TUIU.TI
REVISTA TECNOLOGIA DA INFORMAÇÃO
Universidade Tuiuti do Paraná

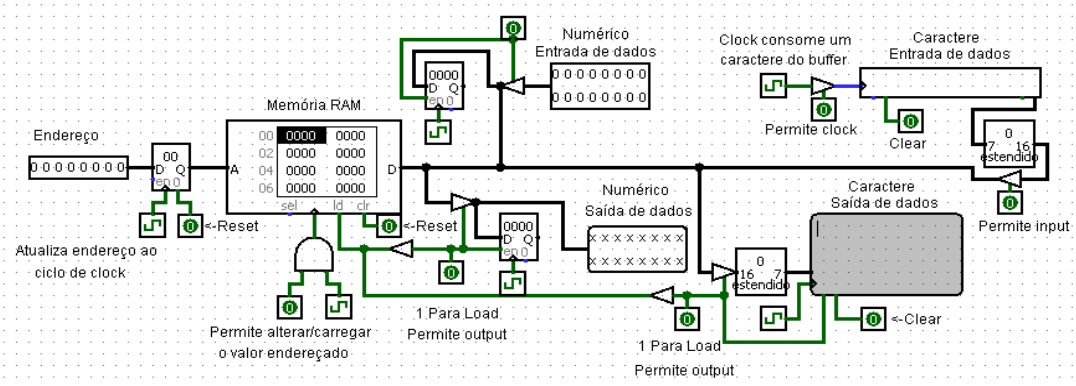
SIMULADOR DE MEMÓRIA RAM

Gabriel Dalabrida Petry Martarello[1]
Gabriel Mendes Alves de Lima[2]
João Pedro Ospedal dos Santos[3]
Renan Nagano[4]
Victor Hugo Piontkievitz da Cruz[5]

INTRODUÇÃO

Este estudo apresenta um Simulador de Memória RAM (*Random Access Memory*), no qual um valor pode ser armazenado na memória no endereço informado, como também pode ser retirado da memória e apresentado ao usuário.

A implementação do simulador foi realizada por meio de um circuito usando o software Logisim. O seu funcionamento se baseia em guardar uma informação em uma posição de memória, informando o seu endereço (operação STORE), e também a leitura da informação no endereço informado da memória (operação LOAD). O Circuito Simulador de Memória RAM pode ser visto na Figura 1.



[1] gabriel.martarello@utp.edu.br
[2] gabriel.lima3@utp.edu.br
[3] jpospedal25@gmail.com
[4] renan.nagano@utp.edu.br
[5] victor.cruz@utp.edu.br

Figura 1: Circuito Simulador de Memória RAM. Fonte: Os próprios autores.

A entrada de dados pode ocorrer de duas formas, assim como as saídas. Uma entrada é para números em forma de bits, junto com sua respectiva saída, e a outra é para caracteres pelo *buffer* do teclado, junto com a saída em uma “tela” TTY.

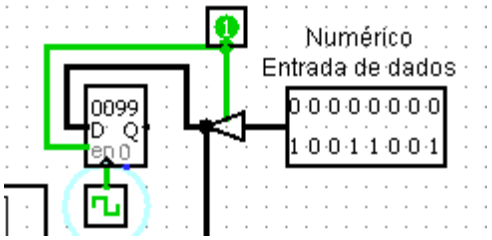


Figura 2: Exemplo de entrada de dados binários. Fonte: Os próprios autores.

No exemplo apresentado na Figura 2, o número 99 hexadecimal, que é representado pelos bits 10011001, é mostrado dessa forma no registrador e na memória.

No circuito existe uma entrada com o rótulo “Endereço” que está ligada a um registrador que, por sua vez, está ligado à memória RAM. É por esse caminho (pino) que será endereçado o valor que se deseja ser acessado na memória, como pode ser visto na Figura 3. Ele servirá para acessar a memória, e somente é atualizado quando o ciclo de *clock*, que está ligado ao registrador, for atualizado. O botão ao lado do *clock* é para limpar o endereço da memória em que está registrado. Quando o registrador for atualizado, a memória RAM também irá atualizar o endereço procurado.

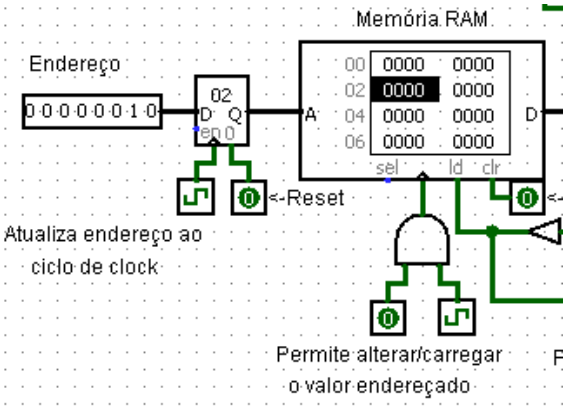


Figura 3: Registrador de Endereço de Memória e Memória RAM. Fonte: Os próprios autores.

A Figura 3 evidencia que o endereço 2 foi acessado pela memória RAM. No circuito existe um *clock* ligado a uma porta AND. Esse *clock* permite que a memória opere, tanto para armazenar um valor no endereço (comando STORE), quanto para carregar um valor no endereço (comando LOAD). Porém, para não ter risco de sobreposição indesejada da memória, existe um pino ligado a ele pela porta AND, para permitir sua operação. Esse pino deve ser ligado quando a memória necessitar ser usada. Se for ligado (valor 1), o sinal de *clock* poderá operar nessa memória.

Na memória RAM existe também o barramento por onde os dados trafegam. Ele serve tanto como entrada quanto para saída de dados. Caso a entrada ld, que significa *load*, esteja com o sinal 1, ou sinal nível alto, o *load* será ativado, e então permitirá que os dados localizados no endereço acessado saiam da memória, podendo ser mostrados por uma das saídas de dados. Caso esteja com o sinal 0, ou nível baixo, o comando store pode executando, fazendo com que os dados recebidos pela entrada sejam armazenados no endereço informado.

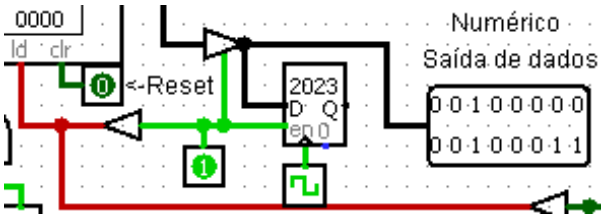


Figura 4: Exemplo de saída de dados. Fonte: Os próprios autores.

A imagem na Figura 4 mostra a saída de dados numéricos (bits). Existe um pino que permite os dados passarem pelo *buffer* controlado, e também, está ligado à entrada “ld” da memória RAM, que quando em 1 permite o carregamento de dados na saída. Os dados que passarem, ao sinal *clock*, serão direcionados para o registrador e para a saída.

Mais à direita há a entrada de caracteres. Apenas um caractere pode ser armazenado por endereço. Uma vez que o dado inserido está em código ASCII, o caractere “b” é armazenado como “0062” na memória RAM, como mostrado na Figura 5.

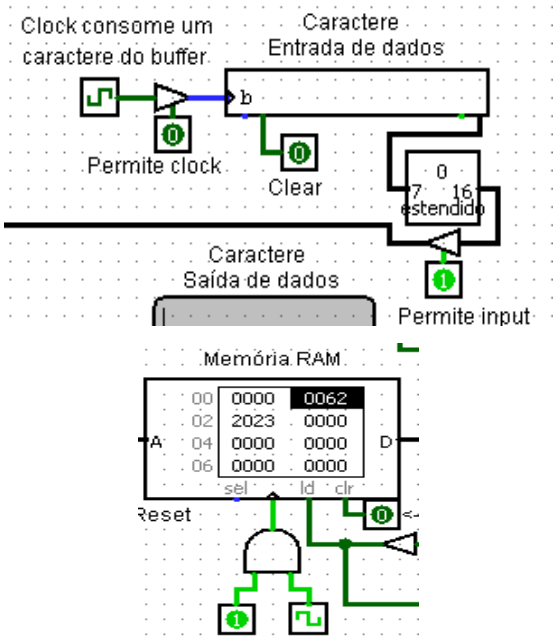


Figura 5: Entrada de Caractere de Dados. Fonte: Os próprios autores.

EXEMPLO DE UTILIZAÇÃO DO CIRCUITO

No exemplo mostrado na Figura 6, o valor “a”, que foi informado na entrada de caracteres, é inserido na memória, no endereço “2”. O pino serve para permitir o dado passar para o *buffer* que está ligado.

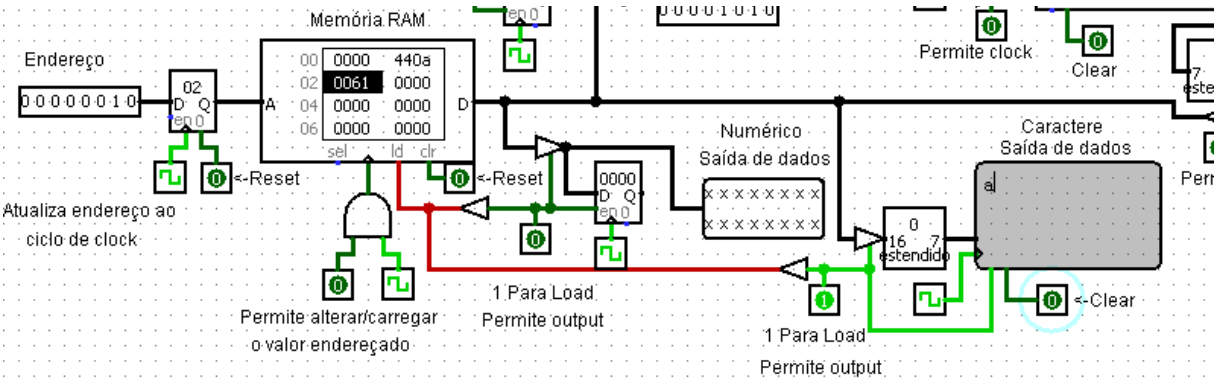


Figura 6: Exemplo Utilização do Circuito Simulador de Memória RAM. Fonte: Os próprios autores.

Por fim, após desligar os pinos necessários e permitir a saída de caracteres pelo pino, o valor endereçado “a” é mostrado na tela TTY. Enquanto o pino estiver ligado, o valor “a” continuará a ser mostrado na tela.

A memória tem largura de 8 bits para o endereçamento e 16 bits para os dados, mas pode ter suas propriedades alteradas, se desejado, bastando mudar os bits nos pinos de entrada e saída com seus *buffers*, registradores e extensores.

CONCLUSÃO

Foi construído um circuito que simula uma memória RAM, no qual o usuário pode escolher o tipo de dado que deseja armazenar na memória, sendo um número, que pode ser recebido como bit ou caractere, para assim guardá-lo em um endereço da memória informado em bits. Os números são informados nos pinos, e os caracteres em um buffer chamado “teclado”. Se a memória estiver ligada, o valor é guardado, e eles podem ser mostrados em uma das saídas de dados se ligado corretamente.

Assim, o circuito foi demonstrado neste estudo, bem como uma análise do circuito, com explicações de como ele funciona, em cada uma de suas partes.

REFERÊNCIAS

OLIVEIRA, Atila. Memória - Logisim - Parte 1. Youtube, 13/08/2021 . Disponível em : https://www.youtube.com/watch?v=_xuG9epNW3k . Acesso em 11/2023.

MARQUES FILHO, Sergio Luiz. Materiais da Disciplina de Arquitetura e Organização de Computadores. Universidade Tuiuti do Paraná, 2023. Curitiba.

NeutronNick11. Logisim part 10:RAM. Youtube. Disponível em: https://www.youtube.com/watch?v=lk5iSGwzmgY . Acesso em 11/2023.