Universidade Federal do Rio Grande do Norte

Centro de Tecnologia - CT

Departamento de Engenharia Elétrica

Relatório Técnico

ELE1717 - Grupo 01 - Problema 01 - Projeto

Albertho Siziney

Allysson Andrade

Anny Pinheiro

Isaac de Lyra

Natal, 14 de Junho de 2021

Resumo

No presente trabalho é descrito de forma sequencial o processo de criação de um circuito integrado para o sistema digital de uma secretária eletrônica, apresentada como o primeiro problema da matéria de Sistemas Digitais, que possui capacidade de armazenamento de até 25 mensagens. Para o desenvolvimento deste processo foi aplicado o método de projeto RTL, o qual utiliza máquinas de estados de alto nível, máquinas de estados finitos, blocos de controle e operacional em sua implementação. O trabalho é finalizado com toda a teoria e desenvolvimento do projeto realizado, permitindo que a implementação deste possa ser aplicada posteriormente.

Palavras-chaves: Sistemas digitais; Circuitos integrados; Máquinas de estado de alto nível; Projeto RTL.

Sumário

1 Introdução

O foco do presente trabalho é a descrição e desenvolvimento em projeto do funcionamento de uma secretária eletrônica digital capaz de armazenar até 25 mensagens para possível implementação do circuito integrado deste sistema digital. A secretária possui diversos comandos e funcionalidades a serem contemplados além do armazenamento de mensagens - uma mensagem contendo o mínimo de 10 amostras digitais cada com 8 bits - tais como gravar e reproduzir mensagens, resetar o sistema apagando todo o conteúdo gravado e também um estado de repouso no qual aguarda-se algum comando externo qualquer em relação às suas funcionalidades.

Para o desenvolvimento do circuito integrado deste sistema foi proposto o método de projeto em nível de transferência entre registradores, também conhecido como projeto RTL. Este permite a descrição das possíveis operações a serem realizadas com os dados de entrada, de saída e dos registradores, além de definir o controle destas operações (VAHID, 2008). O intuito deste método é capturar o comportamento do sistema e permitir que este seja convertido em um circuito composto por bloco de controle e bloco operacional.

2 Desenvolvimento

O método - segundo Vahid (2008) - de forma geral é constituído de uma sequência de passos de tal forma que seja possível captar o comportamento desejado e converte este em um circuito integrado. O projeto RTL é definido em quatro passos: a obtenção de uma máquina de estados de alto nível, a criação de um bloco operacional, a implementação da conexão do bloco de controle com o operacional e por fim a obtenção da máquina finita de estados do bloco de controle e a implementação deste.

2.1 Máquina de Estados de Alto Nível (MDE)

Sendo a criação da MDE o primeiro passo do projeto RTL, o desenvolvimento desta é feito com base em estados e transições que descrevem o comportamento desejado. Estes são variados de acordo com condições definidas para além de operações booleanas com os bits de entrada e saída.

Segundo Vahid (2008), uma MDE é um modelo de computação similar a uma máquina de estados finitos (FSM), mas com características adicionais que permitem a descrição de cômputos envolvendo mais do que simplesmente dados booleanos. Ou seja, cada funcionalidade pode ser definida como um estado e suas respectivas transições indicando os sinais e os dados de entrada e saída compondo todo o sistema. Com isso, a MDE é uma extensão da FSM em que as entradas e saídas podem envolver outros tipos de dados além de apenas booleanos, registradores locais podem ser incluídos e as condições podem envolver expressões aritméticas.

2.2 Bloco Operacional (*Datapath*)

Com base na MDE e as funcionalidades descritas do sistema, executa-se a criação de um bloco operacional (*datapath*). Utilizando de componentes como comparadores, multiplexadores e registradores, este bloco deve ser capaz de realizar as operações que envolvem dados do sistema, ou seja, a manipulação, armazenamento e transferência de dados não booleanos deve ser implementada nesta etapa. Este bloco permitirá a substituição da MDE por uma FSM que controla o bloco operacional (VAHID, 2008).

Esta etapa pode ser dividida em alguns passos para melhor entendimento e construção do datapath. Os quais são:

I. Todas entradas/saídas de *dados* são entradas/saídas do *datapath*;
II. Implementar o armazenamento de dados acrescentando registradores;
III. Examinar cada estado e transição adicionando e conectando novos componentes operacionais para que as computações com dados sejam implementadas.

Ao fim destes passos a construção de um bloco operacional está completa, definindo todas as entradas/saídas necessárias para a conexão com o bloco de controle que irá ser definido posteriormente para a finalização do circuito integrado do sistema digital proposto.

2.3 Conexões do Bloco de Controle

Neste passo do projeto RTL, simplesmente realizamos a conexão do bloco de controle com o bloco operacional através da inserção das entradas e saídas booleanas do sistema digital. O bloco de controle, ainda não desenvolvido, é apenas visualizado como uma caixa/sistema que irá realizar determinadas funções de acordo com as entradas do sistema e retorna uma gama de saídas. Ou seja, a partir das entradas/saídas definidas no *datapath* irá ser definido quais entram e saem do bloco de controle permitindo a interação dos blocos, criando um circuito integrado.

2.4 Máquina Finita de Estados (FSM) e Implementação do Bloco de Controle

Em conhecimento das variáveis de entradas/saídas corretas para pleno funcionamento da comunicação do *datapath* com o bloco de controle, a obtenção da FSM é imediata. Esta terá os mesmos estados e transições que a MDE, com a diferença de que as entradas/saídas agora são booleanas e as operações, que determinavam as ações/condições presentes na MDE, passam a ser representadas por valores apropriados nos sinais de controle do *datapath* na FSM (VAHID, 2008). Assim, é descrito o comportamento do bloco de controle ao se definir a FSM.

Finalizada a etapa de construção da FSM, é implementada a construção do projeto do bloco de controle o qual é responsável por realizar a implementação da lógica combinacional que irá atuar nas entradas recebidas e fornecer as entradas de controle do *datapath*.

O projeto do bloco de controle é composto por cincos passos, sendo estes:

I. Criação da FSM que descreva o comportamento desejado (já realizada);
II. Criação da arquitetura padrão do bloco utilizando de um registrador de estados e uma lógica combinacional. Onde o registrador de estado irá controlar os bits que definem o estado (atual e futuro) da máquina e a lógica combinacional relaciona o estado e a entrada da FSM para determinar a saída do bloco de controle;
III. Codificação dos estados, onde são atribuídos números binários para representar cada estado existente na máquina. Cada codificação deverá ser única;
IV. Criação da tabela de estados para relacionar os estados com as entradas e saídas. Esta deverá ser uma tabela verdade que define para cada estado os valores de cada entrada e saída desejada/possível, e assim possa ser, posteriormente, gerada a lógica combinacional necessária para o bloco de controle;
V. Implementação da lógica combinacional. Com base no passo anterior, são retiradas equações lógicas da tabela-verdade desenvolvida e estas implementam a lógica combinacional do bloco de controle.

Finalizado todos os passos desta etapa, a construção do bloco de controle foi realizada e também foi implementado todo o projeto RTL para o sistema digital desejado.

3 Projeto

3.1 Criação da MDE

Para desenvolvermos a máquina de alto nível, apresentada a seguir, utilizamos como base as exigências presentes no atividade Problema 1, assim como funções comumente utilizadas em projetos do tipo RTL.

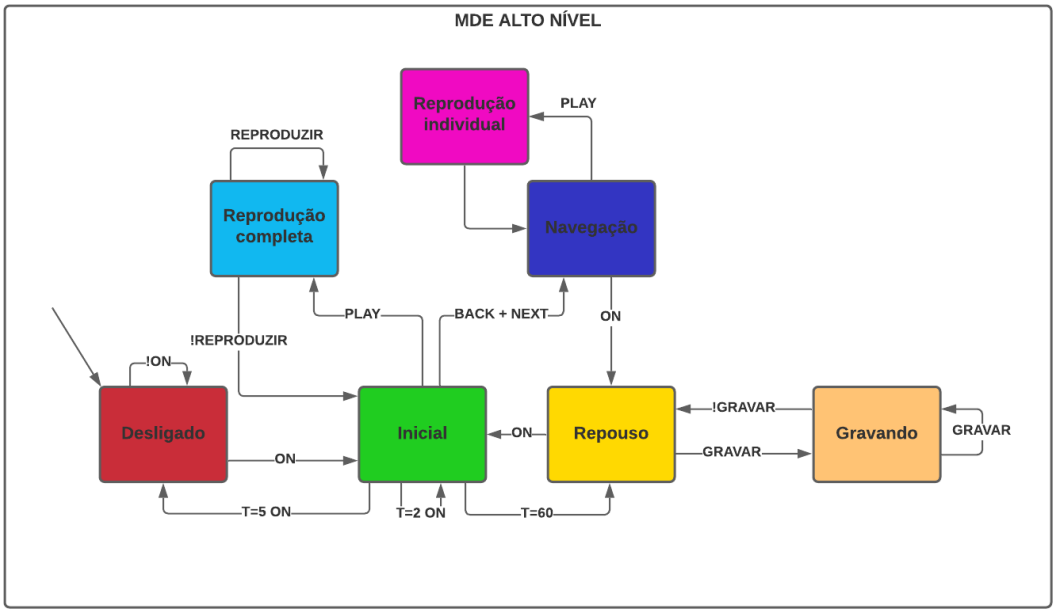


Figura 1: Máquina de estados de alto nível

A máquina se inicia no estado “Desligado”, o qual tem como função o desligamento da secretária eletrônica, ao entrar nele todas as entradas referentes aos botões e saídas nos leds e no display são dados como não acionados, saímos desse estado com o acionamento do botão ON, o qual nos direciona para o próximo estado.

O estado seguinte é o “Inicial”, a partir dele que o usuário seleciona entre as principais funções da secretária eletrônica, nele se define se o próximo estado será o “Reprodução completa”, onde seriam reproduzidas todas as mensagens salvas na memória em ordem de recebimento, “Navegação”, onde seria selecionada uma mensagem específica para ser reproduzida, onde em ambos os casos a mudança de estado é determinado pelo acionamento de botões pelo usuário, e além disso temos o direcionamento para o estado “Repouso”, responsável pela desativação do display, que é determinada a partir do não acionamento de nenhum dos botões por um minuto; Temos como principais ações no estado “Inicial” o acionamento do LED O, o qual informa que o sistema está ligado, assim como a ativação do display onde é possível visualizar o número de mensagens salvas na memória, e dependendo da existência de mensagens e do quantidade destas também podem ser acionados os LEDs F e M, responsáveis por sinalizar respectivamente memória cheia e a existência de alguma mensagem salva.

Ao acionar o botão Play a partir do estado inicial, será feita a transição por estado de “Reprodução completa”, onde continuará até que a leitura das mensagens seja concluída e voltará para “Inicial”; Já ao pressionar as teclas Back ou Next o sistema seguirá para o estado de “Navegação” onde selecionará uma das mensagens para executar pressionando o Play, o que nos levaria ao estado “Reprodução individual”, caso não se deseje selecionar mais nenhuma faixa individualmente é possível ir para o estado “Repouso” pressionando a tecla ON a partir de “Navegação”.

A partir de “Repouso” pode-se ir ao estado “Inicial” acionando o botão ON, ou “Gravando” no caso do recebimento de uma nova mensagem, permanecendo neste até o fim da gravação.

3.2 Implementação do *Datapath*

O bloco operacional foi iniciado pensando na problemática da navegação nas mensagens através do usuário e na memória a ser utilizada. Foi definido que a implementação fosse feita através de uma memória do tipo RAM, a qual possui como entradas: um dado, que no caso desse projeto, é uma mensagem com 10 amostras de 8 bits; possui também uma entrada que indica o endereço acessado dessa memória, sendo de 5 bits (25 mensagens no máximo); uma entrada de 1 bit para sinalizar se será realizada uma leitura ou escrita na memória, e por último, uma entrada também de 1 bit para atuar como *enable* do componente.

Uma vez que as mensagens gravadas na memória podem ser percorrida de forma individual, através do estado de navegação, e pode ser de forma contínua pelo estado de reprodução completa, foi pensado na implementação de um contador que indique a quantidade de mensagens gravadas até então (tendo no máximo 25) e outro que indique a mensagem que o usuário está lendo no momento.

Para isso, existe um contador/decrementador de leitura, o qual atua nos estados de navegação, sendo possível passar ou voltar uma mensagem, e um contador de escrita que incrementa sempre que uma mensagem é gravada na memória. O primeiro é composto por um somador e um subtrator, que conectam-se a um multiplexador de duas entradas. Esse multiplexador é conectado ao registrador de leitura, que guarda o valor do endereço da última mensagem lida. O registrador de leitura possui como *enable* a variável “en_r” recebida do bloco de controle, e possui como *reset* a variável “RESET_REG” recebida de um comparador de igualdade de 5 bits, o qual recebe 1 quando a saída do registrador de leitura for igual à do registrador de escrita, forçando o contador a ir para o endereço inicial.

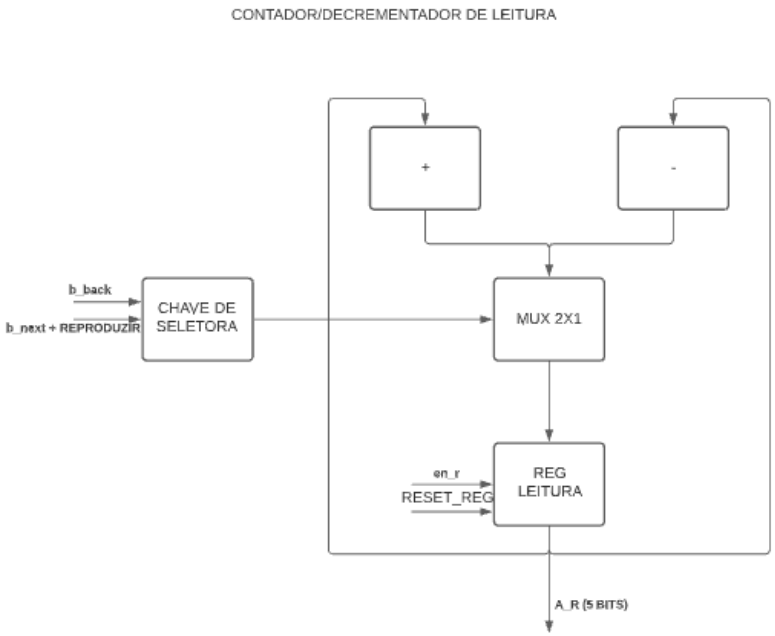


Figura 2: Contador/Decrementador de Leitura

A chave seletora do multiplexador citado se trata do botão *back* “b_back” para o caso do registrador receber o valor do subtrator, isto é, regredir a mensagem. E também tem o resultado lógico de uma porta *or* entre o sinal do botão *next* e a variável “REPRODUZIR”. A variável citada também é resultado de um comparador entre ambos os registradores, no entanto, ela recebe sinal lógico alto enquanto o registrador de leitura for menor que o de escrita, indicando no caso do estado de reprodução completa, que ainda há mensagens a serem lidas. Segue abaixo a representação do bloco do contador/decrementador de leitura:Figura 2: Bloco Contador/Decrementador de Leitura.

O bloco do contador de escrita segue a mesma ideia do de leitura, sendo que agora só é possível incrementar ou zerar o contador. O mux recebe então as saídas de dois somadores, os quais somam com 1 ou com zero dependendo da chave seletora do multiplexador. A seleção é feita através da variável “GRAVAR” obtida através do bloco de controle. Uma vez que “GRAVAR” = 1, a cada pulso de clock o contador incrementa um no endereço da memória.

O *enable* do registrador de escrita é a variável “en_w” e seu *reset* é através da variável “reset”, ambas obtidas do bloco de controle. Abaixo está e representação do bloco do contador de escrita:

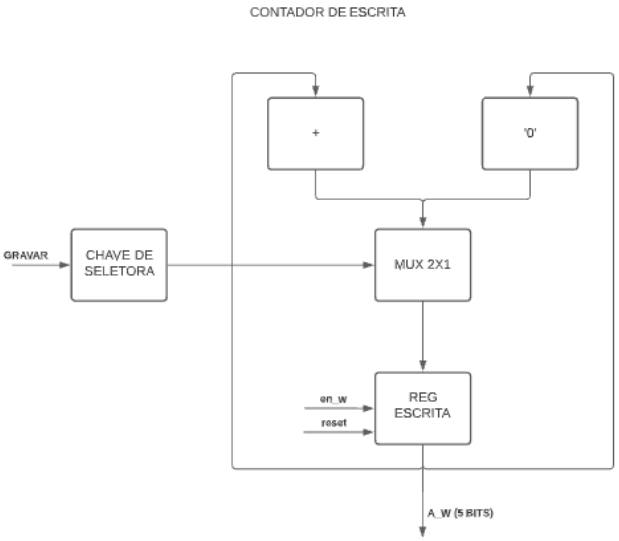


Figura 3: Bloco Contador de Escrita.

Vale salientar que os contadores citados servem apenas para identificar qual valor de memória será enviado para a entrada da memória RAM, para que a mesma possa ler ou escrever um determinado dado na alocação indicada. Como não é possível ler e escrever dados ao mesmo tempo, se faz necessário a utilização de um multiplexador que indique se o endereço de memória enviado é para leitura ou escrita. Veja a figura abaixo:

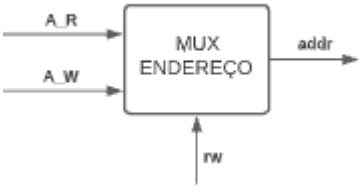


Figura 4: Multiplexador de Endereço.

Nesse multiplexador visto na figura acima, a saída “addr”, que se trata da entrada do endereço da memória RAM, recebe o valor do registrador de leitura ou o de escrita, sendo que a chave seletora é a variável “rw” advinda do bloco de controle. Essa variável recebe valor lógico 1 quando estiver lendo um dado, e 0 quando estiver escrevendo.

O bloco operacional também compõe 4 comparadores, sendo 2 deles referentes aos já citados, comparadores dos registradores, e também mais dois que atuam na resposta das variáveis “l_msg” e ”l_full”, as quais, respectivamente, acionam os leds que indicam que há mensagem na memória e que a memória está cheia. Veja abaixo a representação desses comparadores:

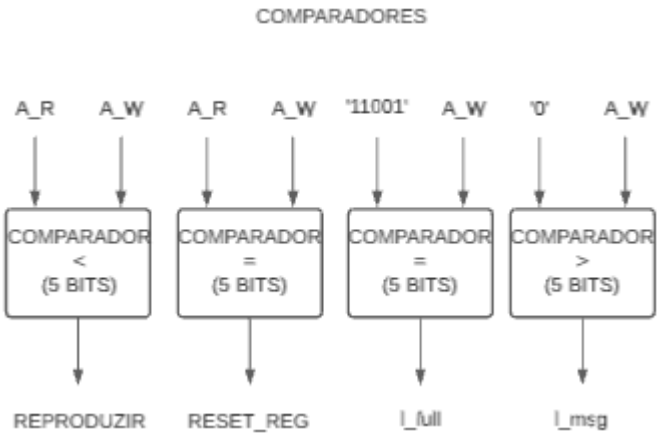


Figura 5: Comparadores.

Veja que para comparar se a memória está cheia, basta comparar o registrador de escrita com o valor binário equivalente ao número máximo de mensagens, o qual é 25 (11001). Já, para verificar se há mensagem, é comparado se o número no registrador de escrita é maior que 0.

Para a verificação dos tempos descritos na problemática, isto é, a contagem que define se a secretária entra em repouso (60 segundos), se a máquina vai desligar (5 segundos pressionando o botão ON) ou se será resetado o contador de escrita da memória (2 segundos pressionando o botão ON), foram implementados dois *timers* , um referente ao temporizador dos 60 segundos, descrito pela inatividade do usuário na tela de início, e outro para o botão pressionado. Veja abaixo o bloco comentado:

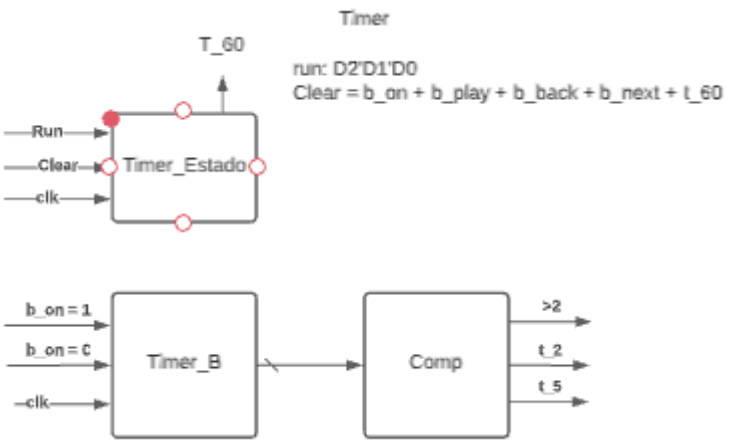


Figura 6: Temporizadores.

Veja que as entradas dos temporizadores estão de acordo com a ideia do projeto, e suas saídas são as variáveis de tempo que representam entradas no bloco de controle.

Por último, foi implementada a ideia dos displays do projeto. Como se tratam de dois displays de 7 segmentos cada, a alternativa encontrada foi enviar o valor binário do registrador de leitura a um conversor binário-bcd, que por sua vez, enviará os dados de 4 bits referentes à unidade e dezena a cada um do respectivo display. Essas saídas foram chamadas de “DISPLAY 1” e “DISPLAY 2”, que são variáveis de saídas do bloco operacional. Veja a imagem abaixo:

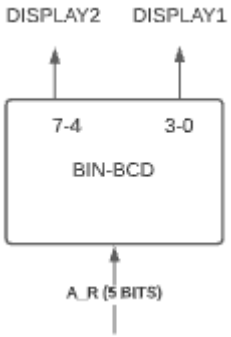


Figura 7: Conversor BIN-BCD e Displays.

3.3 Criação da FSM

Na máquina de estados de baixo nível, também conhecida como máquina de estados finita (FSM) é onde descrevemos as ações tomadas em cada em cada transição de estado, no caso da nossa onde usamos modelo Mealy, ou em cada um dos estados, no caso da adoção do modelo Moore, e a partir das definições das saídas e das entradas necessárias para as transições podemos montar equações lógicas para representar as variações de estado das variáveis.

A seguir temos o projeto de máquina de estados de baixo nível definido pelo nosso grupo, onde melhor detalhamos o sistema, mas por conta das dimensões da figura iremos subdividi-la para proporcionar uma melhor análise.

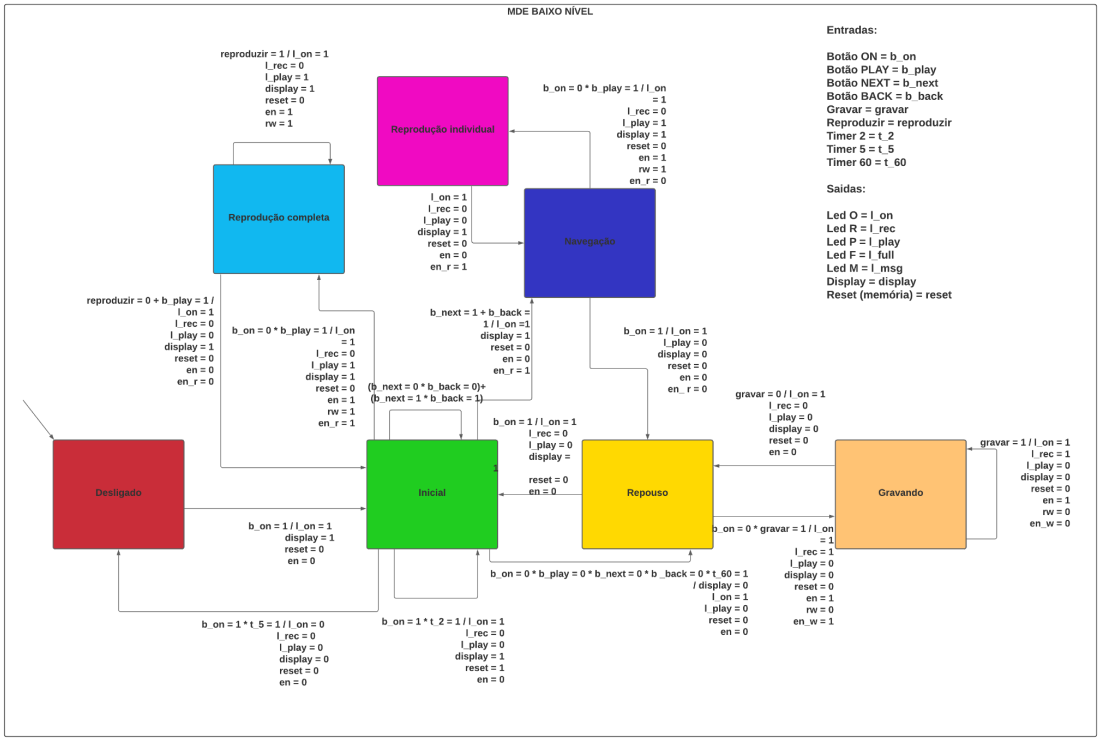


Figura 8: Máquina de estados de alto nível

3.3.1 Detalhamento número 1: Repouso por inatividade

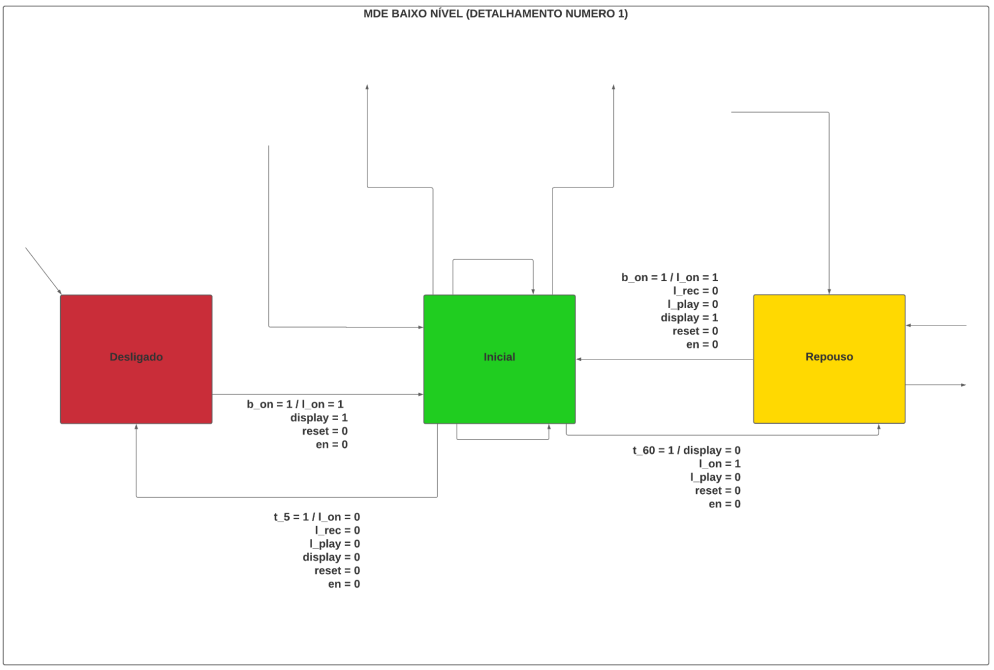


Figura 9: Detalhamento número 1: Repouso por inatividade

Na figura acima podemos ver a parte da máquina de estados responsável pelo acionamento e pelo repouso por inatividade.

Para irmos de “Desligado” para “Inicial” temos como entrada o botão ON, quando este for igual a 1 iremos mudar de estado, e durante esta transição

definiremos o LED O e o display como 1, valor que estamos usando para representar ligado, e os comandos de reset e enable como 0, valor que corresponde a desativado, a variável enable está sendo usada como habilitadora da utilização da memória, consequentemente como não efetua-se nenhuma ação na memória durante essa mudança de estados ela está desativada.

Para irmos do estado “Inicial” ao “Desligado” precisamos pressionar o botão ON por 5 segundos, informação esta que obtemos pela variável t_5 a qual vem de um sistema do datapath, indo de “Desligado” a “Inicial” ativamos os LEDs e o display e ao voltar fazemos o procedimento inverso.

Usamos a variável t_60 a qual é oriunda de um sistema com temporizador no datapath para sinalizar a inação por um minuto no estado “Inicial”, e ao esta ter valor 1 a máquina vai para o estado “Repouso” onde o display é desativado e ao voltar o reativamos, a condição que utilizamos para a volta é o acionamento do botão ON.

3.3.2 Detalhamento número 2: Gravação de mensagem

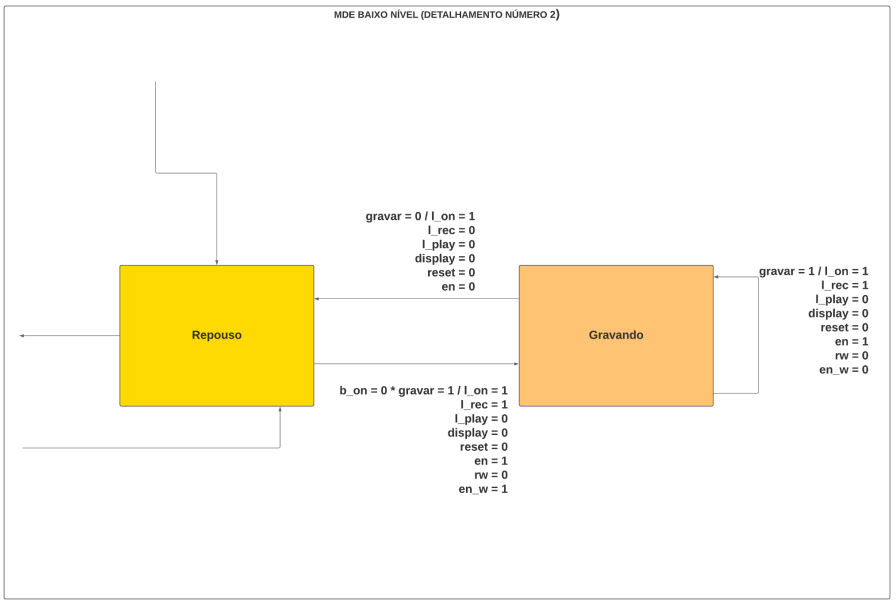


Figura 10: Detalhamento número 2: Gravação de mensagem

Do “Repouso” para “Gravando” temos como condição de ida a chegada de uma mensagem e durante este momento o botão ON não estar pressionado, e para a volta temos como condição a variável gravar a qual corresponde a chegada de uma mensagem se tronar 0, enquanto isso não acontecer continuamos no estado “Gravando”. As mudanças observáveis nas saídas são o acionamento do LED R enquanto no estado “Gravando” e também é habilitada a memória por meio de en = 1 no modo de gravação por meio de rw = 0 e en_w =0.

3.3.3 Detalhamento número 3: Reset do Sistema

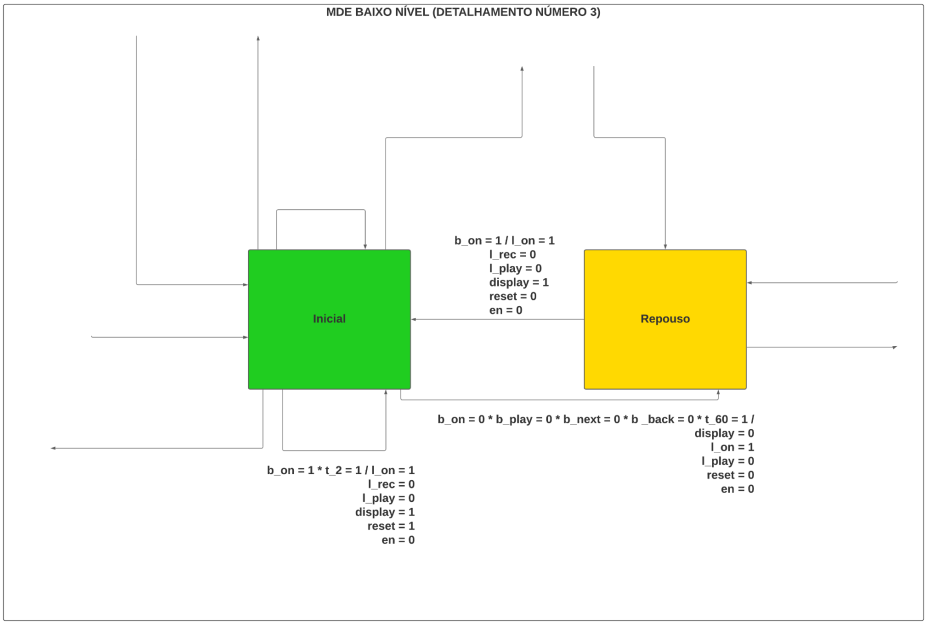


Figura 11: Detalhamento número 3: Reset do Sistema

Para que a memória do sistema seja resetada é preciso que o botão ON seja pressionado por um período de 2 segundos, quando no estado “Inicial”, para saber se este requisito foi atendido utilizamos a entrada t_2 oriunda de um sistema de timer e comparador presente no datapath.

Ao atender-se os requisitos os LEDs são desligados com exceção do LED O e a memória é apagada.

3.3.5 Detalhamento número 4: Reprodução total

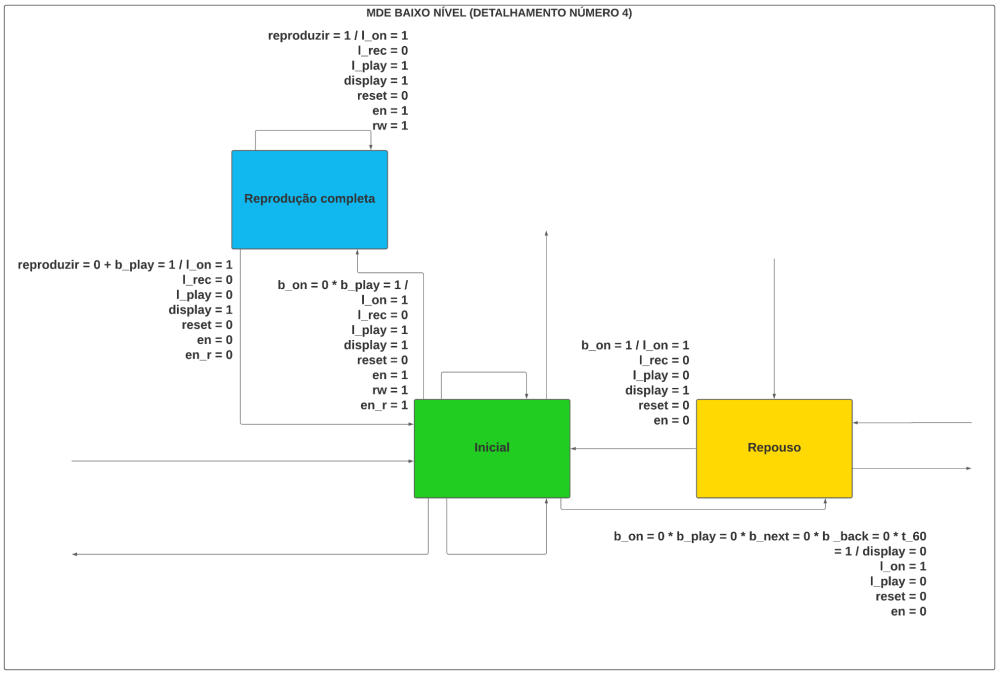


Figura 12: Detalhamento número 4: Reprodução total

Para que o sistema faça a reprodução total é preciso estar no estado “Inicial” e nele pressionar o botão Play, chamado de b_play, por meio da variável reproduzir avaliamos se a reprodução foi concluída ou não para que assim possamos voltar ao estado anterior, caso o botão Play seja acionado durante a reprodução também voltaremos para o estado anterior.

Durante a execução o LED P estará aceso e a memória estará ativa no modo de leitura por meio das variáveis en = 1 e rw = 1, ao finalizar a leitura ambas as variáveis e o LED são desativados, assim como reproduzir fica = 0.

3.3.6 Detalhamento número 5: Reprodução individual e reprodução individual com memória cheia.

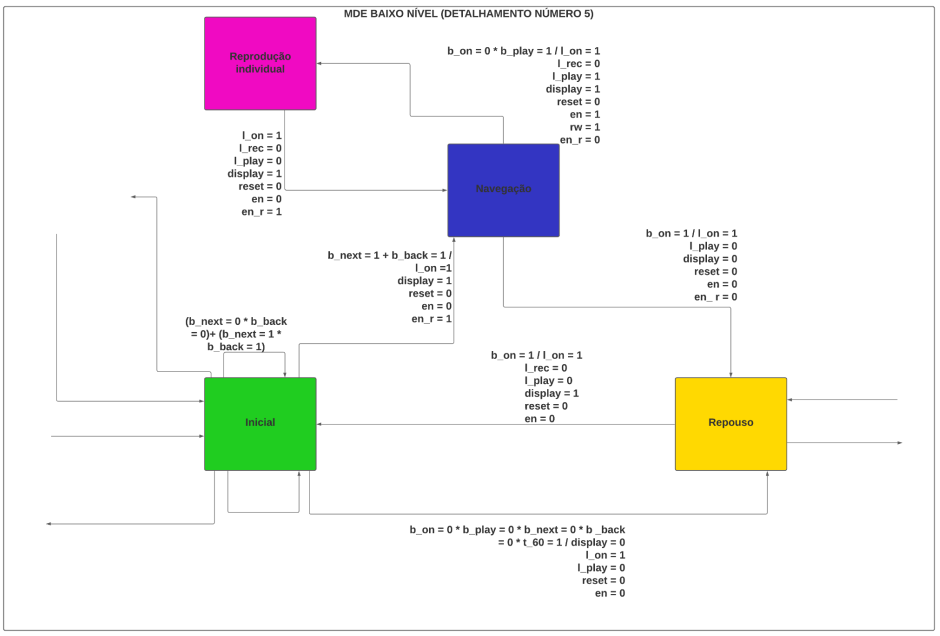


Figura 13: Detalhamento número 5: Reprodução individual e reprodução individual com memória cheia.

Para executarmos a função de reprodução individual é preciso pressionar ou o botão Next ou o Back, os quais possuem os nomes b_next e b_back respectivamente, ao pressionarmos um destes iremos ir do estado “Inicial” ao estado “Navegação”, a lógica responsável por incrementar e decrementar o número da amostra selecionada é feito no datapath por meio de um incrementador/decrementador, ao chegarmos no número que pretendemos ouvir, deve-se pressionar o botão Play (b_play), mudando para o estado “Reprodução Individual” e retornando ao “Navegação” logo em seguida.

Enquanto a mensagem estiver sendo reproduzida o LED P (led_p) estará acionado e a memória estará habilitada para leitura com en =1, rw = 1. Também vale comentar a variável en_r que aciona o mecanismo de navegação no datapath quando for igual a 1.

3.4 Implementação do Bloco de controle

Criada a FSM e em conhecimento das entradas e saídas existentes para a conexão entre o *datapath* e o bloco de controle, foi montada a arquitetura padrão deste para melhor visualizar o processo como mostrado na **figura x** abaixo. Nesta estão dispostos os 3 bits relacionados aos estados, onde Q2, Q1 e Q0 possuem o valor correspondente ao estado atual e D2, D1 e D0 correspondente ao estado futuro. O bloco denominado "Lógica combinacional" contém a implementação das equações booleanas que serão desenvolvidas a partir da tabela-verdade posteriormente.

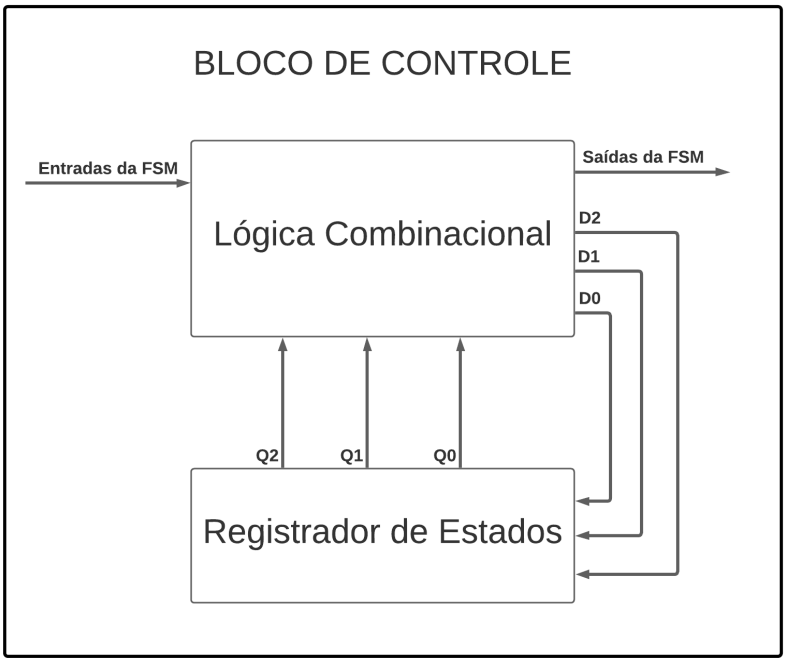


Figura 14: Arquitetura padrão do bloco de controle

Em conhecimento da arquitetura padrão, os passos seguintes para a construção do bloco de controle é a codificação dos estados e criação da tabela-verdade que relacione os estados atuais/futuros com as entradas/saídas do sistema. A **figura y** é a parte 1 da tabela-verdade onde se relacionam os estados atuais e suas respectivas entradas para a geração da tabela-verdade onde se relaciona os estados futuros e suas saídas (**figura z**).

ESTADO	ESTADO ATUAL			ENTRADAS								
	Q2	Q1	Q0	b_on	b_play	b_back	b_next	gravar	reproduzir	T_2	T_5	T_60
DESLIGADO	0	0	0	0	x	x	x	x	x	x	x	x
DESLIGADO	0	0	0	1	x	x	x	x	x	x	x	x
INICIAL	0	0	1	0	0	0	0	x	x	x	x	1
INICIAL	0	0	1	1	x	x	x	x	x	x	1	x
INICIAL	0	0	1	1	x	x	x	x	x	1	x	x
INICIAL	0	0	1	0	1	x	x	x	x	x	x	x
INICIAL	0	0	1	0	0	1	x	x	x	x	x	x
INICIAL	0	0	1	0	0	x	1	x	x	x	x	x
REPOUSO	0	1	0	1	x	x	x	x	x	x	x	x
REPOUSO	0	1	0	0	x	x	x	1	x	x	x	x
REPRODUÇÃO COMPLETA	0	1	1	x	x	x	x	x	1	x	x	x
REPRODUÇÃO COMPLETA	0	1	1	x	x	x	x	x	0	x	x	x
REPRODUÇÃO COMPLETA	0	1	1	x	1	x	x	x	x	x	x	x
NAVEGAÇÃO	1	0	0	1	x	x	x	x	x	x	x	x
NAVEGAÇÃO	1	0	0	0	1	x	x	x	x	x	x	x
GRAVANDO	1	0	1	x	x	x	x	1	x	x	x	x
GRAVANDO	1	0	1	x	x	x	x	0	x	x	x	x
REPRODUÇÃO INDIVIDUAL	1	1	0	x	x	x	x	x	x	x	x	x

Figura 15: Parte 1 da tabela-verdade

ESTADO	ESTADO FUTURO			SAÍDAS								
	D2	D1	D0	l_on	l_rec	l_play	display	reset	en	rw	en_r	en_w
DESLIGADO	0	0	0	0	0	0	0	0	0	0	0	0
INICIAL	0	0	1	1	0	0	1	0	0	x	0	0
REPOUSO	0	1	0	1	x	0	0	0	0	x	x	x
DESLIGADO	0	0	0	0	0	0	0	0	0	x	x	x
INICIAL	0	0	1	1	0	0	1	1	0	x	x	x
REPRODUÇÃO COMPLETA	0	1	1	1	0	1	1	0	1	1	1	x
NAVEGAÇÃO	1	0	0	1	0	0	1	0	0	x	1	x
NAVEGAÇÃO	1	0	0	1	0	0	1	0	0	x	1	x
INICIAL	0	0	1	1	0	0	1	0	0	x	x	x
GRAVANDO	1	0	1	1	1	0	0	0	1	0	x	1
REPRODUÇÃO COMPLETA	0	1	1	1	0	1	1	0	1	1	x	x
INICIAL	0	0	1	1	0	0	1	0	0	x	0	x
INICIAL	0	0	1	1	0	0	1	0	0	x	0	x
REPOUSO	0	1	0	1	x	0	0	0	0	x	0	x
REPRODUÇÃO INDIVIDUAL	1	1	0	1	0	1	1	0	1	1	0	x
GRAVANDO	1	0	1	1	1	0	0	0	1	0	x	0
REPOUSO	0	1	0	1	0	0	0	0	0	x	x	x
NAVEGAÇÃO	1	0	0	1	0	0	1	0	0	x	1	x

Figura 16: Parte 2 da tabela-verdade

A partir das tabelas acima, são geradas as equações booleanas **(figura w)** que definem o funcionamento do bloco "lógica combinacional" existente dentro da arquitetura padrão do bloco de controle. São essas equações que executam as mudanças de estados e saídas de acordo com as entradas fornecidas. Ou seja, são essas equações que fornecem os valores de sinais corretos para o funcionamento do *datapath*.

LÓGICA COMBINACIONAL DAS SÁIDAS	
SAÍDAS	EQUAÇÃO BOOLEANA
D2	~Q2 ~Q1 Q0 ~B_ON ~B_PLAY B_NEXT + ~Q2 ~Q1 Q0 ~B_ON ~B_PLAY B_BACK + Q1 ~Q0 ~B_ON GRAVAR + Q2 ~Q0 ~B_ON B_PLAY + Q2 ~Q1 Q0 GRAVAR + Q2 Q1 ~Q0
D1	~Q2 ~Q1 Q0 ~B_ON ~B_BACK ~B_NEXT T_60 + ~Q2 ~Q1 Q0 ~B_ON B_PLAY + ~Q2 Q1 Q0 REPRODUZIR + Q2 ~Q1 ~Q0 B_PLAY + Q2 ~Q1 ~Q0 B_ON + Q2 ~Q1 Q0 ~GRAVAR
D0	~Q2 ~Q0 B_ON + ~Q2 B_ON ~T_2 + ~Q2 Q0 ~B_ON B_PLAY + ~Q2 Q1 GRAVAR + ~Q2 Q1 Q0 + Q2 ~Q1 Q0 GRAVAR
L_ON	B_ON T_2 + ~Q2 Q0 ~B_ON T_60 + ~Q1 Q0 ~B_ON B_NEXT + ~Q1 Q0 ~B_ON B_BACK + ~Q2 Q0 ~B_ON B_PLAY + ~Q2 Q1 GRAVAR + ~Q2 Q1 Q0 + Q2 ~Q1 B_PLAY + Q2 ~Q1
L_REC	~Q2 Q1 ~Q0 ~B_ON GRAVAR + Q2 ~Q1 Q0 GRAVAR
L_PLAY	~Q2 ~Q1 Q0 ~B_ON B_PLAY + ~Q2 Q1 Q0 REPRODUZIR + Q2 ~Q1 ~Q0 ~B_ON B_PLAY
DISPLAY	~Q2 ~Q0 B_ON + ~Q2 B_ON T_2 + ~Q2 Q0 ~B_ON B_NEXT + ~Q2 Q0 ~B_ON B_BACK + ~Q2 Q0 ~B_ON B_PLAY + ~Q2 Q1 Q0 + Q2 ~Q0 ~B_ON B_PLAY + Q2 Q1 ~Q0
RESET	~Q2 ~Q1 Q0 B_ON T_2
EN	~Q2 ~Q1 Q0 ~B_ON B_PLAY + ~Q2 Q1 ~Q0 ~B_ON GRAVAR + ~Q2 Q1 Q0 REPRODUZIR + Q2 ~Q1 ~Q0 ~B_ON B_PLAY + Q2 ~Q1 Q0 GRAVAR
RW	~Q2 ~Q1 Q0 ~B_ON B_PLAY + ~Q2 Q1 Q0 REPRODUZIR + Q2 ~Q1 ~Q0 ~B_ON B_PLAY
EN_R	~Q2 ~Q1 Q0 ~B_ON B_NEXT + ~Q2 ~Q1 Q0 ~B_ON B_BACK + ~Q2 ~Q1 Q0 ~B_ON B_PLAY + Q2 Q1 ~Q0
EN_W	~Q2 Q1 ~Q0 ~B_ON GRAVAR

Figura 17: Equações booleanas do bloco de controle

Finalizada a construção do bloco de controle, encerra-se o desenvolvimento do projeto RTL, criando o projeto de um circuito integrado do sistema digital da secretária eletrônica proposta. Abaixo está a relação entre o bloco de controle e o bloco operacional:

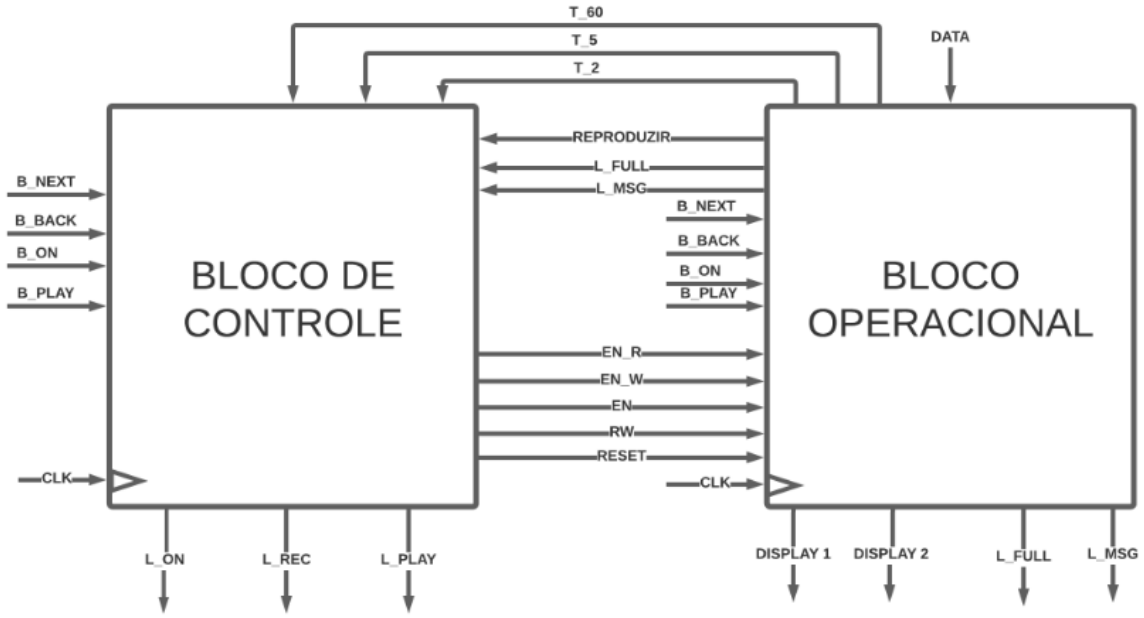


Figura 18: Bloco de Controle e Bloco Operacional.

3.5 Resultados e Conclusões

Com base no que foi apresentado ao decorrer do trabalho, é possível observar a criação de todo o projeto do circuito integrado de um sistema digital para a secretária eletrônica proposta. Foram utilizados conceitos e métodos explicados neste trabalho com base na literatura referenciada e a partir deste projeto é possível a implementação em VHDL do problema para visualização do funcionamento em tempo real, atingindo assim o propósito final deste trabalho.

Referências

[1] VAHID, Frank. **Sistemas Digitais: projeto, otimização, hdls.** São Paulo: Bookman, 2008.

[2] TOCCI, Ronald J.. **Sistemas Digitais: princípios e aplicações.** São Paulo: Pearson, 2011.

Anexos

A.1 Equipe

ANEXO A - Relato semanal

Líder: Albertho Síziney Costa

Tabela 1 - Identificação da equipe.

FUNÇÃO	INTEGRANTE
Redator	Anny Pinheiro
Debatedor	Allysson Andrade
Videomaker	Isaac de Lyra
Auxiliar	-

Fonte: Autores.

A.2 O problema

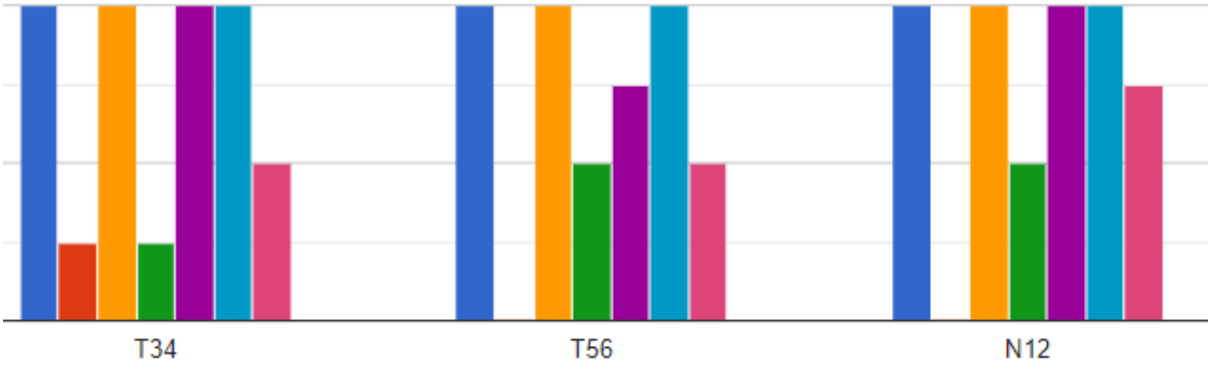
A problemática se volta para a realização de um projeto que, a partir dele, seja possível implementar um sistema de uma secretária eletrônica. Esse sistema possui funções de ler e escrever dados digitais em uma memória, a qual, para esse projeto, se trata de uma memória RAM. Além disso, deve ser possível dar *play* e *pause* na execução dos dados, e *next* ou *back* para navegar nas memórias gravadas.

Para a solução, nos foi requisitada a existência de certas telas, como: repouso, inicial, desligado, navegação e etc. O sistema conta também com dois displays de 7 segmentos para sinalizar a mensagem analisada, e conta com leds de sinalização de existência de mensagens, memória cheia, reprodução de áudio, gravação e funcionamento.

A.3 Registro do brainstorming

A primeira reunião ocorreu na Terça-Feira (08/06), e nela foi decidido os cargos de cada um e foi comentada a disponibilidade de horário de cada integrante. Foi elaborado pelo líder um questionário com os melhores dias e horários para que fosse possível adequar o brainstorming à todos. Segue o resultado gráfico:

Gráfico 1 - Melhores horários.



Fonte: Autores.

Da esquerda para a direita, cada coluna representa o dia da semana iniciando pela Segunda-Feira. Foi decidido que os horários seriam predominantemente à noite.

Na Quarta-Feira todos os componentes estudaram o projeto e iniciaram as pesquisas de forma individual. Já, na Quinta-Feira, nos encontramos para discutir o que foi encontrado. Anny e Allysson apresentaram dois modelos de máquina de estados, o líder e Isaac iniciaram a passagem das ideias para um arquivo compartilhado no drive. Na Sexta-Feira e no Sábado o líder e Allysson fizeram a MDE de baixo e alto nível, e montaram a tabela de estados. No domingo e na Segunda-Feira (14/06) foram feitas as últimas adequações e correções do projeto.

A.4 Pontos-chaves

O ponto chave desse projeto foi pensar em qual tipo de memória seria utilizado, e qual a sua dimensão. Foi pensado também a forma de tratamento dos dados lidos e gravados na memória, onde foi decidido que seria tratada de mensagem em mensagem, isto é, a cada 10 amostras.

A.5 Questões de pesquisa

Foi necessário relembrar os conceitos de máquinas de estados, bem como o sistema RTL. Além disso, foram vistas algumas implementações da utilização da memória RAM em exemplos do livro VAHID (2008). Para o melhor entendimento da problemática, o grupo também buscou compreender a ideia dos conversores digitais e analógicos através de pesquisas rápidas na internet.

A.6 Planejamento da pesquisa

Predominantemente, foram utilizadas as bibliografias de VAHID (2008) e TOCCI (2011), para o melhor entendimento do projeto. As pesquisas foram realizadas, em sua maior parte, de forma individual, e cada um apresentou suas ideias durante debates em grupo, por meio do aplicativo Discord.