**UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA**

DEPARTAMENTO DE TECNOLOGIA

TEC498 PROJETO DE CIRCUITOS DIGITAIS

**PROBLEMA 03: “Memorizando!”**

Allen Hichard Marques dos Santos

**Tutor (a):** Márcia Prado

# 1. INTRODUÇÃO

Um dispositivo básico para armazenamento de uma informação binária (registrador) foi proposta pelo grupo Inova Digital Bahia S.A., para armazenar informações de formas contínuas, sem que haja uma perda de dados no sistema, para isso propôs uma memória, dispositivo capaz de armazenar temporariamente ou permanentemente alguns bits de maneira estruturada, habitualmente na forma de um arranjo bidimensional, onde as operações endereçadas de leitura e escrita são realizadas através de comandos e executados por um mecanismo de controle.

Com base no dispositivo já existente de Matriz de Leds, foi proposto aprimorar o protótipo para sinalizar até oito coordenadas informadas pelo usuário. Nesse sentido, a equipe técnica resolveu lançar um projeto de um circuito digital capaz de realizar comparações sucessivas a partir de valores binários armazenados em uma estrutura de memória e dados oriundos de um barramento em paralelo de oito bits (um byte). Contudo a finalidade é controlar a visualização da matriz que deve ser capaz de manter acesos todos os Leds cuja comparação foi dada como válida contendo um limite máximo de Leds acesas (8 Leds).

# 2. FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão expostos os conceitos dos dispositivos e periféricos utilizados na confecção do produto, assim como os aspectos teóricos dos mesmos à luz das informações já existentes.

## 2.1 Flip-Flop

Os flip-flops são dispositivos biestáveis síncronos, também conhecidos como multivibradores biestáveis. Nesse caso, o termo síncrono significa que a saída muda de estado apenas no momento especificado pela entrada de disparo denominado de clock (CLK), a qual é indicada como uma entrada de controle (C); ou seja, as mudanças na saída ocorrem em sincronismo com o clock (FLOYD, 2007).

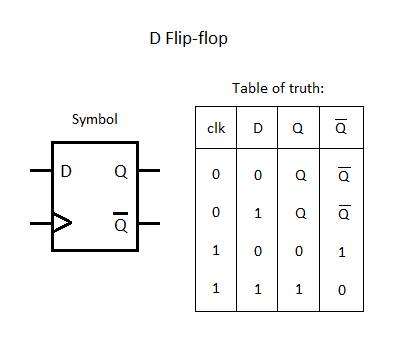
Um flip-flop disparado por borda muda de estado na borda positiva (borda de subida) ou na borda negativa (borda de descida) do pulso de clock e é sensível às entradas apenas nas transições do clock.

**2.1.1 Flip-Flop D Disparado por Borda**

O flip-flop D é usado quando um único bit de dado (0 ou 1) é para ser armazenado (FLOYD, 2007).. O bit armazenado só é liberado se tiver um pulso de clk (Clock), pois esse tipo de dispositivo biestável só é disparado por borda.

Observe que o flip-flop na Tabela 1 tem apenas uma entrada chamada D, se desconsiderarmos o detector de bordas. A entrada D, pode armazenar um bit de sinal que é liberado apenas pelo detector de bordas (Usamos o pulso de Clock como detector de borda), o detector de bordas pode ser ativado em nível lógico alto ou baixo a depender de sua aplicação.

Tabela 1: Imagem de um flip-flop D, e sua tabela da verdade.

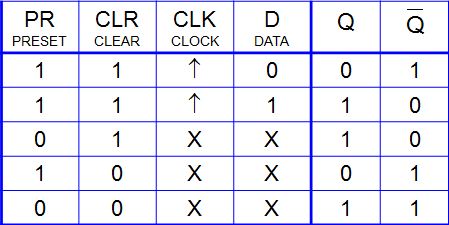


Fonte: http://imgarcade.com/1/d-flip-flop-circuit/.

A operação lógica do flip-flop D disparado por borda positiva é resumida na Tabela 2. A operação de um dispositivo disparado por borda negativa é evidentemente a mesma, exceto que o disparo ocorre na borda de descida do pulso de clock. Lembre-se, a saída Q segue a entrada D na borda ativa ou de disparo do clock.

O Clear e o Preset são usados para melhor controle do Flip-Flop, pois ele não leva em consideração a entrada D e detector de bordas, o papel principal e Setar ou Resetar diretamente a saída. O Clear reseta a saída e põe diretamente em nível lógico baixo, já o Preset seta a saída e coloca em nível lógico alto (TOCCI, 2000).

Tabela 2: Tabela da Verdade com o Clear e Preset do Flip-Flop D.



Fonte: http://andrescorrea.weebly.com/introduction-to-flip-flops.html

## 2.2. Contador

Os flip-flops podem ser conectados juntos para realizar operações de contagem. Tal grupo de flip-flops é um contador. O número de flip-flops usados e a forma na qual eles são conectados determinam o número de estados (denominado módulo) e também a sequência específica de estados que o contador percorre durante cada ciclo completo (FLOYD, 2007).

Os contadores são classificados em duas grandes categorias de acordo com a forma que eles recebem os pulsos do clock: assíncronos e síncronos (FLOYD, 2007). Nos contadores assíncronos, normalmente chamados de contadores ondulantes (ripple counters), o primeiro flip-flop recebe o clock por meio de um pulso de clock externo e todos os flip-flops sucessivos recebe o clock através da saída do flip-flop anterior. Em contadores síncronos, a entrada de clock é conectada a todos os flip-flops de forma que eles recebem o clock simultaneamente.

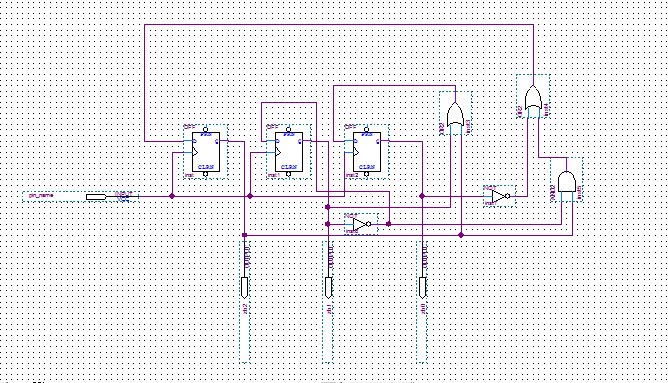
Dentro de cada uma dessas categorias, os contadores são classificados principalmente pelo tipo de sequência, o número de estados, ou o número de flip-flops no contador.

**2.2.1 Contador Síncrono**

O termo síncrono se refere aos eventos que têm uma relação de tempo fixa de um em relação ao outro. Um contador síncrono é aquele no qual todos os flip-flops recebem pulsos de clock ao mesmo tempo por meio de uma linha comum (TOCCI, 2000).

A Figura 1 é o contador desenvolvido pela equipe, ele conta 3 bits de memória que vai dos bits 000(0) até o 111(7), tendo um papel de selecionar a memória atual e verificar se os bits estão sendo registrado no byte correto, além de sinalizar em qual memória o sistema está atualmente, podendo ser visualizada no display.

Figura 1: Contador de 3 bits tendo estados de 000 à 111 em Binário.



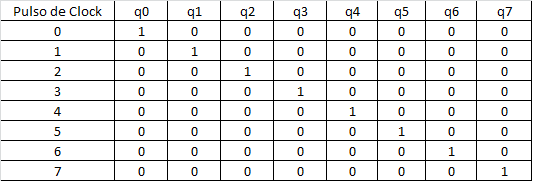
Fonte: O próprio autor.

**2.2.1 Teoria e aplicação da ideia de um Contador Anel**

O contador em anel utiliza um flip-flop para cada estado em sua sequência. Ele tem a vantagem de não necessitar de portas de decodificação. No caso de um contador em anel de 8 bits, existe uma única saída para cada dígito decimal (FLOYD, 2007)..

Como visualizado na Tabela 3, se correlacionando com o problema, daria como um seletor de memória atual, mas não foi usado um Contador em Anel, foi usado sua tabela como auxiliadora do contador de 3 bits, ou seja, não foi preciso usar flip-flops para ter a mesma tabela, foi usado apenas um simples codificador demux que faria o papel de um codificador anel.

Tabela 3: Contador Anel de Três Bits baseado no pulso do Clock



Fonte: Floyd, Thomas L. Sistemas digitais [recurso eletrônico]: fundamentos e aplicações / Thomas L. Floyd; tradução José Lucimar do Nascimento. – Dados eletrônicos. – 9. ed. – Porto Alegre : Bookman, 2007.

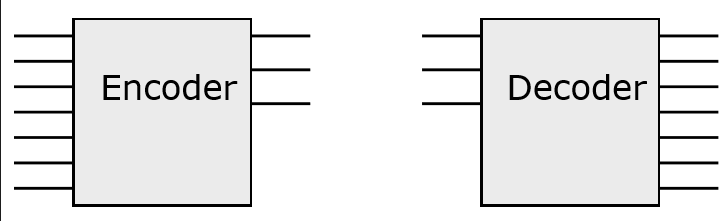
## 2.3. Decodificador e Codificador

Um decodificador é um circuito digital que detecta a presença de uma combinação específica de bits (código) em suas entradas indicando a presença desse código através de um nível de saída especificado (FLOYD, 2007)..

Um codificador é um circuito lógico que realiza essencialmente a função “inversa” do decodificador. Um codificador aceita um nível ativo em uma de suas entradas representando um dígito, tal como um dígito decimal ou octal, e o converte em uma saída codificada, tal como binário ou BCD. Codificadores também podem ser implementados para codificar vários símbolos e caracteres alfabéticos. O processo de conversão de símbolos familiares ou números para um formato codificado é denominado de codificação (FLOYD, 2007).

Conforme a Figura 2, Encoder (Codificador) recebe 7 entradas e converte para 3 bits, esses 3 bits é a representação daquela entrada, com 3 bits de saídas eu posso codificar até 8 possíveis entradas. O Decoder (Decodificador) é uma função inversa ao codificador, ele recebe 3 bits e libera o valor correspondente a sua entrada.

Figura 2: Imagem de como seria a visualização de um codificador e decodificador.



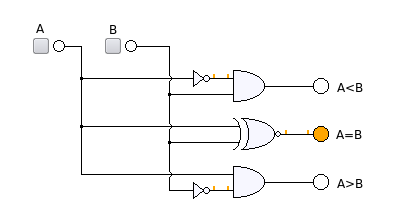
Fonte: Armando Sanca Sanca, Professor de Circuitos Digitais - Uefs.

## 2.4. Comparadores

A função básica de um comparador é comparar as magnitudes de dois números binários para determinar a relação comparativa entre eles. Em sua forma mais simples, um circuito comparador determina se dois números são iguais (FLOYD, 2007).

O comparador tem outras finalidades menos usadas, mas é melhor deixar claro, como mostrado na figura 3, é possível fazer três comparações de dados, maior, menor e a mais usada o igual, uma combinação com portas lógicas pode ser utilizada para mostrar o resultado obtido.

Figura 3: Comparador de 2 bits.



Fonte: http://es.wikipedia.org/wiki/Circuito\_comparador.

De acordo com a Tabela 4, pode se notar que foram usadas portas AND com entradas negadas em lugares diferentes para indicar se os bits são maiores ou menores podendo ser visualizada na figura 06, mas que principalmente a parte da igualdade é uma simples XNOR, pois com valores iguais a saída respectiva é 1 que representaria a igualdade.

Tabela 4: Tabela da Verdade de um comparador de 2 bits.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A<B | A=B | A>B |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |

Fonte: http://es.wikipedia.org/wiki/Circuito\_comparador.

## 2.5. Registrador

Os registradores de deslocamento são um tipo de circuito lógico muito parecido com os contadores digitais. Os registradores são usados principalmente no armazenamento de dados digitais e não possuem uma característica interna de sequência de estado como os contadores (FLOYD, 2007).

## 2.5.1. Armazenamento de Dados em Paralelo

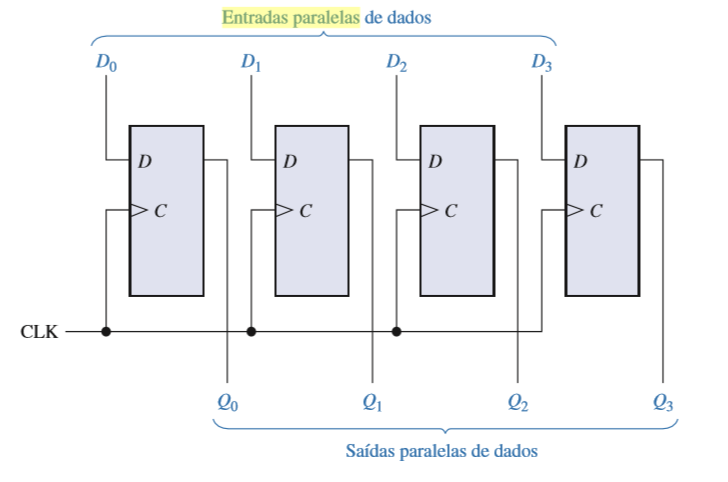
Uma necessidade comum em sistemas digitais é armazenar diversos bits de dados em linhas em paralelo simultaneamente num grupo de flip-flops. Conforme a tabela 8: Cada uma das quatro linhas paralelas de dados é conectada na entrada D de um flip-flop. As entradas de clock dos flip-flops são conectadas juntas, de forma que todos os flip-flops são disparados pelo mesmo pulso de clock. Nesse exemplo são usados flip-flops disparados por borda positiva, assim os dados nas entradas D são armazenados simultaneamente pelos flip-flops na borda positiva do clock.

Esse grupo de quatro flip-flops é um exemplo de uso de um registrador básico para armazenamento de dados. Em sistemas digitais, os dados são normalmente armazenados em grupos de bits (geralmente oito ou múltiplos dele) que representam números, códigos ou outras informações (TOCCI, 2000).

Como informado na figura 4 o que caracteriza se um registrador é totalmente paralelo é se as entradas D, são entradas individuais e o Clock de ativação é o mesmo para todos os flip-flops, ou seja, quando isso acontece e o flip-flop é disparado todos os bits armazenados tem o mesmo tempo de saída.

Observação: Não importa o meio de organização dos flip-flops, eles podem ser organizados de forma serial “Como visualizado na figura 4”, ou pode ser organizado de forma paralela, “Um abaixo do outro”, isso não interfere o conceito real de entradas e saídas, tendo em vista que muitas pessoas confundem e acham que é serial só por causa da organização dos flip-flops.

Figura 4: Registradores de deslocamento com entradas e saídas paralelas



Fonte: Floyd, Thomas L. Sistemas digitais [recurso eletrônico]: fundamentos e aplicações / Thomas L. Floyd; tradução José Lucimar do Nascimento. – Dados eletrônicos. – 9. ed. – Porto Alegre : Bookman, 2007.

## 2.6. Botão (Pushbutton)

## O botão é um mecanismo de controle que envia um bit de sinal para alguma ação tanto no físico tanto em lógica programação, normalmente utilizado para gerar uma onda de clock manual em um dispositivo de memória biestável.

## Normalmente o botão vem com o problema de bounce, que seria a trepidação que ocorre no momento que você clica no botão e solta, onde gera várias ondas irregulares, tendo que haver um tratamento adequado para anular ou minimizar a trepidação, normalmente chamado de Debounce que é apenas um dentre outras formas de solucionar tal problema.

## 2.7. Memória e Armazenamento

As RAMs são memórias nas quais os dados podem ser escritos ou lidos a partir de qualquer endereço selecionado em qualquer seqüência. Quando uma unidade de dados é escrita num determinado endereço numa RAM, a unidade de dado armazenada anteriormente nesse endereço é substituída pela nova unidade de dado. Quando uma unidade de dado é lida a partir de um determinado endereço na RAM, a unidade de dado permanece armazenada e não é apagada pela operação de leitura. Essa operação de leitura não destrutiva pode ser vista como uma cópia do conteúdo de um endereço enquanto deixa o conteúdo intacto. Uma RAM é usada tipicamente para armazenamento de dados de curta duração porque ela não pode manter os dados quando a alimentação é desligada (FLOYD, 2007).

Uma ROM contém permanentemente ou semi-permanentemente dados armazenados, que podem ser lidos da memória, mas não podem ser alterados ou não podem ser alterados sem um equipamento especializado. Uma ROM armazena dados que são usados repetidamente em aplicações tais como tabelas, conversões ou instruções programadas para a inicialização e operação de um sistema. As ROMs retêm os dados armazenados quando a alimentação é desligada, sendo, portanto, memórias não voláteis (FLOYD, 2007).

# 3. METODOLOGIA

Nesta seção serão discutidas as ideias, decisões e ferramentas utilizadas durante as seções tutoriais e em laboratório, que foram uteis para que o produto fosse concluído.

## 3.1 Circuitos Lógicos no Quartus

Após analisar o problema e ter conhecimento dos componentes necessários para a sua resolução, foi desenvolvido um circuito lógico sequencial que se adaptaria ao projeto de matriz e dos displays já existentes, só que teria novas características, dentre elas armazenarem oito bytes na memória e fazer comparações para validações de cada entrada.

**3.1.1. Métodos para Iniciar o Circuito Lógico:**

O problema implica que seja acrescentado mais um display de sete segmentos para a representação da posição da memória e também inserir um PushButton para alternar entre as posições e inserção de dados no sistema. O novo Display ficará responsável apenas por mostrar os números de 0 a 7.

O protótipo mantém as mesmas entradas e saídas dos protótipos anteriores só com o acréscimo da entrada do Pushbutton e saída para mais um display ser selecionado, com base nessas informações foram iniciados o circuito sequencial.

**3.2 Quartus**

O circuito lógico programação foi implementado através da ferramenta da altera Quartus 9.0, responsável por ter a finalidade de executar as funções propostas pelo problema e de garantir que toda a funcionalidade seja atendida.

**3.2.1. Processo de Implementação Geral**

Baseado no protótipo da matriz de Leds foi analisado todas as alterações que seriam feitas para que o problema fosse atendido, no entanto a matriz ainda não é capaz de acender mais de um ponto sem que a nova coordenada apague a anterior, tudo que se tinha em mãos era o circuito físico para poder se basear e estruturar a nova solução.

Com base no anterior, temos 8 entradas para o sistema, onde são divididas entre linhas (0-6 em binário) e colunas (A-E em binário), onde são codificadas e representadas para o usuário, para isso requerendo 12 saídas como informado entre 0-6 e A-E.

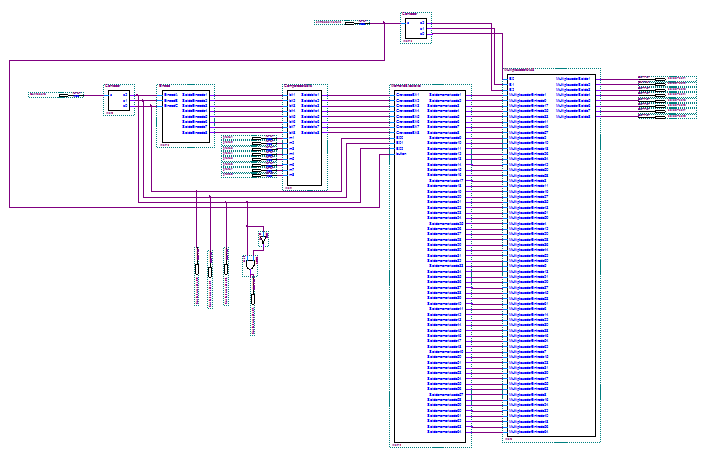
O sistema possui dados de entradas e saídas. As entradas eram informadas no circuito físico e sua representação era mostrada ao usuário, sempre que era informada uma coordenada à saída fazia sua representação e aquela coordenada anterior simplesmente não existiria mais, então foi pensado em formas de armazenar as coordenadas anteriores mantendo elas acesas e assim o usuário poder informar uma nova coordenada sem que haja perda de dados, no caso não apagando as Leds anteriores.

Através de toda a fundamentação teórica (mostrada anterior) desse problema, foi chegada a conclusão de que para fazer o circuito ter toda a funcionalidade proposta, seria necessário ter um contador para especificar cada posição de memória, um comparador para verificar se a coordenada é válida, um registrador para mostrar ao usuário todas as informações geradas para o sistema, além disso um multiplexador constante que iria mostrar na matriz em tempo real todas as coordenadas tanto a atual como as anteriores, enviando assim dados constantes para os circuitos decodificadores da matriz de Leds e displays.

**3.2.2. Análise para o desenvolvimento**

Como mostrado na figura 5, Circuito lógico tem uma memória de 8 Bytes(64bits), possuindo capacidade de armazenar 8 coordenadas tendo cada coordenada 8 bits que representa a linha e a coluna a ser mostrada na matriz. Para que se fosse Possível chegar a esse armazenamento o circuito passa por uma série de análises que serão explicadas abaixo.

Figura 5: Apenas o Problema 3: Capaz de armazenar 8 bytes na memória.



Fonte: O próprio autor.

**3.2.2.1. Protótipo dos Problemas Anteriores**

O circuito até então que foi constituído pela equipe tinha duas funcionalidades:

* A primeira era mostrar apenas um único ponto na Mariz de Leds respectivas a coordenadas de entrada.
* O segundo era mostrar ao usuário em dois displays quais foram às coordenadas indicadas para acender a matriz.

O aprimoramento foi encima das duas funcionalidades anteriores, reaproveitando partes da primeira e da segunda para atender os requisitos Atuais. No primeiro problema não houve nenhum alteração na decodificação, já no segundo sofreu várias alterações.

Alterações feitas nas decodificações dos displays:

* O decodificador de Linhas e Colunas, não decodificava o número 7, sendo assim tivemos que refazer algumas letras do display para poder visualizar o novo número, pois, era necessário porque a memória será mostrada em mais um display que vai de 0 até 7.
* O multiplexador ganhou novas características, houve um aumento de seleção de 1 bit para 2, sendo necessário para mandar 12 bits por apenas 4 saídas e isso não é possível tendo apenas 1 seletor. Os 12 bits são as coordenadas da linha, coluna e da posição de memória que vão ser codificadas e representadas em três displays.
* O seletor que ao invés de alternar apenas entre dois display irá se alternar para 3, para isso foi necessário fazer um contador de 2 bits que tinha como finalidade fazer essas alterações constantes com a ajuda de um clock. Esse contador libera bits como 00, 01, 11 onde é codificado e selecionando seu display representante, cada display tem característica baseada em cada saída do contador.

Pronto, chegamos ai a aprimorar o segundo circuito para receber informações do terceiro problema, como os anteriores problemas não tinha capacidade de memória, eles se resumem apenas a decodificar bits e mostrar no circuito físico o resultado.

**3.2.3. Fase final do produto em relação à construção lógica do Quartus (Processo de Construção do Problema 3)**

Foi pensado em fazer o projeto que se adaptasse as alterações e aperfeiçoamento feitos. Cada posição do contador tem um número binário que representa linha e coluna, logo o valor atual que é liberado pelo codificador é comparado com os valores de entradas, se os valores forem iguais às entradas são liberadas para memória e se os valores forem diferentes os que entram na memória é uma entrada barrada tipo 0000/0000.

Cada memória tem uma característica de seleção, por exemplo, chamado de seleção em anel ou contador anel, que irá verificar a posição atual da memória que está se comparando e autorizando que ocorram mudanças de acordo com as entradas, se o contador estiver na posição 100, os bits comparados naquele instante pelo circuito entrará apenas na memória 100, a característica é semelhante a um Demux.

Independente de qual posição esteja a memória, o multiplexador irá enviar as informações dos 8 bytes que estão na memória para serem codificadores, ou seja, ele garante que desde a primeira coordenada até a última coordenada, nenhuma delas irão se apagar da matriz de Leds. O que acontece com as memórias que ainda não foram comparadas? Se estivermos na comparação da memória 101(5), o restante terá ainda saídas zeradas o que não acontece nada no sistema e até a cinco será multiplexada.

Com a memória cheia, serão visualizadas todas as informações na matriz de Leds independentes se todas as comparações foram dadas como válidas ou não, o que irá mostrar se a memória chegou ao fim é apenas um display que mostrará a posição atual na memória para o usuário possa saber qual comparação a ser feita desde a primeira até a última combinação.

1

## 4.0. RESULTADOS E DISCUSSÕES

Nesta subseção serão explicadas as decisões finais para que fosse resolvido o problema e montagem do circuito físico através da descrição dos passos utilizados para as principais partes do circuito.

### 4.1. Montagem do circuito físico

A montagem do novo protótipo não sofre muitas alterações, apenas o acréscimo de um PushButton e mais um display de sete seguimentos para visualizar a posição da memória atual no sistema.

**4.1.2 Equações simplificadas dos segmentos do display feitas a partir do Mapa de Karnaugh.**

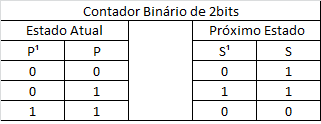
Começando pelo codificador dos displays, ele sofreu alterações referente ao problema 2, como existe uma posição de memória 111(7), houve uma melhoria na codificação para poder receber o novo dado sem a necessidade de um novo codificador: as novas equações podem ser visualizadas abaixo:

1. **= a’c + cd’ + a’b’d’ + a’bd + abd’**
2. **= a’b’ + a’c’d’ + b’cd’ + a’cd + abc’d**
3. **= a’c’ + a’d + a’b + bc’d + ab’c**

Além das novas expressões para codificar o número 7, Karnaugh foi utilizado para decodificar cada saída do contador, pois cada saída do contador de 3 bits tem um representante de 8 bits que representa linha e coluna, e também utilizado para simplificar as equações do contador de 2 e 3 bits.

Abaixo estão as tabelas onde foi usado Karnaugh e suas expressões simplificadas.

Tabela 5: Contador de 2 bits contendo 3 estados, utilizado para alternar entre os 3 displays.



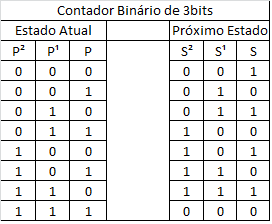
Fonte: O próprio autor.

Expressões da Tabela 5:

P0 = a’

P1 = a’b

Tabela 6: Contador de 3 bits contendo 8 estados, utilizado para mostrar a memória atual no display e por enviar bits para decodificação de cada posição de memória.



Fonte: O próprio autor.

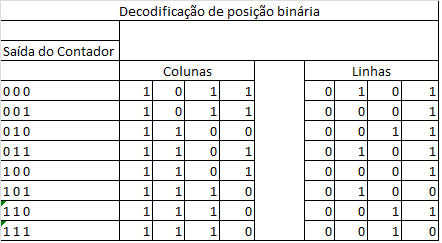
Expressões da Tabela 6:

P0 = c’

P1 = b’c + bc’

P2 = ab’ + ac’ + a’bc

Tabela 7: decodificador de cada posição do contador para conseguir seu representante



Fonte: O próprio autor.

Expressões da Tabela 7:

P0 = 1

P1 = b + a

P2 = a’b’ + b’c + ab

P3 = b’c’ + a’c

P4 = 0

P5 = a’b’c’ + a’bc + a’b’c

P6 = bc’ + ab

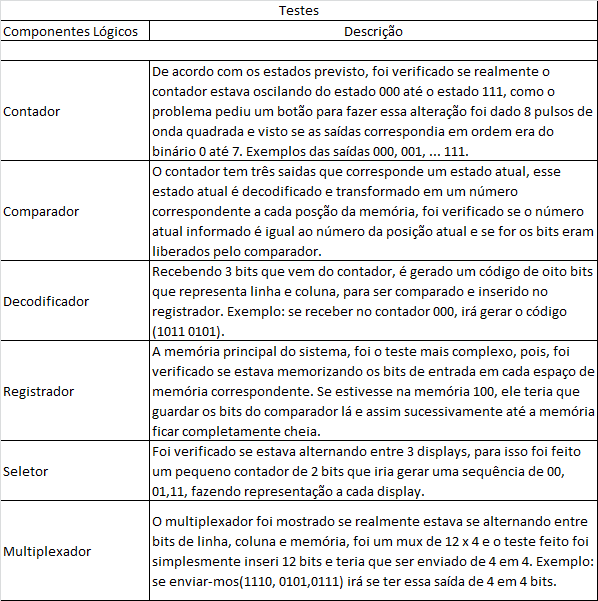
P7 = a’ + c’

### 4.2 Testes no Quartus II

O teste do Quartus foi realizado de acordo com suas entradas e suas respectivas saídas. No processo de construção foram feitas vários macros, macros esses que tem funções independentes no circuito, como é o caso do decodificador e do multiplexador.

Antes de fazer o teste geral do circuito, foi pensando em fazer os testes por partes de cada modulo (Macro). Os testes são mostrados na Figura 6 a seguir:

Figura 6: Descrição dos testes realizados no Quartus.



Fonte: O próprio autor.

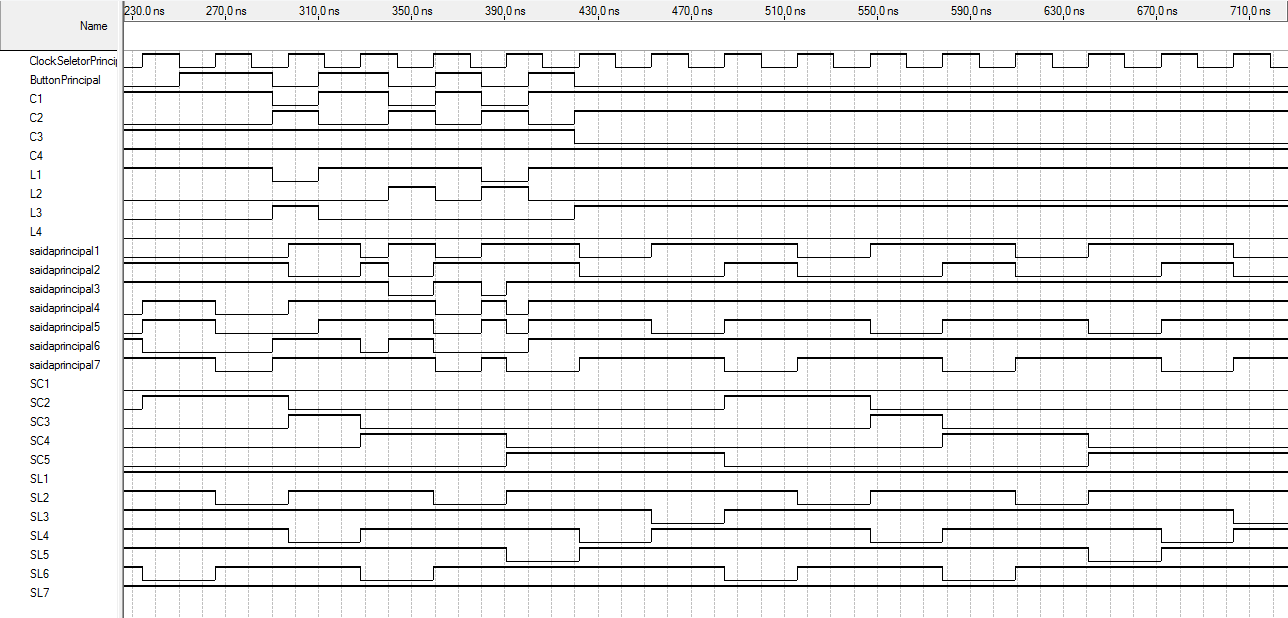
Após todos os testes dos macros serem feitas foi montado o projeto final para testar sua funcionalidade por meio de dois tipos de teste (Funcional e Temporal).

#### 4.2.1. Functional (Funcional)

No teste funcional foi verificado se os valores estavam armazenados na memória sem que houvesse a perda de dados do sistema.

A figura 7 foi inserida todas as coordenadas válidas em seus respectivos espaços de memória, e analisado as saídas com a memória cheia. Saídas na figura foram denominadas SC1 a SC5 (Coluna) e SL1 a SL7 (Linha), de acordo com o tempo é possível verificar o nível lógico se variando o que significa que os valores estão armazenados na memória corretamente.

Figura7: Entradas e saídas do teste funcional no Quartus.

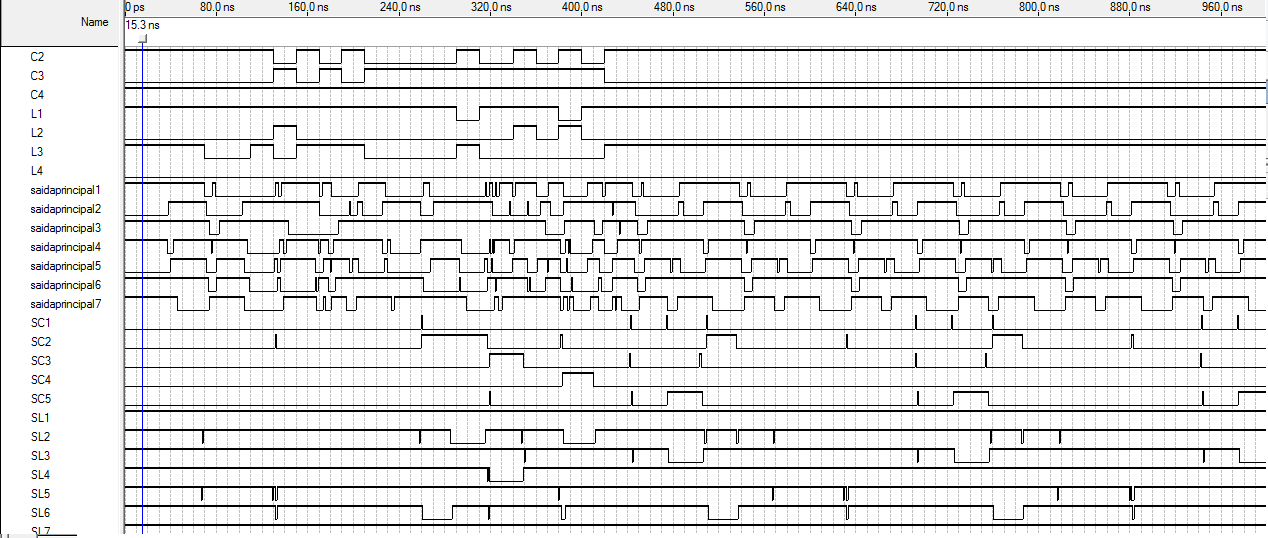


Fonte: Próprio Autor

#### 4.2.2. Timing (Temporal)

Com o teste realizado na Figura 8 pelo Timing conseguimos verificar de acordo com o tempo o que acontece detalhadamente com o circuito, podendo verificar nos mínimos detalhes os baixos picos de alternância entre os ciclos de trabalho do circuito.

Figura 8: Entradas e saídas do teste temporal no Quartus.



Fonte: Próprio Autor

Tanto os testes funcionais como o teste do tempo foram testados do circuito completo como um todo alcançando 100% dos requisitos, pela análise dos testes pode notar que existe 8 bytes armazenados, que foram validados a entradas onde a comparações aprovou a inserção na memória, percebesse que o contador ondula de 0 até 7 e os 3 displays ficam alternando entre linha/coluna/memória.

#### 4.2.3. Teste de Erros

* Novamente foi verificado se estava sendo inserida linha e coluna, caso o contrário o display não mostraria nenhum valor no display;
* Pressionar o Pushbutton duas vezes seguidas;
* A memória chegar ao limite e for informada uma nova coordenada;
* Selecionar a memória errada;
* Não dar um clock continua para melhor multiplexar os dados das memórias;

Uma informação importante: O Circuito Físico já vai iniciar da posição 000, aonde informada uma coordenada será analisado pelo comparador se é válida ou não, dai pressionado o botão para ir a memória 001.

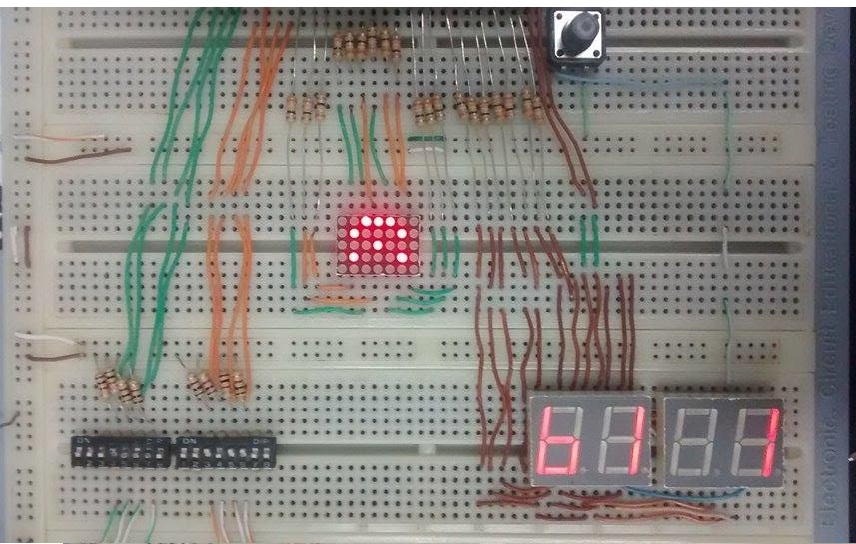
### 4.3. Circuito Final

As únicas mudança no protótipo foi apenas acrescentar um novo display de 7 segmentos para representar a memória do sistema e um PushButton que fará o contador ser ativado sempre que pressionado, além de selecionar a memória atual.

Sendo assim o circuito só precisa de apenas 1 entrada e 1 saída nova para seu completo funcionamento, não havendo mais nenhuma mudança para atender o propósito do problema.

A Figura 9 mostra como ficou o produto final da montagem com exemplo da memória completamente cheia, mostrando todas as posições consideradas validas na Matriz de Led, sendo possível verificar a memória atual, com a respectiva linha e coluna inserida.

Figura9: Circuito físico completo.

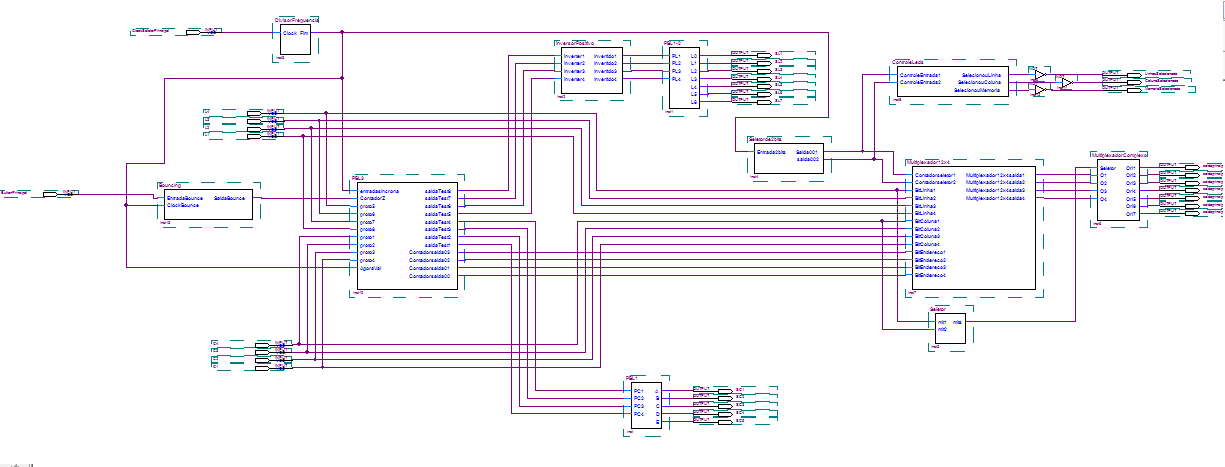


Fonte: Próprio Autor

O botão no circuito é a forma de endereçar na memória correta sua respectiva coordenada, além de passar para o próximo estado da memória, sendo responsável por enviar bits a serem decodificados para mostrar nos display qual a posição atual.

De acordo com a Figura 10, do projeto final do Quartus, foi integrado aos dois protótipos anteriores o que garante que após a memorização das coordenadas na memória, todos os dados serão decodificados e mostrado em tempo real em uma matriz de Leds.

Figura 10: Projeto final do Quartus

 Fonte: Próprio Autor

# 5. CONCLUSÃO

Analisando os requisitos solicitados, ficou claro como seria o processo de construção de todas as funcionalidades do circuito digital. O protótipo novo veio a agregar os dois anteriores que só mostrava uma coordenada na Matriz de Led podendo visualizar a mesma no Display. A grande motivação da equipe era uma utilização de memória para armazenar dados externos e os tornarem permanentes enquanto execução do programa.

O novo sistema foi construído com o acréscimo de um botão que sinalizaria a memória do sistema em um novo display, sendo assim o que faltava era a aplicação lógica. A aplicação lógica conseguiu atender os principais critérios solicitados, tornando possível fazer comparações de entrada com valorem armazenados na memória, e armazenar até 64 bits na mesma.

O sistema alcançou o sucesso devido à sincronização entre o circuito lógico e físico. O circuito lógico foi testado para garantir que todo o projeto seria atendido sem restrições, chegando a atingir 100% das funcionalidades. Quando foi ligado o circuito físico com a funcionalidade obtida no lógico, foi possível verificar a memória armazenando os bits para ser mostrados do display. Cada ponto no display era a coordenada válida para aquela posição, ao completar a memória foi possível notar todos os 8 pontos acesos, o que caracterizou o protótipo de memória finalizado.

Todo o projeto foi implementado de forma ergonômica, usando o mínimo de portas e armazenamentos de memória possíveis (o que não alterou os requisitos), além de aperfeiçoar o uso do Botão, que foi agregado à função de passar a memória e deixar a posição anterior gravada. Sendo assim todo objetivo solicitado para a nova versão protótipo foi atendida com sucesso.

# 6. REFERÊNCIAS

TOCCI, RONALD J; WIDMER, NEAL S. Sistemas Digitais: Princípios e Aplicações 7ª edição.

FLOYD, THOMAS L. Sistemas digitais [recurso eletrônico]: fundamentos e aplicações / Thomas L. Floyd ; tradução José Lucimar do Nascimento. – Dados eletrônicos. – 9. ed. – Porto Alegre : Bookman, 2007.