**FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”**

MANTENEDORA DO CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA – UNIVEM

**Bacharelado em Ciência da Computação - Organização e Arquitetura de Computadores - 5° Termo**

# Prof. Ildeberto de Genova Bugatti

# Projeto 01

# Subsistemas de Memória (RAM Estática de 2Kx32)

**Grupo 4**

**Fillipo Martinez de Almeida – RA: 591491**

**Luís Guilherme Cantos Sobral – RA: 587737**

**Thales Sakatauskas Brito – RA: 592609**

1. **Objetivos do trabalho**

Objetivo deste projeto é apresentar os conceitos sobre a classificação, organização, aplicação e construção de um subsistema de memória, essa apresentação é feita utilizando a associação serie paralela entre componentes eletrônicos para assim fazer a arquitetura correta de uma memória semicondutora funcional.

1. **Descrição Geral do subsistema implementado:**

O subsistema é constituído por componentes eletrônicos para que os mesmos possam fazer funções essenciais, como um sistema de memória tem dois tipos de funções então se faz necessário se ter um componente que tem uma função dupla no sistema, este componente é denominado de componente de leitura e escrita, pois ele é responsável por gravar ou exibir os dados, este componente precisa tanto de uma entrada de dados como uma saída de dados, então se faz necessário ter registradores desse tipo, para que se consiga resgatar os dados armazenados na memória para exibi-los é necessário um conversor, para se ter acesso a memória tanto para gravar ou para exibir os dados na mesma se precisa ter um sistema que saiba onde fica cada endereço de memória, porém uma memória grande nada mais é do que pequenas memórias associadas tanto em serie quanto em paralelo para aumentar sua capacidade tanto de espaço de bytes quanto de bits.

Na associação em série o conjunto de componentes são colocados em uma ordem que a quantidade de memória aumenta quando se multiplica sua memória atual pela quantidade de componentes de memórias colocados em serie (Ex: 2 componentes de memória de 32Bx8bits associado em serie teria uma memória no final de 64Bx8bits), por outro lado a associação em paralelo é formada por uma sequência de componentes eletrônicos que aumenta os bits na memória (Ex: 4 componentes de 32Bx8bits associado em paralelo se transforma em uma memória de 32Bx32bits), estes elementos eletrônicos precisam ser iguais para que se possa ter uma organização de fácil entendimento.

O sistema em sua última fase conta com 4 memórias de 2kBx8 em cada, após ser feita a associação em paralelo estas memórias se transformam em uma única memória de 2kBx32, para se chegar em uma memória de 2kBx8 bits foram feitas associações em serie começando com um componente de 32B, o qual sempre se associava em conjuntos de 4 componentes iguais de memória, fazendo com que o sistema aumente em uma progressão geométrica de fator 4, fazendo assim com que o sistema comece com 4 componentes de 32Bx8 saindo da primeira fase com um componente de 128Bx8, passando para a sua segunda fase que se têm 4 componentes de 128Bx8 se tornando em um único componente de 512Bx8, seguindo está linha na sua terceira fase começa com a associação de 4 componentes de 521Bx8 e terminando com um componente de 2kBx8, até a terceira fase todos os elementos são associados em serie, porém na quarta fase se tem 4 elementos de 2kBx8 associado em paralelo fazendo com que ele tenha um componente de 2kBx32, fazendo com que o projeto seja realizado em 4 partes.

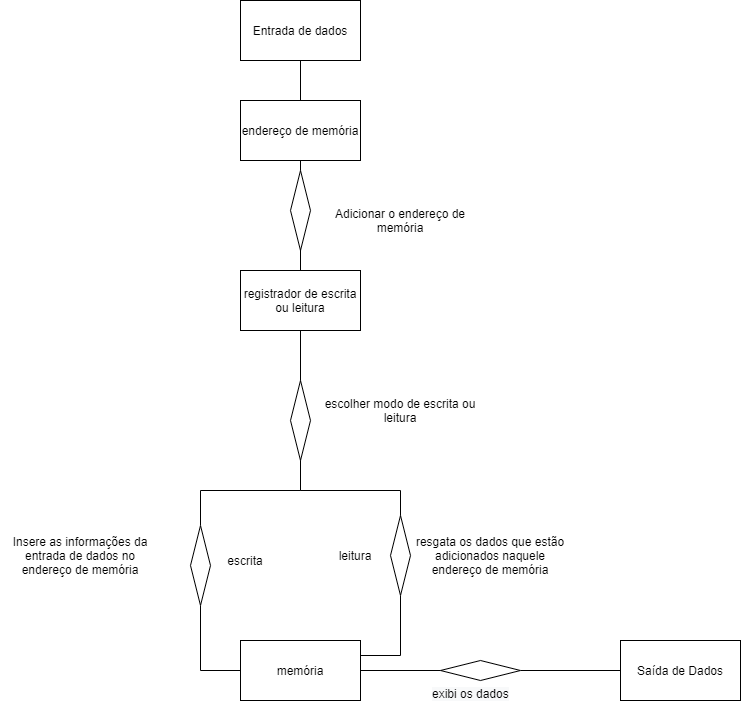


Figura 1 - Organização lógica de um subsistema de memória

1. **Descrição dos módulos do subsistema:**

Quando se tem um sistema de memória tende a se utilizar de duas funções (leitura e escrita), relacionado a isso temos componentes que são projetados para que se tenha uma especificação deste sistema, formando assim blocos, os mesmos são arquitetados para que se tenha funções específicas, o principal componente que se usa neste sistema é o registrador de endereçamento de memória, este objeto é ligado a encontrar o lugar da memória onde será armazenado ou onde terá seu dado capturado para exibição, e isso é diretamente a ligado a função da memória, sem esta parte não é possível se ter uma ordenação ou até mesmo um processo de busca por informações, praticamente o registrador de memória armazena em sua pequena memória um endereço específico do sistema, este endereço fica salvo até o momento que for alterado por outro endereço, outro elemento deste sistema é o registrador de leitura/escrita, este item tem dois estados um alto (leitura) e outro baixo (escrita), assim que um dado é inserido no bloco de endereçamento é analisado o registrador para que se possa saber se terá que gravar ou ler um dado, após identificar o tipo de função que será exercida o sistema busca o endereço dentro da memória se caso o método seja o de escrita, após localizar o espaço da memória será buscado qual foi a entrada de dados e a mesma será inserida naquele endereço, porém se a função for a de leitura o sistema funciona de forma diferente, ela acaba copiando os dados que estão localizados naquele endereço de memória para o registrador de saída de dados, fazendo com que o usuário consiga ver o que está escrito naquele local.

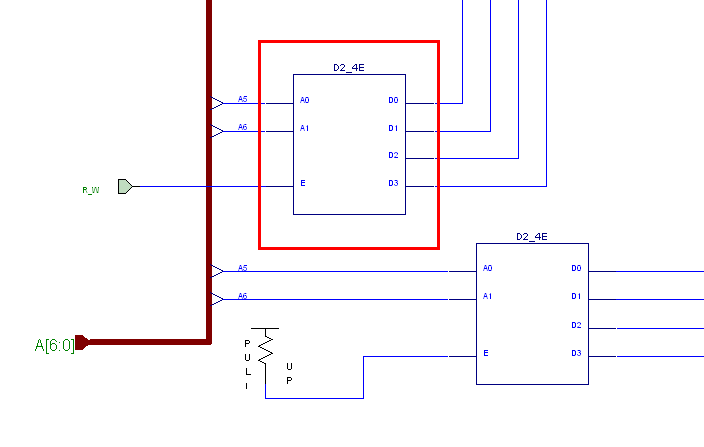


Figura 2 – Line Decoder/Demultiplexer

Está figura denominada como D2\_4E é um decodificador/demultiplexador ele é responsável por interpretar o registrador de escrita e leitura, podendo estar de dois estados como o registrador, porém estes dois estados são de endereço binário de 2 bits, sendo assim tendo 4 saídas de dados, já o pull-up é um resistor responsável por manter o segundo decoder em um estado alto para que sempre possa haver uma saída de dados daquele endereço de memória, porém o estado alto é responsável pela função de escrita então mais a frente deste circuito existe um inversor fazendo com que o estado do decodificador seja baixo ativando a função de leitura de dados, porém mesmo assim acaba tendo um conflito pois na saída dos dados eles tem mais de uma terminação então um buffer de saída vem para unificar essa terminação.

Para que se haja uma troca dos valores nos registradores é necessário utilizar um clock, este é responsável por manter o sistema com uma mudança de estado entre alto e baixo, quando se tem a borda de descida acaba não mudando o valor do registrador de saída, porém com a borda de subida há uma mudança de estado dentro do registrador de saída, essa mudança ocorre pelo fato do registrador de entrada estar com um valor diferente de um registrador de saída, caso o registrador de escrita esteja ativo este valor é inserido na memória, mas se o registrador estiver em estado de leitura ele somente modificar o registrador de saída.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome do Banco** | **Endereço inicial (binario)** | **Endereço final (binario)** | **Endereço inicial (hexa)** | **Endereço final (hexa)** |
| B0 | 000 0000 | 001 1111 | 00 | 1F |
| B1 | 010 0000 | 011 1111 | 20 | 3F |
| B2 | 100 0000 | 101 1111 | 40 | 5F |
| B3 | 110 0000 | 111 1111 | 60 | 7F |

Tabela 1 – Tabela de endereço inicial e endereço final do banco de memória 128Bx8

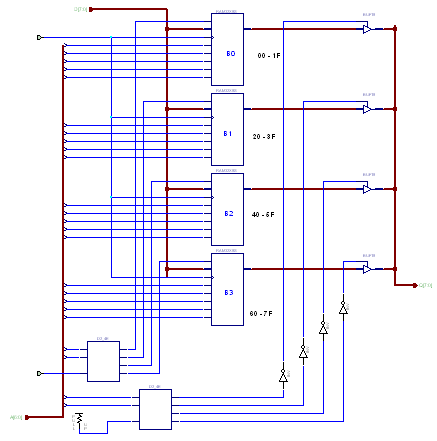


Figura 3 – Imagem de um Sistema de memória de 128Bx8

Podemos verificar as entradas de dados estão vindo das portas A (7 bits) e D (8 bits), com decodificadores para poder determinar se estará escrevendo ou lendo os dados daquele espaço de memória, podemos ver mais abaixo um circuito para que possa retornar todos os dados naquele endereço pela saída de dados denominada como O (8 bits).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dados de entrada (hexadecimal) | Endereço de memoria  (hexadecimal) | R/W | CLCK | Lógica – saída de dados |
| DADOS (00 – FF) | Endereço (00 – 7F) | 1 | 1 | O dado inserido se mantém no registrador de entrada |
| DADOS (00 – FF) | Endereço (00 – 7F) | 1 | 0 | A borda do clock fica em estado baixo, estado que se confere o tipo de função |
| DADOS (00 – FF) | Endereço (00 – 7F) | 1 | 1 | Após ter a subida da borda do clock a função de escrita de dados foi feita no endereço declarado |
| DADOS (00 – FF) | Endereço (00 – 7F) | 0 | 0 | A uma descida da borda do clock o qual confere qual a função será feita |
| DADOS (00 – FF) | Endereço (00 – 7F) | 0 | 1 | No momento da subida da borda do clock o sistema faz uma leitura no endereço solicitado e grava em seu registrador de saída |

Tabela 2 – Tabela Verdade do sistema de 128Bx8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome do Banco** | **Endereço inicial (binario)** | **Endereço final (binario)** | **Endereço inicial (hexa)** | **Endereço final (hexa)** |
| B0 | 0 0000 0000 | 0 0111 1111 | 000 | 07F |
| B1 | 0 1000 0000 | 0 1111 1111 | 080 | 0FF |
| B2 | 1 0000 0000 | 1 0111 1111 | 100 | 17F |
| B3 | 1 1000 0000 | 1 1111 1111 | 180 | 1FF |

Tabela 3 – Tabela de endereço inicial e endereço final do banco de memória 512Bx8

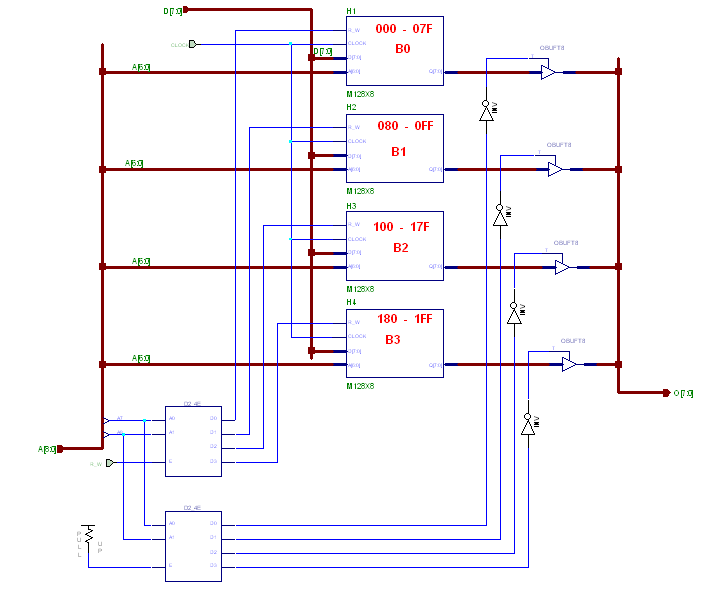


Figura 4 – Imagem de um Sistema de memória de 512Bx8

Podemos verificar as entradas de dados estão vindo das portas A (9 bits) e D (8 bits), com decodificadores para poder determinar se estará escrevendo ou lendo os dados daquele espaço de memória, podemos ver mais abaixo um circuito para que possa retornar todos os dados naquele endereço pela saída de dados denominada como O (8 bits).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dados de entrada (hexadecimal) | Endereço de memoria  (hexadecimal) | R/W | CLCK | Lógica – saída de dados |
| DADOS (00 – FF) | Endereço (00 – 1FF) | 1 | 1 | O dado inserido se mantém no registrador de entrada |
| DADOS (00 – FF) | Endereço (00 – 1FF) | 1 | 0 | A borda do clock fica em estado baixo, estado que se confere o tipo de função |
| DADOS (00 – FF) | Endereço (00 – 1FF) | 1 | 1 | Após ter a subida da borda do clock a função de escrita de dados foi feita no endereço declarado |
| DADOS (00 – FF) | Endereço (00 – 1FF) | 0 | 0 | A uma descida da borda do clock que confere qual a função será feita |
| DADOS (00 – FF) | Endereço (00 – 1FF) | 0 | 1 | No momento da subida da borda do clock o sistema faz uma leitura no endereço solicitado e grava em seu registrador de saída |

Tabela 4 – Tabela Verdade do sistema de 512Bx8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome do Banco** | **Endereço inicial (binario)** | **Endereço final (binario)** | **Endereço inicial (hexa)** | **Endereço final (hexa)** |
| B0 | 000 0000 0000 | 001 1111 1111 | 000 | 17F |
| B1 | 010 0000 0000 | 011 1111 1111 | 200 | 3FF |
| B2 | 100 0000 0000 | 101 1111 1111 | 400 | 57F |
| B3 | 110 0000 0000 | 111 1111 1111 | 600 | 7FF |

Tabela 5 – Tabela de endereço inicial e endereço final do banco de memória 2048Bx8

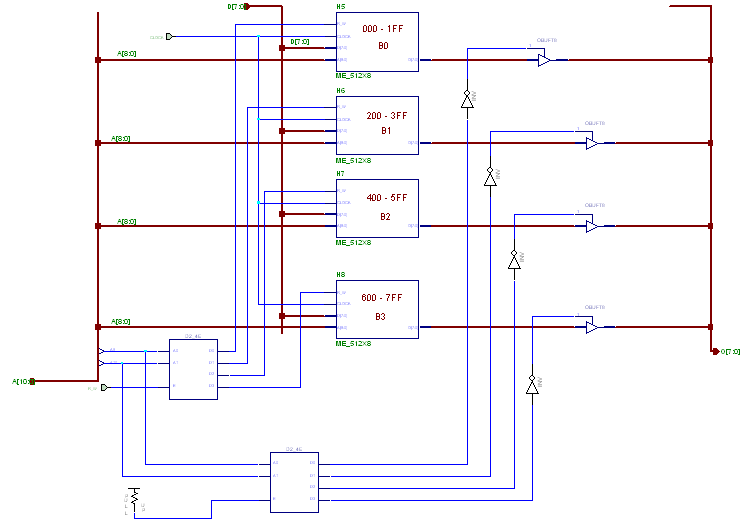


Figura 5 – Imagem de um Sistema de memória de 2048Bx8

Podemos verificar as entradas de dados estão vindo das portas A (11 bits) e D (8 bits), com decodificadores para poder determinar se estará escrevendo ou lendo os dados daquele espaço de memória, podemos ver mais abaixo um circuito para que possa retornar todos os dados naquele endereço pela saída de dados denominada como O (8 bits).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dados de entrada (hexadecimal) | Endereço de memoria  (hexadecimal) | R/W | CLCK | Lógica – saída de dados |
| DADOS (00 – FF) | Endereço (00 – 7FF) | 1 | 1 | O dado inserido se mantém no registrador de entrada |
| DADOS (00 – FF) | Endereço (00 – 7FF) | 1 | 0 | A borda do clock fica em estado baixo, estado que se confere o tipo de função |
| DADOS (00 – FF) | Endereço (00 – 7FF) | 1 | 1 | Após ter a subida da borda do clock a função de escrita de dados foi feita no endereço declarado |
| DADOS (00 – FF) | Endereço (00 – 7FF) | 0 | 0 | A uma descida da borda do clock que confere qual a função será feita |
| DADOS (00 – FF) | Endereço (00 – 7FF) | 0 | 1 | No momento da subida da borda do clock o sistema faz uma leitura no endereço solicitado e grava em seu registrador de saída |

Tabela 6 – Tabela Verdade do sistema de 2048Bx8

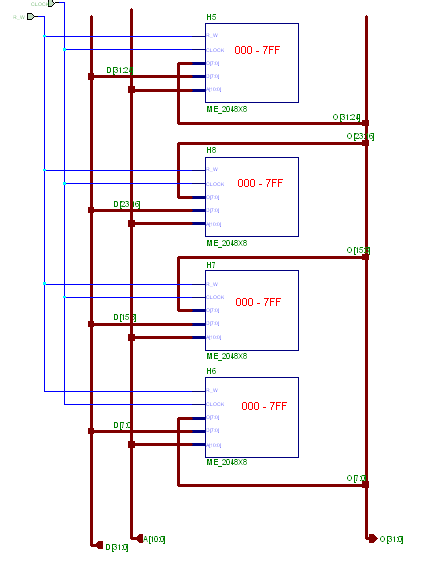


Figura 6 – Imagem de um Sistema de memória de 2048Bx32

Podemos verificar as entradas de dados estão vindo das portas A (11 bits) e D (32 bits), com decodificadores para poder determinar se estará escrevendo ou lendo os dados daquele espaço de memória, podemos ver mais abaixo um circuito para que possa retornar todos os dados naquele endereço pela saída de dados denominada como O (32 bits), nesta parte é necessário fazer uma soma de 8 bits em 8 bits na entrada de cada elemento de entra e saída para que se possa ter a associação em paralelo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dados de entrada (hexadecimal) | Endereço de memoria  (hexadecimal) | R/W | CLCK | Lógica – saída de dados |
| DADOS (  0000 0000  -  FFFF FFFF) | Endereço (00 – 7FF) | 1 | 1 | O dado inserido se mantém no registrador de entrada |
| DADOS (  0000 0000  -  FFFF FFFF) | Endereço (00 – 7FF) | 1 | 0 | A borda do clock fica em estado baixo, estado que se confere o tipo de função |
| DADOS (  0000 0000  -  FFFF FFFF) | Endereço (00 – 7FF) | 1 | 1 | Após ter a subida da borda do clock a função de escrita de dados foi feita no endereço declarado |
| DADOS (  0000 0000  -  FFFF FFFF) | Endereço (00 – 7FF) | 0 | 0 | A uma descida da borda do clock que confere qual a função será feita |
| DADOS (  0000 0000  -  FFFF FFFF) | Endereço (00 – 7FF) | 0 | 1 | No momento da subida da borda do clock o sistema faz uma leitura no endereço solicitado e grava em seu registrador de saída |

Tabela 7 – Tabela Verdade do sistema de 2048Bx32

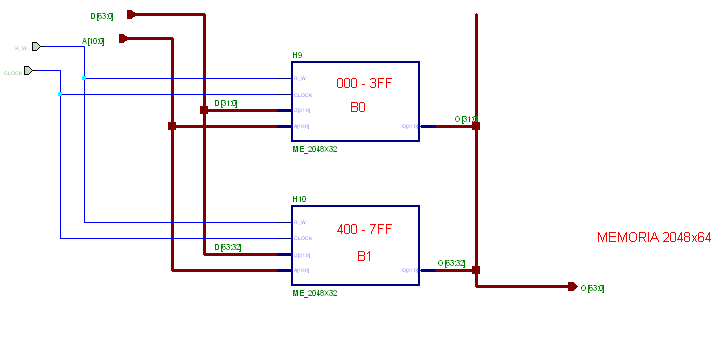


Figura 7 – Imagem de um Sistema de memória de 2048Bx64

Podemos verificar as entradas de dados estão vindo das portas A (11 bits) e D (64 bits), com decodificadores para poder determinar se estará escrevendo ou lendo os dados daquele espaço de memória, podemos ver mais abaixo um circuito para que possa retornar todos os dados naquele endereço pela saída de dados denominada como O (64 bits), nesta parte é necessário fazer uma soma de 32 bits em 32 bits na entrada de cada elemento de entra e saída para que se possa ter a associação em paralelo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dados de entrada (hexadecimal) | Endereço de memoria  (hexadecimal) | R/W | CLCK | Lógica – saída de dados |
| DADOS (  0000 0000 0000 0000  -  FFFF FFFF FFFF FFFF) | Endereço (00 – 7FF) | 1 | 1 | O dado inserido se mantém no registrador de entrada |
| DADOS (  0000 0000 0000 0000  -  FFFF FFFF FFFF FFFF) | Endereço (00 – 7FF) | 1 | 0 | A borda do clock fica em estado baixo, estado que se confere o tipo de função |
| DADOS (  0000 0000 0000 0000  -  FFFF FFFF FFFF FFFF) | Endereço (00 – 7FF) | 1 | 1 | Após ter a subida da borda do clock a função de escrita de dados foi feita no endereço declarado |
| DADOS (  0000 0000 0000 0000  -  FFFF FFFF FFFF FFFF) | Endereço (00 – 7FF) | 0 | 0 | A uma descida da borda do clock que confere qual a função será feita |
| DADOS (  0000 0000 0000 0000  -  FFFF FFFF FFFF FFFF) | Endereço (00 – 7FF) | 0 | 1 | No momento da subida da borda do clock o sistema faz uma leitura no endereço solicitado e grava em seu registrador de saída |

Tabela 8 – Tabela Verdade do sistema de 2048Bx64

1. **Simulação do circuito (Project Manager)**

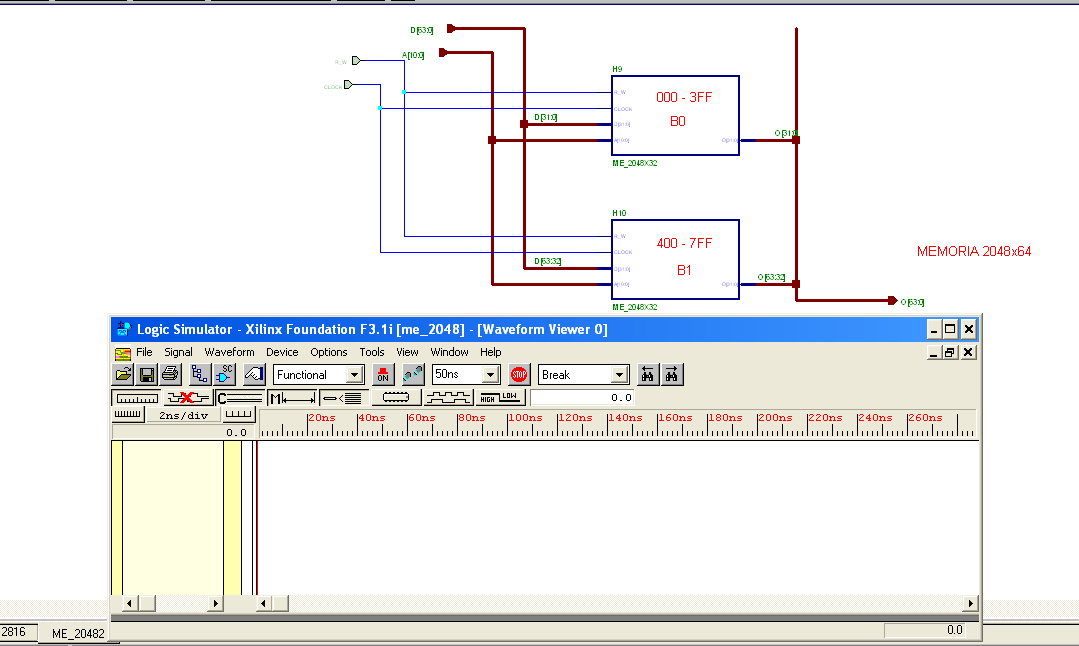


Figura 8 – Imagem do início da simulação de um Sistema de memória de 2048Bx64

Primeiro é necessário inserir sinais para que possamos nos comunicar com este sistema, vamos setar todos os valores de sinais (endereço de memória, entrada de dados, clock, escrever/ler, saída de dados).

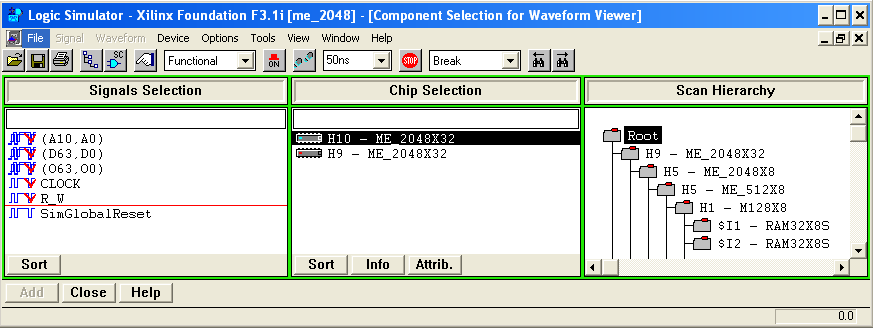


Figura 9 – Selecionando sinais de um Sistema de memória de 2048Bx64

Após adicionar o sinal precisamos adicionar alguns valores para que nosso clock e nosso componente de leitura/escrita possa ficar oscilando entre estado alto e baixo, porém o clock terá uma oscilação constante por isso estaremos adicionando um valor padrão, já o componente de escrita/leitura será ativado pela letra “w”.

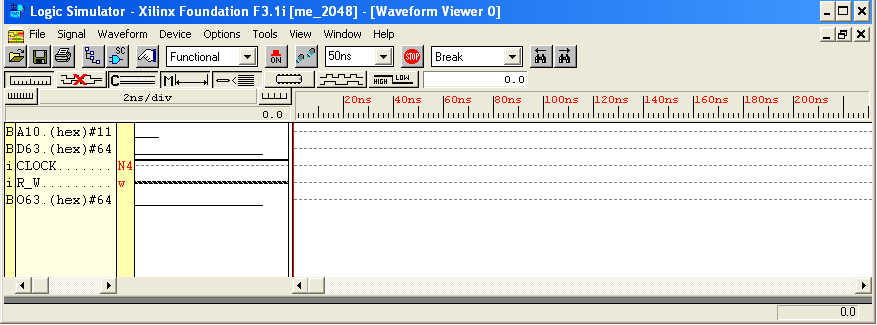


Figura 10 – Adicionando valores de modificação para o clock e o item de leitura/escrita.

Após colocar o valor da borda do componente em estado de leitura, é adicionado um valor no A10 onde é relacionado ao valor do endereço de memória no banco de dados, após escolher o local onde o dado será adicionado, adicionamos o valor deste dado. Após definir os valores se inicia a simulação, após se passar 50ns podemos ver que os valores que foram setados na entrada de dados e endereço de memória já foram adicionados ao sistema, porém podemos analisar que na saída de dados temos o valor zerado, isto é um padrão para que não haja “sujeira” na memória então sempre o espaço de memória vazia se inicia zerado.

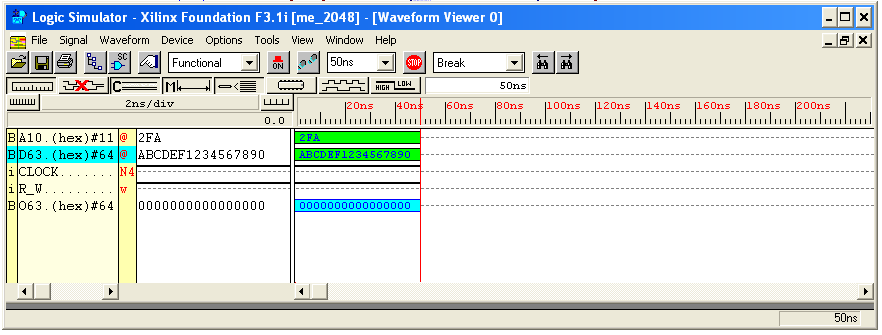


Figura 11 – Registrando os valores após 50ns.

Após a passagem de mais 30ns podemos verificar que a borda do clock acaba abaixando, fazendo com que o sistema procura qual função está sendo pedida pelo sistema, visualizando o componente de escrita/leitura no estado alto, sendo assim ativando a função de escrita de dados na nossa memória.

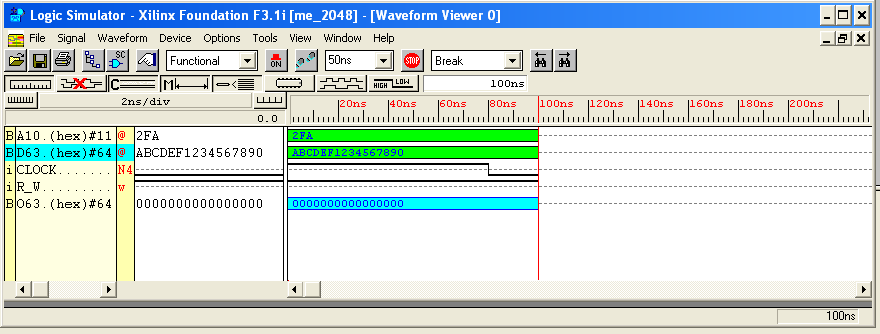


Figura 12 – Registrando os valores após 100ns.

Após a passagem de 160ns podemos analisar a borda do clock ter uma subida, neste momento podemos verificar que o sistema consegue registrar em sua memória o dado que foi inserido na entrada de dados, assim que é inserido podemos ver que a saída de dados daquele endereço de memória contém uma informação diferente do que é visto antes da borda de subida do clock.

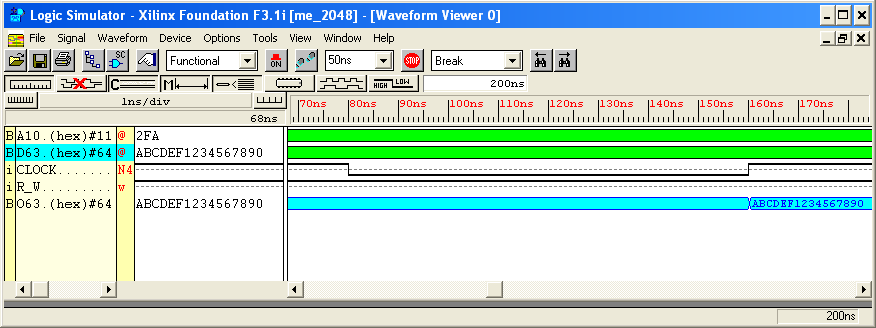


Figura 13 – Registrando os valores após 200ns.

Depois que foi registrado o conteúdo no endereço de memória “2FA” foi feita a alteração para o endereço de memória “5FA”, no próximo instante que é rodado o sistema podemos ver que a saída de dados apresenta o valor zerado, pois como estamos em um novo endereço de memória ele aparece zerado.

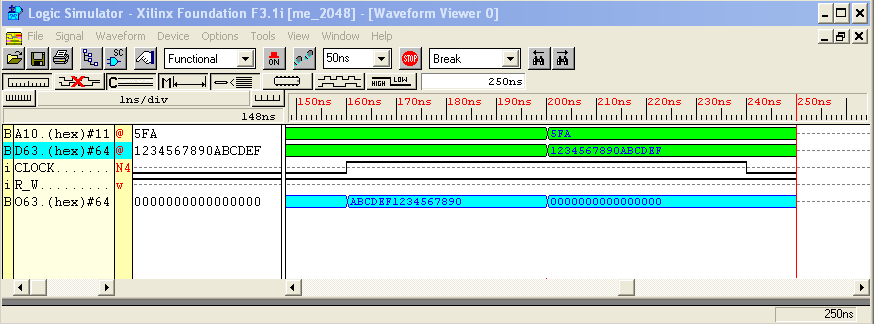


Figura 14 – Momento após a inserção de novos valores no endereço de memória e na entrada de dados.

Após a borda de subida do clock podemos reparar que no endereço “5AF”, ouve o registro do valor “1234567890ABCDEF”, pois como dito antes nesta borda de subida conseguimos gravar naquele endereço de memória este valor.

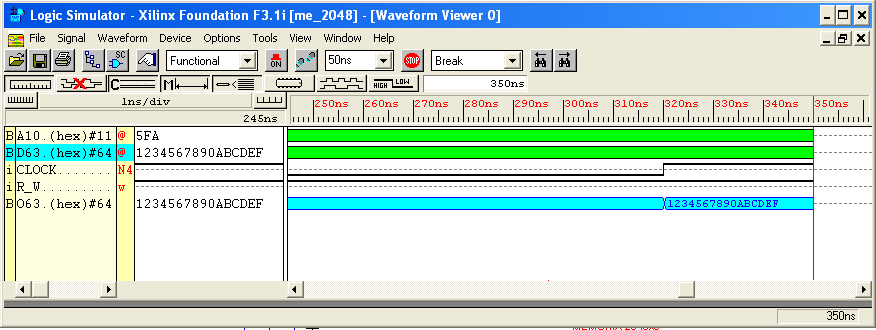


Figura 15 – Momento que o valor é inserido na memória.

Diante da segunda inserção o registrador de read/write foi selecionado para a função leitura, fazendo com que ele tivesse sua borda decaindo, após isso foi modificado o endereço de memória para “2FA”, para que pudesse ser extraído a informação do local da memória, este dado retornado seria “ABCDEF1234567890”.

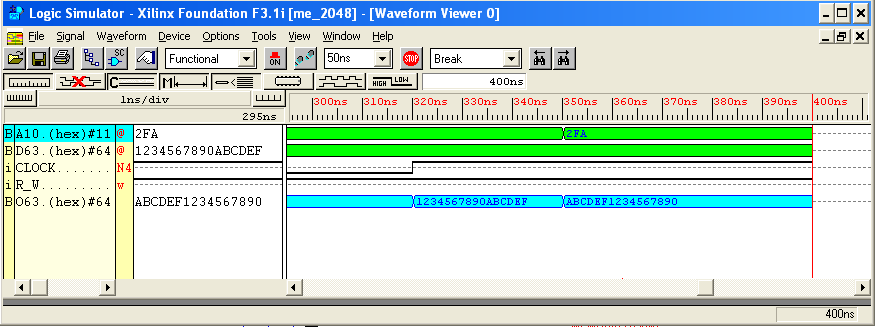


Figura 15 – Momento que o endereço de memória “2FA” é lido.

1. **Aplicações**

Hoje temos muitas aplicações para subsistemas de memória e estes sistemas são cada vez maiores em espaço de armazenamento e com um tamanho cada vez menor, porém tudo se inicia com pouco voltado a área de ti, o primeiro HD foi feito de 1955 a 1975 por Pablo Yuri, está memória foi considerado o primeiro HD da história, o mesmo havia capacidade de 5MB, a partir daí foram implementadas cada vez mais tecnologias nas memória fazendo com que ela tivesse mais armazenamento e menos espaço físico, atualmente implementamos sistemas de memória em computadores, celulares, Smart Watch, drones, Arduínos, etc. Tudo que necessita que algo seja salvo é utilizado um sistema de memória.

1. **Conclusões**

A partir deste relatório se conclui com seu objetivo alcançado, o mesmo seria obter uma memória RAM de 2Kx32, além de obter este objetivo conseguimos ir além chegando a uma memória RAM de 2Kx64, a partir do material de aula podemos avançar os modelo do subsistema de menor quantidade de armazenamento até o atual subsistema que conta com 2Kx8, após isso necessitamos entender o conceito de associação em paralelos voltado a componentes de memória, acabando nos confundindo na parte de relação com o endereçamento de memória pois demorou a entender que quanto mais se aumenta a quantidade de bits da memória maior será os espaço de dados que será adicionado a um banco de memória.

Este projeto contribuir para o entendimento base de uma máquina pois sem memória não haveríamos dados importantes salvos, dados estes que podem ser essenciais para a vida no planeta terra, este projeto instrui como podemos mexer com o nível mais baixo do dado, para que se possa entender como poderemos otimizar sistemas utilizando processos mais simples, porém mais no core do programa.

1. **Bibliografia utilizada e/ou consultada**

Xilinx et al. Xilinx 7 Series FPGA Libraries Guide for Schematic Designs. Xilinx Inc: San Jose, Califórnia, E.U.A. 2011. Disponível em:

https://www.xilinx.com/support/documentation/sw\_manuals/xilinx13\_2/7series\_scm.pdf Acesso em: 20 abr. 2021.

Daniel Saraiva, J.E. MEMÓRIA ARTIFICIAL Editora Biblioteca da Universidade de oxford, 2000.

Acesso em: 20 abr. 2021.