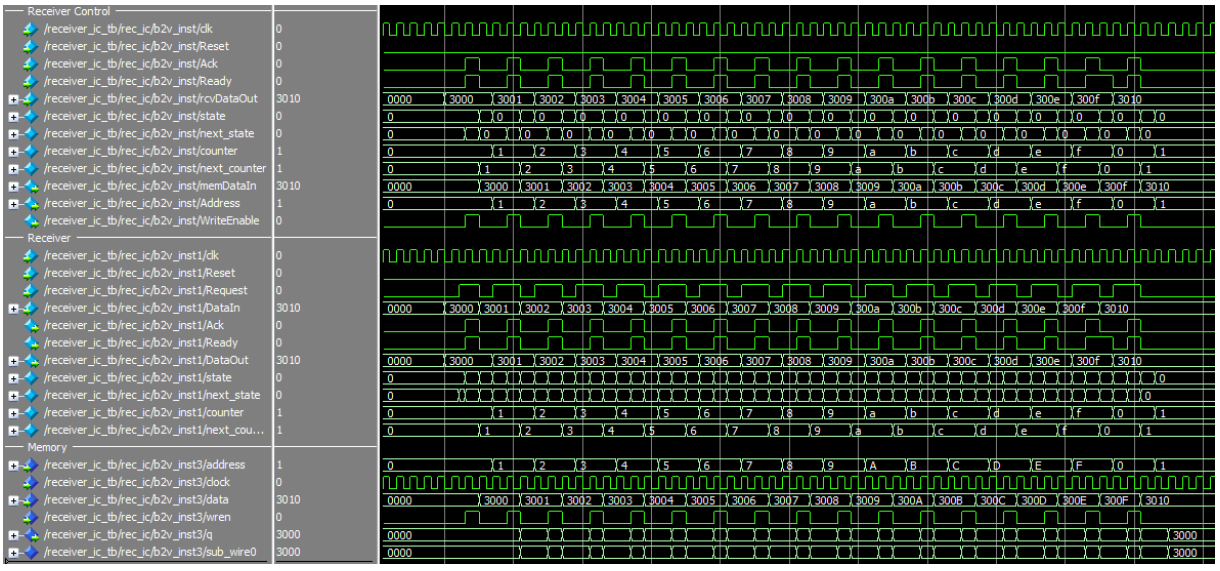

Figura 02: Waveform do módulo Receiver.

Para uma melhor visualização dos dados presentes no Receiver Memory, foi implementado um módulo de memória RAM não IP para testes. O testbench utilizando essa memória está presente na Figura 03 e mostra que os dados foram salvos com sucesso.

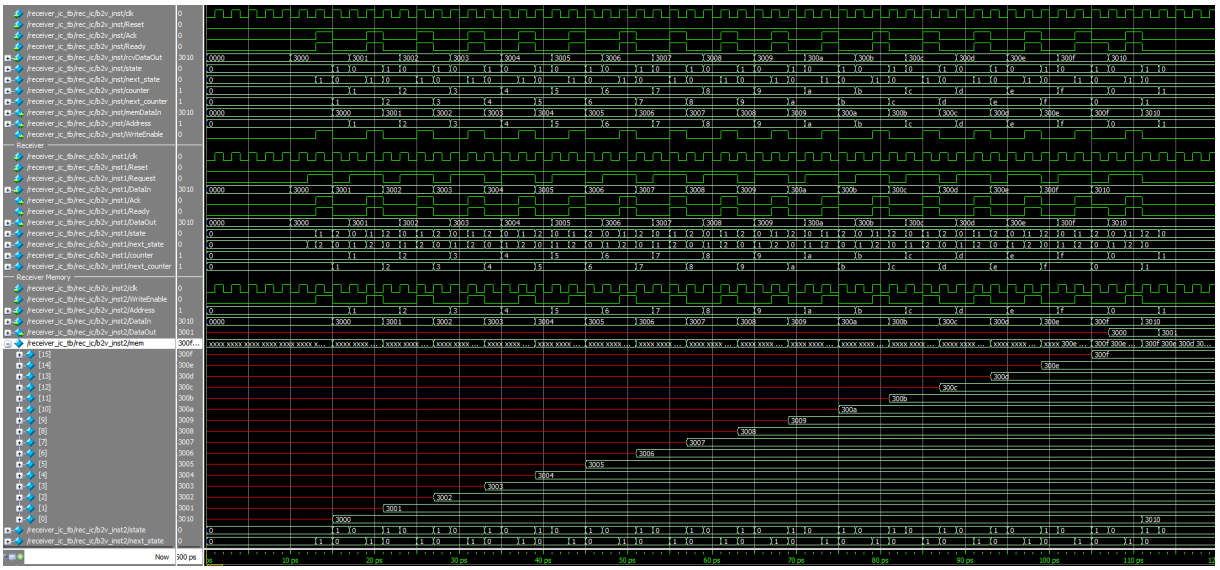

Figura 02: Waveform do módulo Receiver com Memória não IP.

Além da exibição Waveform, os testbenches também contam com impressão textual dos sinais com uso da diretiva $monitor.

5. Conclusão

O presente projeto cumpriu o objetivo proposto e permitiu um maior aprendizado das ferramentas de desenvolvimento de hardware como o Quartus e o ModelSim. Além disso, aprimoraram-se habilidades de codificação com linguagem Verilog.

Referências Bibliográficas

ALTERA. DE2-115 User Manual: World Leading FPGA Based Products and Design Services. [S. l.: s. n.], 2003-2013. Disponível em: <https://www.terasic.com.tw/cgi-bin/page/archive_download.pl?Language=English&No=502&FID=cd9c7c1feaa2467c58c9aa4cc02131af>. Acesso em: 21/10/22.