TEC 498 - MI- PROJETO DE CIRCUITOS DIGITAIS 2019.1
Relatório – Batalha Naval Digital

Ian Zaque Pereira de Jesus dos Santos[1], Marco Antonio Silva Rios de Almeida[2], Weslei Santos Pinheiro[3]
Curso de Bacharelado em Eng. de Computação – Universidade Estadual de Feira de Santana
(UEFS) – Campus Feira de Santana
CEP 44.036-900 – Feira de Santana – BA – Brasil

ianzaque.uefs@gmail.com[1], marco12y@outlook.com[2], weslei200912@hotmail.com[3]

1.Resumo

Este relatório tem o objetivo de apresentar e detalhar uma possível solução para o desenvolvimento de um produto baseado no jogo batalha naval. O produto - a solução do problema proposto - está totalmente esclarecida e explicada nos segmentos deste relatório técnico. É explicitado o funcionamento geral do circuito e de cada componente lógico destas combinações lógicas, assim como os conceitos que fundamentam o estudo dos autores sobre circuitos digitais. A fim de se ter um entendimento sobre o que foi produzido, é necessário uma leitura desta exposição e a visualização e teste do produto final em laboratório apropriado.

2.Introdução

A pedido da empresa InovaTEC498 S.A., grupo científico industrial atuante no mercado de produção de sistemas e circuitos digitais integrados, um produto foi requisitado. Este produto se trata de um circuito digital (programado no software ALTERA Quartus II 9.0) que implementasse de modo preciso o jogo, bastante popular, batalha naval. O circuito em questão deve ser implementado em *FPGA* hardware reconfigurável usando o kit de desenvolvimento ACEX1K e de seus periféricos. Um destes periféricos refere-se à matriz de LEDs. Tais LEDs são emissores de luz vermelha fabricados em materiais diversos e são vantajosos dentro de determinado contexto, tal como o contexto do problema. Entre estas vantagens estão a diversidade de cores, tamanho reduzido, bom custo-benefício, respostas rápidas aos comandos etc.

O solicitado descrito - com informações mais específicas nos demais tópicos do relatório - foi proposto a alguns alunos da Universidade Estadual de Feira de Santana (UEFS). Estes alunos, cursando o M.I (Módulo Integrado), que engloba a disciplina teórica (TEC401 Circuitos Digitais) e a prática (TEC498 Projetos de Circuitos Digitais), devem resolver o problema proposto utilizando os materiais citados acima e seguindo os requisitos da InovaTEC498 presentes na descrição da folha do projeto.

Para que houvesse um devido esforço de mais vigor, foi usada a metodologia de aprendizagem PBL (Problem Based Learning) a qual visa desenvolver o aluno quanto à participação em discussões, evolução do estudo e absorção de conhecimento mais consistentemente.

3.Fundamentação teórica

Sistemas digitais tornaram-se parte fundamental no processo evolutivo tecnológico. Segundo Tocci (2000,pg.6) sistema digital "é uma combinação de dispositivos projetados para manipular informação lógica ou quantidades físicas representadas no formato digital", ou seja, é um sistema que busca quantificar valores de forma discreta e precisa. Os sistemas digitais são de supra importância nos diversos setores da sociedade, desde o relógio de pulso até grandes maquinários industriais. Em contrapartida ao sistema digital existe o sistema analógico, que é representado por uma "grandeza que apresenta valores contínuos" (FLOYD, pg.20) e representa valores que podem ser medidos quantitativamente na natureza. Uma das característica dos sistemas analógicos é a variação, uma exemplo da representação analógica são as ondas de rádios ou ondas de som emitidas por um microfone.

Os sistemas digitais possuem algumas características que os qualificam como sistemas mais adequados, são eles: eficiência no processamento; confiabilidade no processamento de informações; armazenamento e transcrição de dados (Floyd, pg.20). Contudo, o mundo é analógico e devido a isso se faz necessária uma comunicação direta entre estes sistemas, comumente é necessária a transcrição de um sistema para o outro por meio de codificadores e decodificadores, um exemplo é o aparelho de rádio que recebe ondas de rádio (analógico) e as converte para um som audível por meio de sistemas digitais, várias outras aplicações também se utilizam dessa mesma dinâmica entre os sistemas.

Um sistema digital eletrônico é composto basicamente por 1's e 0's que representam estados (*on/off, true/false*) que quando aplicados a determinada lógica realizam funções específicas. Esses estados fazem parte do sistema binário de numeração, nele cada estado lógico é nomeado de "bit" e o conjunto de 8 bits é considerado um "byte", um circuito eletrônico digital interpreta esses estados como sendo níveis de tensão auto (1) e baixo (0) que em conjunto formam as diretrizes do sistema.

Um sistema digital é composto por diversos circuitos lógicos, que são conjuntos de expressões lógicas. Existem circuitos digitais denominados de portas lógicas que representam operações lógicas básicas entre binários, são eles: OR (OU), AND (E), NOT (NÃO), NOR (NÃO-OU), NAND (NÃO-E). Figura 1 abaixo apresenta os símbolos das portas lógicas.

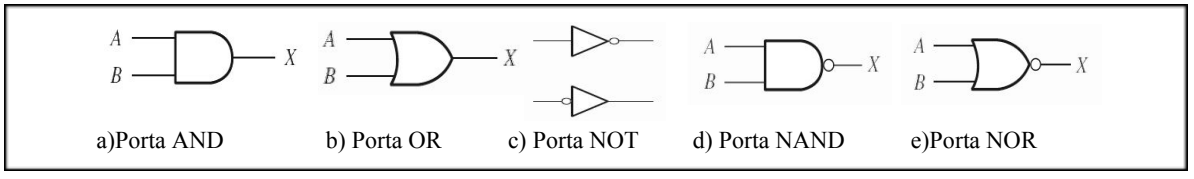

Figura 1 – Portas lógicas AND, OR, NOT, NAND e NOR

Fonte: Sistemas Digitais - fundamentos e aplicações.

A adição entre binários é feita por meio da porta OR, quando uma das variáveis assume o valor 1 o resultado da operação será sempre 1, a representação algébrica para duas variáveis é S=A+B (Tocci, 2000, pg.51). A tabela 1.b demonstra a tabela-verdade de uma porta OR. A multiplicação é realizada por intermédio de portas AND, quando uma das variáveis assume valor 0 a saída será sempre 0, quando as duas variáveis assumem valor 1, a saída será sempre 1, sua representação algébrica se dá pela expressão S=A.B. Tabela 1.a demonstra a tabela-verdade de porta AND (Tocci, 2000, pg.55).

Tabela 1 –Tabela-verdade das operações de adição e multiplicação

ENTRADAS A	B	SAÍDA X
0	0	0
0	1	0
1	0	0
1	1	1

1 = ALTO, 0 = BAIXO

a)Tabela verdade porta AND.

ENTRADAS A	B	SAÍDA X
0	0	0
0	1	1
1	0	1
1	1	1

1 = ALTO, 0 = BAIXO

b)Tabela verdade porta OR.

Fonte: Sistemas Digitais - fundamentos e aplicações.

As expressões algébricas possuem algumas leis que permitem sua manipulação são elas leis associativas em que A + (B+C) = (A+B) + C e A(BC) = (AB)C, leis comutativas em que A+B=B+A e AB=BA, por fim há também as leis distributivas onde A(B+C)= AB+AC (Flody,2007,pg.202). Além dessas leis, existem também algumas regras booleanas que permitem a simplificação de um circuito. Ver Figura 2 que demonstra regras de simplificação na álgebra booleana.

1. $A + 0 = A$	**7.** $A \cdot A = A$
2. $A + 1 = 1$	**8.** $A \cdot \overline{A} = 0$
3. $A \cdot 0 = 0$	**9.** $\overline{\overline{A}} = A$
4. $A \cdot 1 = A$	**10.** $A + AB = A$
5. $A + A = A$	**11.** $A + \overline{A}B = A + B$
6. $A + \overline{A} = 1$	**12.** $(A + B)(A + C) = A + BC$

A, *B* ou *C* podem representar uma única variável ou uma combinação de variáveis.

Figura 2: Regras da álgebra booleana.

Fonte: Sistemas Digitais - fundamentos e aplicações.

Outra forma de simplificação bastante utilizada é o teorema de DeMorgan, ele possui dois teoremas:

- 1º Teorema diz que o complemento de um produto de variáveis é igual à soma dos complementos das variáveis *(XY)' = X' + Y'* (Floyd, pg 207).
- 2º Teorema diz que o complemento de uma soma de variáveis é igual ao produto do complemento das variáveis *(X + Y)' = X' Y'* (Floyd, pg 207).

Existem algumas portas até então não mencionadas, chamadas de portas especiais são elas OU EXCLUSIVO e COINCIDÊNCIA. A porta OU EXCLUSIVO consiste em fornecer nível lógico alto (1) na saída de combinação de variáveis diferentes. Em contrapartida a porta COINCIDÊNCIA faz a operação contrária fornecendo nível lógico alto (1) na saída de variáveis idênticas. Essas portas são muito importantes nos circuitos combinacionais, alguns componente como somadores e comparadores fazem o seu uso. Ver figura 3 abaixo que demonstra simbologia dessas portas.

Figura 3: Portas lógicas OU EXCLUSIVO e COINCIDÊNCIA.

Fonte: Sistemas Digitais - fundamentos e aplicações.

Com o intuito de representar a relação entre as entradas e saídas em um circuito lógico, é criada uma tabela que contém essas devidas informações, essa tabela é descrita como tabela-verdade, ela auxilia na construção de circuitos lógicos e na simplificação dos mesmos. Ver Tabela 2 que demonstra tabela-verdade de duas variáveis. O número de combinações entre as estradas é dado por 2^n , em que *n* representa a quantidade de entradas no circuito.

A saída depende da expressão booleana entre as entradas. Segundo Tocci (2000,pg.60) as regras a seguir têm de ser obedecidas quando avaliamos uma expressão booleana:
1. Primeiro, realize as inversões de termos simples; ou seja, 0’ = 1 ou 1’ = 0.
2. Em seguida, realize as operações dentro de parênteses.
3. Realize as operações AND antes das operações OR, a menos que os parênteses indicam o contrário.
4. Se uma expressão tiver uma barra sobre, realize a operação indicada pela expressão e, em seguida, inverta o resultado.

TABELA 2: Tabela verdade de duas variáveis.

TABELA VERDADE DE DUAS VARIÁVEIS		
ENTRADAS		SAÍDAS
A	B	X
0	0	1
0	1	0
1	0	1
1	1	0

Fonte: Próprio autor.

Dado o circuito representado na Figura 4, a expressão booleana resultante é S = (A.B) + (C’.D), na S1 (saída 1) tem se a expressão (A.B), na S2 (saída 2) tem se a expressão (C’.D) e na S3 (saída 3) tem se a expressão resultante S = (A.B) + (C’.D). Assim como a partir do esquemático de um circuito é possível retirar a expressão booleana, o caminho contrário também pode ser realizado, e a partir de uma expressão booleana é possível à elaboração do circuito lógico, o método consiste em identificar as portas lógicas, obedecendo às regras para expressões booleanas descritas anteriormente e por fim desenhar as ligações correspondentes.

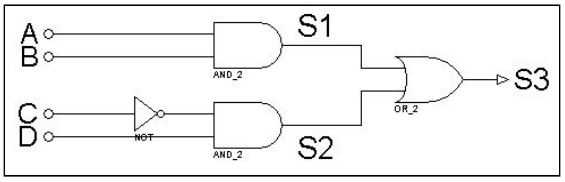

Figura 4 – Circuito combinacional.

Um dos pontos mais importantes na eletrônica digital são os circuitos combinacionais. Neles a saída depende única e exclusivamente das combinações das entradas do circuito (Idoeta, pg.157). Entre os circuitos combinacionais existentes pontuam-se os somadores, comparadores, codificadores e decodificadores.

Somador Completo

O *somador completo* é utilizado para a soma de binários de um algarismo. Ele possui três entradas, duas entradas dos algarismos e um *carry*. Possui, ainda, duas saídas uma o resultado da operação e a outra um *carry* gerado pela soma. Seu circuito lógico é construído a partir de uma porta XOR que soma os bits de entrada e outra porta XOR para somar o *carry* de entrada com o resultado da soma da porta anterior, os *carrys* são gerados através de portas ANDs e são somados por meio de uma porta OR para obter-se o *carry* de saída. É possível utilizar somadores completos em paralelos para operações com mais de um algarismo. Ver figura que demonstra representação lógica de um meio somador completo.

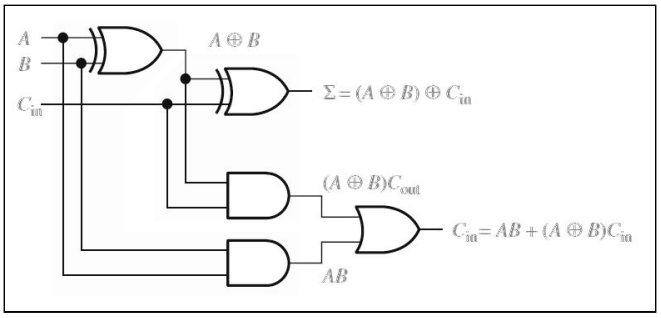

Figura 5- Circuito somador completo.

Fonte: Sistemas Digitais - fundamentos e aplicações.

Comparadores

A função de um *comparador* é determinar a igualdade entre números binários, definindo se são iguais ou diferentes. Existem *comparadores* que determinam a magnitude dos números e são chamados de comparadores de magnitude. O circuito de um *comparador* básico consiste em utilizar da porta EX- OR para determinar se dois algarismos são iguais, pois está porta define nível lógico alto (1) na saída caso os valores sejam diferente, através de inversores e uma porta AND é possível determinar a igualdade entre os números binários. Ver figura 6 que demonstra circuito de um comparador básico.

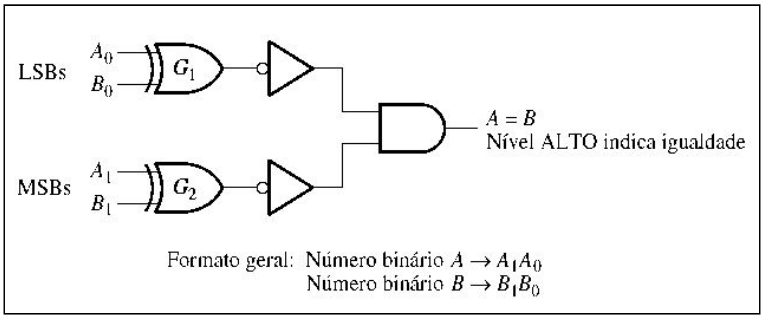

Figura 6: Comparador básico.

Fonte: Sistemas Digitais - fundamentos e aplicações.

Decodificadores

Os *decodificadores* possuem a função de receber um determinado código de entrada e gerar uma saída que representa esse mesmo código. Exemplos de decodificadores são *binário/decimal*, *BCD*, *4 bits* entre diversos outros. Os *decodificadores* podem ser utilizados em diversas aplicações, um exemplo dela é na seleção de periféricos em um computador. O processo para elaboração de um decodificador consiste montar a tabela-verdade e desenvolver as funções de decodificação, manipulando as entradas para se obter a saída desejada. Figura 7 demonstra *decodificador* binário básico na qual recebe um valor binário 1001 e deve repassar um valor alto (1) na sua saída.

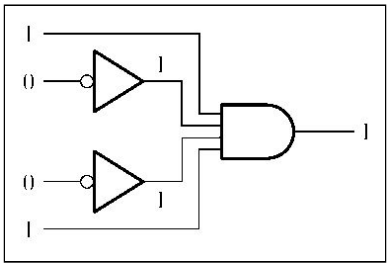

Figura 7: Decodificador binário básico.
Fonte: Sistemas Digitais - fundamentos e aplicações.

Codificadores

Os codificadores realizam o processo inverso ao decodificador. Ele torna possível a passagem de um código conhecido para um código desconhecido. Um exemplo prático são as entradas em decimais de um teclado que devem ser codificadas em binário para ser processada pelo sistema eletrônico. Alguns exemplos de codificadores são codificadores decimal para BCD e o codificador de 8 linhas para 3 linhas. Figura 8 mostra Codificador decimal/BCD.

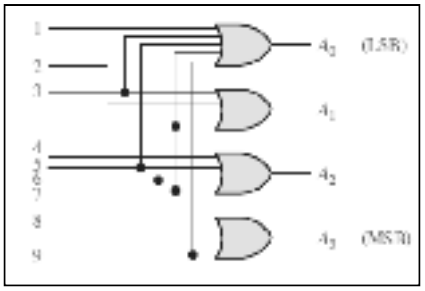

Figura 8: Codificador decimal para BCD.

Fonte: Sistemas Digitais - fundamentos e aplicações.

Com a necessidade da utilização de vários componentes eletrônicos na elaboração de sistemas digitais surgiu o desenvolvimento de dispositivos de lógica programável, que em comparação a CIs de funções fixas conseguem utilizar bem menos espaços, pode também ser reconfigurado com mais facilidade, além de possuir um melhor custo benefício. Existem diversos tipos de lógicas programáveis, elas se dividem em duas categorias **PLD** (*programmable logic device*) e **FPGA** (*field programmable gate array*), os PLDs se subdividem em outras duas categorias conforme demonstra a figura 9 abaixo.

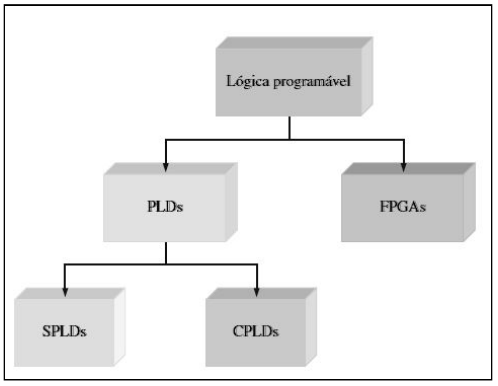

Figura 9: Lógica programável. Fonte: Sistemas Digitais - fundamentos e aplicações.

Os SPLDs são dispositivos simples na qual podem substituir até dez CIs de funções físicas, já os CPLDs são compostos por vários SPLDs aumentando assim sua capacidade e desempenho. As FPGAs possuem uma lógica diferenciada, pois elas possuem uma estrutura formada de blocos lógicos, blocos de I/O (entrada/saída) e interconexões programáveis. Esses dispositivos recebem as informações por meio de conexão com computador, na qual possua um software para desenvolvimento da lógica programável, o passo a passo do processo é demonstrado na figura 10 abaixo.

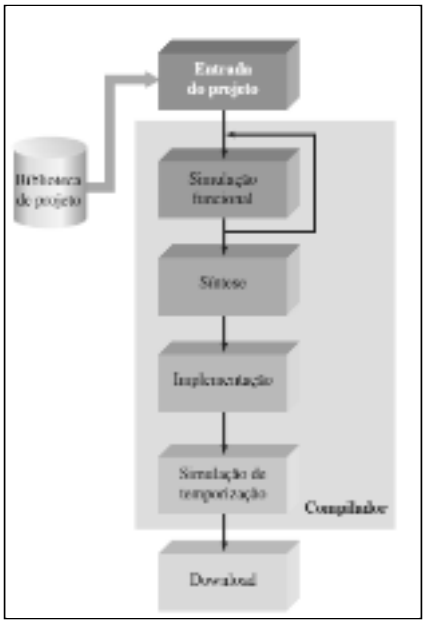

Figura 10: Sequência de lógica programável.

Fonte: Sistemas Digitais - fundamentos e aplicações.

Primeiramente o projeto deve ser inserido no software por meio de entrada de texto (VHDL, verilog, HDL, ABEL) ou desenhos esquemáticos. Após ser inserido ele passa por um compilador, nesse processo o projeto é compilado para um formato na qual possa ser interpretado pelo dispositivo. No decorrer do processo a lógica é testada para verificar possíveis erros; traduzida para o dispositivo específico no processo de síntese; mapeada para o dispositivo físico durante a implementação; simulada e por fim é realizado o download para o dispositivo.

4.Metodologia

Durante a fase inicial do projeto algumas questões foram discutidas, algumas delas foram: o que é um binário natural? Como utilizar a tabela verdade? Como funciona o Quartus (aplicativo de projeto de PLDs)? Como funciona a comunicação com o dispositivo lógico programável? Para responder essas e outras questões foram levantadas algumas ideias iniciais, como por exemplo: utilizar os DIPs e botões presentes na placa FPGA como as entrada de dados do sistema, converte os dados inseridos na entrada para binário natural e criar tabelas-verdade para melhor visualização do problema. A intenção principal foi de criar um projeto que atenda os requisitos do problema e que fosse o mais funcional possível.

Na primeira etapa do projeto foi pensado como se daria a comunicação entre os componentes do sistema. Isso foi feito por intermédio de um diagrama de blocos, que são representações gráficas de um processo. Figura 11 demonstra diagrama de blocos criado. Após a modelagem do sistema foi percebida a necessidade da utilização de circuitos combinacionais que pudessem, a partir das entradas do sistema (usuários), gerar as saídas desejadas pelo problema.

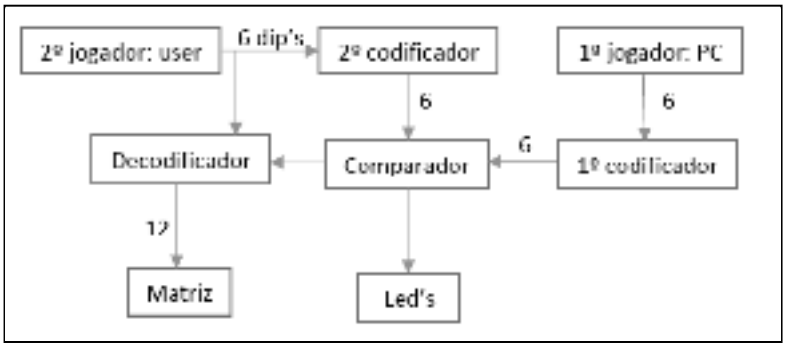

Figura 11: Diagrama de blocos inicial
Fonte: Próprio autor.

Como se trata da implementação de um jogo de batalha naval digital, algumas funcionalidades foram exigidas para manter a qualidade do produto resultante, sendo enumeradas da seguinte forma:

1º - O Jogador 1 (alvo) deverá fornecer ao protótipo a coordenada de um "pixel alvo" na matriz de LEDs, entradas internas codificadas de "0 a 34" em binário natural, correspondentes à posição do pixel;
2º - O Jogador 2 (atirador) deverá fornecer, através de entradas externas "***on/off***" presentes no kit de desenvolvimento ACEX 1K;
3º - As entradas externas deverão ser descritas em código binário natural, correspondentes à linha e a coluna, na tentativa de acertar o alvo;
4º - Se a coordenada do alvo for atingida, tanto a posição da matriz de LEDs como um LED VERDE externo deverão ficar acesos, indicando o sucesso na destruição do ALVO;
5º - Um LED VERMELHO externo à matriz de LEDs deverá indicar ao Jogador 2 que o alvo não foi atingido, podendo proceder com novas tentativas;

As exigências do projeto foram devidamente seguidas, com o objetivo de obter um produto dentro dos padrões propostos. Diversos problemas foram encontrados durante a elaboração do projeto, alguns deles foram enumerados como os problemas mais substanciais encontrados durante o desenvolvimento do sistema, foram listadas também as suas respectivas soluções:

1º Problema - Como criar um circuito que possibilitasse acender um LED específico na matriz de LEDs;
2º Problema - Como converter dois números binários representando linha e coluna em um único número binário natural;
3º Problema - Como comparar as entradas dos jogadores 1(interno) e 2(externo);

Para solucionar esses e outros problemas foram utilizados os conceitos ligados à lógica combinacional e simplificação de circuitos lógicos. As soluções de cada problema serão detalhadas a seguir:

Resolução do problema 1º - O problema possui como requisito que o jogador 2 (externo), deva oferecer as coordenadas do pixel alvo na matriz de LEDs, o problema também informa que essas coordenadas devem estar codificadas em binário natural. Para solucionar o problema proposto foi pensada a utilização de um decodificador para a matriz de LEDs.
A ideia se baseia em o usuário selecionar a posição desejada através dos DIPs presentes na placa FPGA. O número de combinações necessárias para a seleção das colunas e linhas, presentes na matriz de LEDs são de 2^n, ou seja, são necessários no mínimo três bits para representar as colunas e três bits para representar as linhas, conforme mostrado na figura 12.

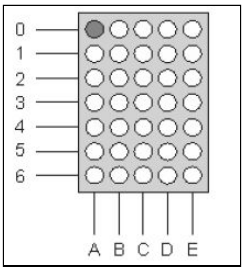

Figura 12: Matriz de LEDs.

Fonte: Problema 1 – MI Circuitos Digitais

As colunas são numeradas de 0 a 4 e as linhas de 0 a 6, caso o usuário quiser selecionar o LED na posição 9 por exemplo, ele teria que inserir os seguintes níveis lógicos nos DIPs 100-001, ou seja, coluna 4 e linha 1 em decimal, lembrando sempre que iniciasse a contagem a partir do zero. O próximo passo foi criar funções para decodificar esses dados em binário para a matriz de LEDs, essas funções estão representadas na Tabela 3.

Tabela 3: Tabela de decodificação de linhas na matriz de LEDs.

DECODIFICADOR DE LINHAS (SAÍDA DESEJADA = 0)			
DECIMAL	BINÁRIO	FUNÇÃO DE DECODIFICAÇÃO	SAÍDA
00	000	(A'.B'.C')'	0111111
01	001	(A'.B'.C)'	1011111
02	010	(A'.B.C')'	1101111
03	011	(A'.B.C)'	1110111
04	100	(A.B'.C')'	1111011
05	101	(A.B'.C)'	1111101
06	110	(A.B.C')'	1111110

Fonte: Próprio autor.

Tabela 4: Tabela de decodificação de colunas na matriz de LEDs.

DECODIFICADOR DE COLUNAS (SAÍDA DESEJADA = 1)			
DECIMAL	BINÁRIO	FUNÇÃO DE DECODIFICAÇÃO	SAÍDA
00	000	A'.B'.C'	10000
01	001	A'.B'.C	01000
02	010	A'.B.C'	00100
03	011	A'.B.C	00010
04	100	A.B'.C'	00001

Fonte: Próprio autor.

A partir dos dados presentes nas tabelas foram feitas as representações gráficas (esquemático) no projeto. As Figuras 13 e 14 demonstram respectivamente decodificador de colunas e linhas. Foi utilizada uma entrada adicional (E) para ativação do circuito, caso esta esteja desativada (0) o circuito não funciona e o LED não é aceso.

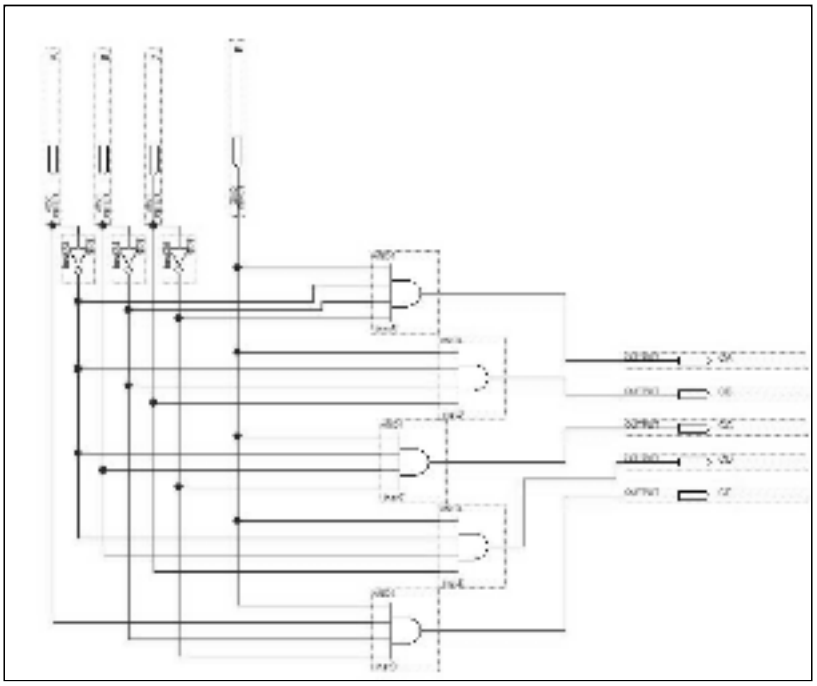

Figura 13 – Circuito do decodificador de colunas.
Fonte: Próprio autor.

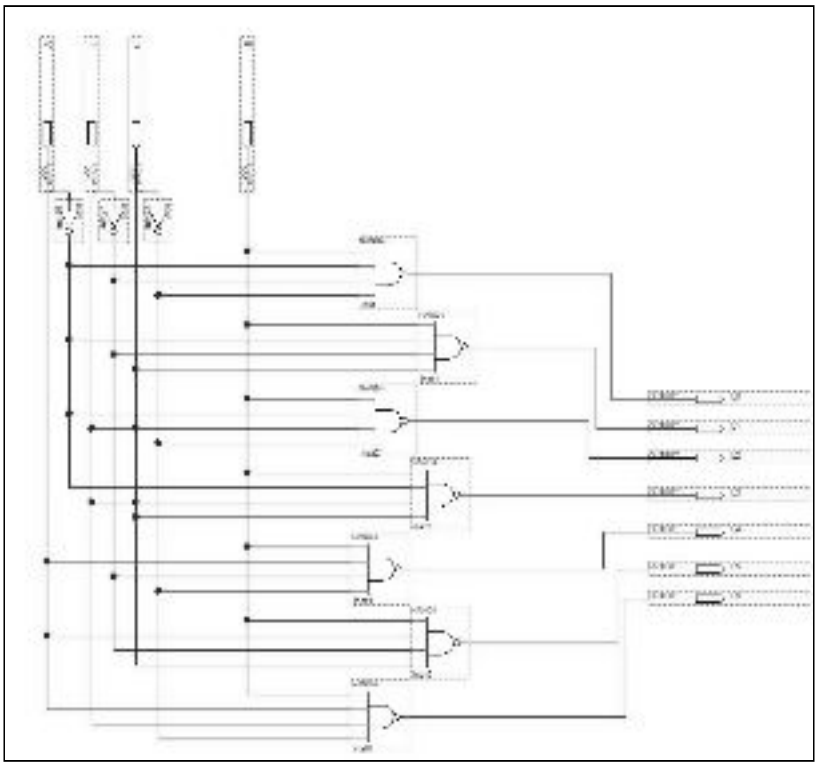

Figura 14 – Circuito do decodificador de linhas.
Fonte: Próprio autor.

Resolução do problema 2º - Um dos problemas encontrados foi o de converter as entradas binárias representando linhas e colunas em apenas um número binário natural, a fim de se fazer comparações entre os valores de entradas do jogador 1(interno) e do jogador 2 (externo). Esse problema foi solucionado de maneira simples. A ideia consiste em multiplicar o valor binário que representa a linha por cinco, em seguida somá-lo com o valor binário que representa a coluna. Por exemplo, caso o usuário informe a posição 100(4) para coluna e 110(6) para linha, de acordo com a figura 15 que representa as posições na matriz de LED, a posição desejada em binário natural é 100010(34), ou seja, a posição trinta e quatro em decimal. As operações algébricas realizadas pelo codificador são demonstradas na figura 16. É possível averiguar que ao se realizar esta operação o resultado desejado é justamente valor binário desejado.

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24
25	26	27	28	29
30	31	32	33	34

Figura 15: Posição dos LEDs na matriz.
Fonte: Próprio autor.

$$(110 * 101) \div 100 = 1000110 \Leftrightarrow (6 * 5) \div 4 = 34$$

Figura 16: Exemplo de expressão algébrica realizada internamente pelo codificador.
Fonte: Próprio autor.

Internamente são utilizados outros circuitos combinacionais para realizar as operações; um deles é o *multiplicador* de três bits, nele os números são multiplicados dois de cada vez e o resultado é uma soma de produtos parciais entre eles. Por tal motivo são utilizados circuitos *somadores* para adicionar os resultados dos produtos. A tabela verdade do circuito é demonstrada na Tabela 5. O resultado gerado pelo *multiplicador* é somado ao valor binário que representa as colunas, isso por meio de um circuito *somador* de seis bits.

Este somador é composto internamente de somadores-completos que realizam a soma de cada bit de forma paralela; os somadores completos possuem em sua estrutura duas entradas para as constantes e outra para um carry de entrada, possui também um carry de saída, com isso esses somadores operam em conjunto para a soma de algarismos acima de 1 dígito. Figura 17 demonstra a estrutura interna do somador de seis bits.

Tabela 5: Tabela verdade do Multiplicador.

TABELA VERDADE - MULTPLICADOR														
DECIMAL	ENTRADAS (A)			DECIMAL	ENTRADAS (B)			SAÍDAS (A)*(B)						DECIMAL
	A1	A2	A3		B1	B2	B3	S1	S2	S3	S4	S5	S6	
0	0	0	0	5	1	0	1	0	0	0	0	0	0	0
1	0	0	1	5	1	0	1	0	0	0	1	0	1	5
2	0	1	0	5	1	0	1	0	0	1	0	1	0	10
3	0	1	1	5	1	0	1	0	0	1	1	1	1	15
4	1	0	0	5	1	0	1	0	1	0	1	0	0	20
5	1	0	1	5	1	0	1	0	1	1	0	0	1	25
6	1	1	0	5	1	0	1	0	1	1	1	1	0	30
7	1	1	1	5	1	0	1	1	0	0	0	1	1	35

Fonte: Próprio autor.

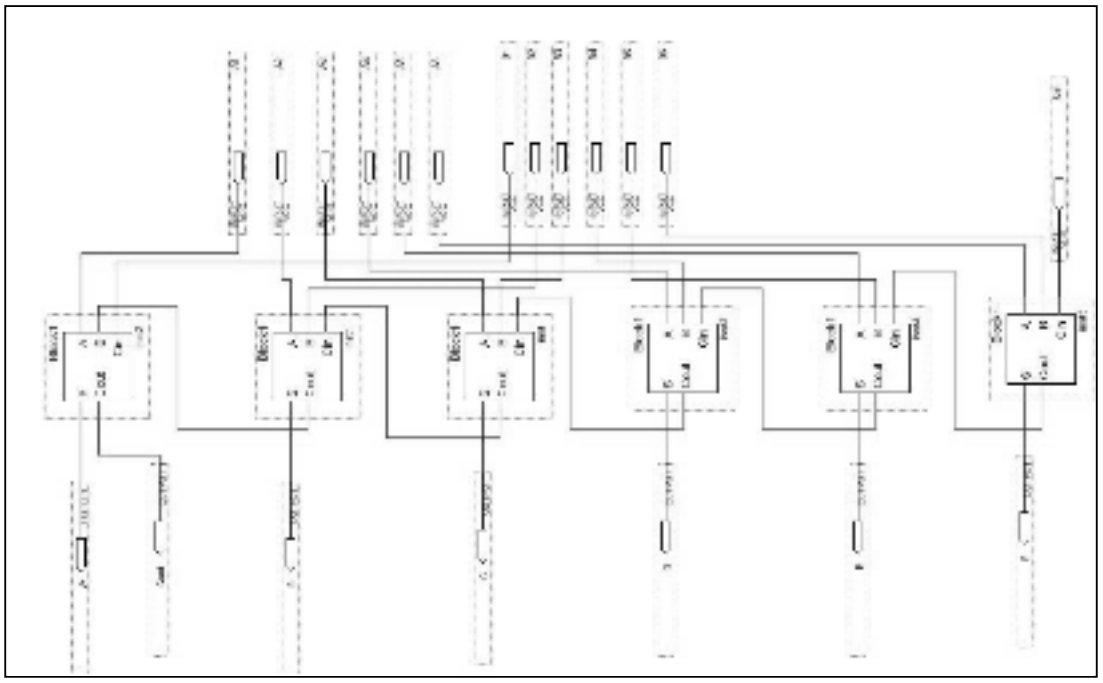

Figura 17: Estrutura interna do *Somador de 6 bits.*
Fonte: Próprio autor.

Resolução do problema 3º- Após os valores binários serem codificados para um único valor binário natural eles devem ser comparados, a fim de confirmar se são iguais. Sendo confirmada essa informação, deve ela ser repassada para que seja exibida para o usuário por meio de LEDs na placa FPGA. Para a solução desse problema foi utilizado um circuito *comparador.* Ele compara bit a bit de um número binário - isso é realizado por meio de portas XOR - a porta XOR possui saída alta (1) para valores iguais. Este *comparador* em especial não apenas compara se dois números são iguais, mais também determina qual deles é maior/menor, é denominado de *comparador de magnitude*. Nesse comparador são utilizadas portas XNOR que possui saída alta(1) quando dois valores são diferentes, são utilizadas também portas ANDs e portas NOTs para validar a grandeza dos valores. Ver figura 18 que demonstra circuito esquemático de um comparador de sinal magnitude de 1 bit.

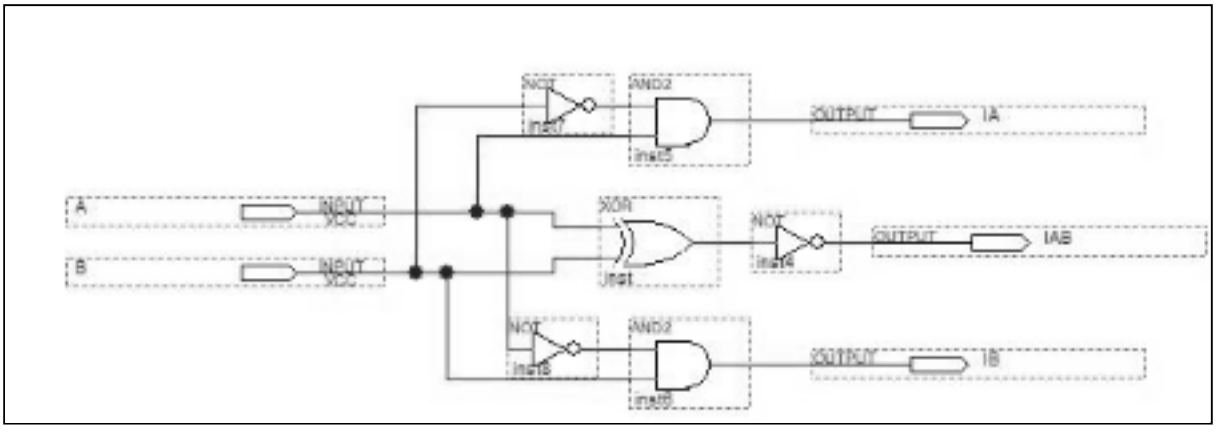

Figura 18: Circuito do comparador magnitude de 1 bit.
Fonte: Próprio autor.

Outros dilemas surgiram durante a projeção do sistema, um deles foi à lógica para acender os LEDs de confirmação na matriz de LEDs, o problema impõe que ao ser

confirmado que os dois usuários (interno/externo) passaram dois valores iguais dentro do conjunto de posições 0 a 34, um LED verde deve ser aceso, demonstrando que o jogador atingiu a posição correta. Caso o jogador tenha errado a posição deve ser aceso um LED na cor vermelha para informar ao jogador que errou a posição. Para solucionar esse dilema foi elaborada uma lógica, na qual o jogador 2 (externo) ao colocar a posição desejada nos DIPs, deve pressionar um botão de confirmação para ser demonstrado o resultado.

O processo funciona da seguinte forma: ao ser pressionado o botão um nível lógico alto (1) é passado para o sistema, ativando os decodificadores e demonstrando a posição inserida na matriz de LEDs. Os valores inseridos nas entradas passam pelos codificadores e em seguida são comparados, por seguinte o valor gerado pelo comparador é passado para uma porta AND que valida qual LED acender a depender do resultado do comparador. A tabela verdade deste circuito é demonstrado na Tabela 6.

Tabela 6: tabela verdade da lógica dos LEDs.

TABELA DA LÓGICA DOS LEDs			
ENTRADAS		SAÍDAS	
START	COMPARADOR	LED_VERDE	LED_VERMELHO
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	0

Fonte: Próprio autor.

Em projetos de engenharia é comum a tratativa de algumas situações na qual possam causar algum tipo de erro no sistema. Uma tratativa de erro necessária no problema foi quando os dois usuários (interno/externo) passam valores iguais e inválidos para o sistema, ou seja, fora do conjunto de 0 a 34. Para resolver esse problema foi utilizado um comparador extra, que checa se o número inserido pelo jogador 1(interno) foi maior que 34. A saída deste comparador é interligada ao outro comparador que valida se as posições inseridas pelos dois jogadores foram iguais. Como resultado quando o jogador 1 (interno) colocar um valor maior que trinta e quatro nas entradas, será então acesso o LED vermelho, informando que foi inserido um valor inválido. De mesmo modo, não será acessa nenhum LED na matriz de LEDs já que o valor não se encontra no conjunto definido.

Todo o projeto foi elaborado através da ferramenta de desenvolvimento Quartus II 9.0, foi também utilizada a Placa FPGA Starter Kit Altera - PI1K100A em conjunto com o KIT de desenvolvimento LabHard 1K. Foi utilizado também uma Matriz de LEDs 2x5x7 presentes no KIT de desenvolvimento.

5.Resultados e discussões

Nas primeiras sessões tutoriais entendeu-se que seria usado um *dip-switch* de 6 entradas e sua saída final seria a matriz de LEDs e LEDs auxiliares, assim como propõe o problema. O jogo se inicia com a escolha de posição pelo jogador 1 ainda por software, logo após isso, utilizando os 6 *dip-switch,* o jogador 2 pode escolher a posição que deseja "disparar" e pressionar o botão de *start*. Nesse momento tanto a escolha do jogador 1 quanto a escolha do jogador 2 passam por um codificador cada um, resultando em um binário natural cada um. Os dois binários são então comparados bit a bit, do mais significativo para o menos, e caso a comparação prove que todos os bit são iguais entre si, a saída do comparador será um nível lógico alto que em conjunto com o botão de *start* pressionado acenderá o "tiro" do jogador 2 na matriz de LEDs e um led verde auxiliar, mostrando que o mesmo acertou.

Caso a comparação prove que pelo menos um bit não é igual entre si, a saída do comparador será um nível lógico baixo e novamente junto com o botão *start* pressionado acenderá o "tiro" do jogador 2 na matriz de LEDs e um led vermelho auxiliar, mostrando que o mesmo errou. Em uma das exceções que poderia ocorrer, o jogador 1 poderia escolher uma posição inválida para a matriz de LEDs, isto é, um número decimal maior que 34. Neste caso é verificado antes mesmo da comparação bit a bit entre os jogadores, e no caso de escolha inválida por parte do jogador 1, a comparação entre os jogadores sequer é feita, retornando um baixo nível como saída do comparador bit a bit dos dois jogadores, fazendo assim, o LED auxiliar vermelho acender.

5.1.Desenvolvimento do codificador

O codificador consiste em 6 *inputs,* sendo 3 delas um número binário referente às colunas e os outros 3 referentes às linhas. Utilizando-se um circuito multiplicador e logo após um circuito somador de 6 bits, conforme explicado na resolução do problema 2º, gerando 6 *outputs* que formam um único número binário. Ver figura 19 a seguir demonstrando circuito interno do codificador.

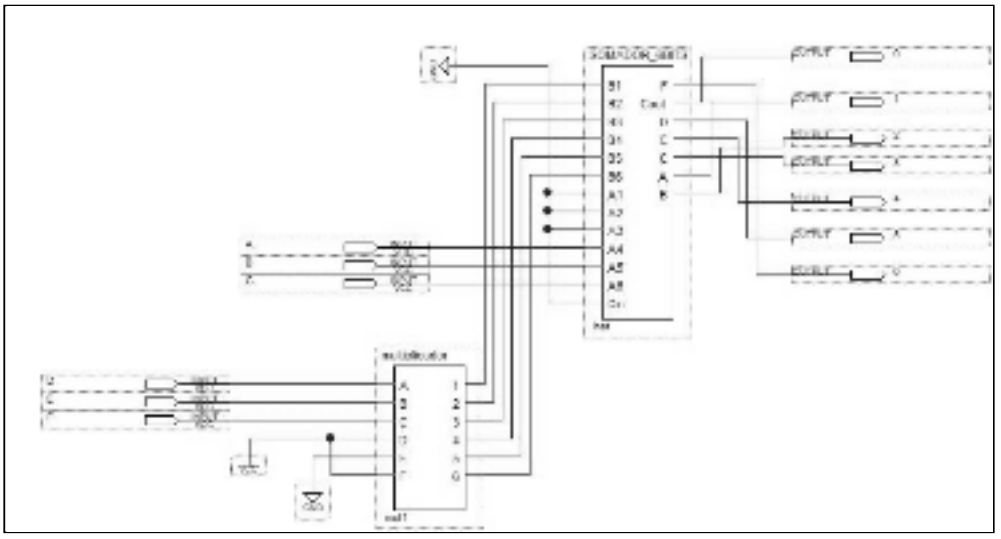

Figura 19: Circuito do codificador.
Fonte: Próprio autor.

5.2.Desenvolvimento do decodificador de colunas

O decodificador de colunas tem 4 *inputs,* 3 deles são entradas dos *dip-switch* que referem as colunas e 1 é a saída do comparador em uma porta AND com um botão. O decodificador de colunas tem 5 *outputs* na matriz de LED's que determinam qual coluna deve ser escolhida, determinando assim uma das duas coordenadas para o "tiro", conforme mostrado na resolução do problema 1º.

5.3.Desenvolvimento do decodificador de linhas

O decodificador de linhas tem 4 *inputs,* 3 deles são entradas dos *dip-switch* que referem as linhas e 1 é a saída do comparador em uma porta AND com um botão. O decodificador de linhas tem 7 *outputs* na matriz de LED's que determinam qual linha deve ser escolhida, selecionando assim a outra coordenada para o "tiro", conforme mostrado na resolução do problema 1º.

5.4.Desenvolvimento do comparador

O comparador bit a bit entre os dois jogadores tem no total 15 *inputs,* sendo 6 delas os bits de saída do codificador referente ao jogador 1 e outros 6 bits referente a saída do codificador do jogador 2. Os últimos 3 são compostos por uma entrada de energia que faz o comparador funcionar e os outros dois para a utilização do comparador em cascata, no qual não foi utilizado e por isso recebeu como entrada o "*ground*". O comparador na sua execução compara internamente bit a bit, conforme foi explicado na resolução do problema 3º e tem como resultado de comparação apenas um *output*.

5.5.Circuito final

A figura 20 mostra o circuito final implementado para resolução do problema.

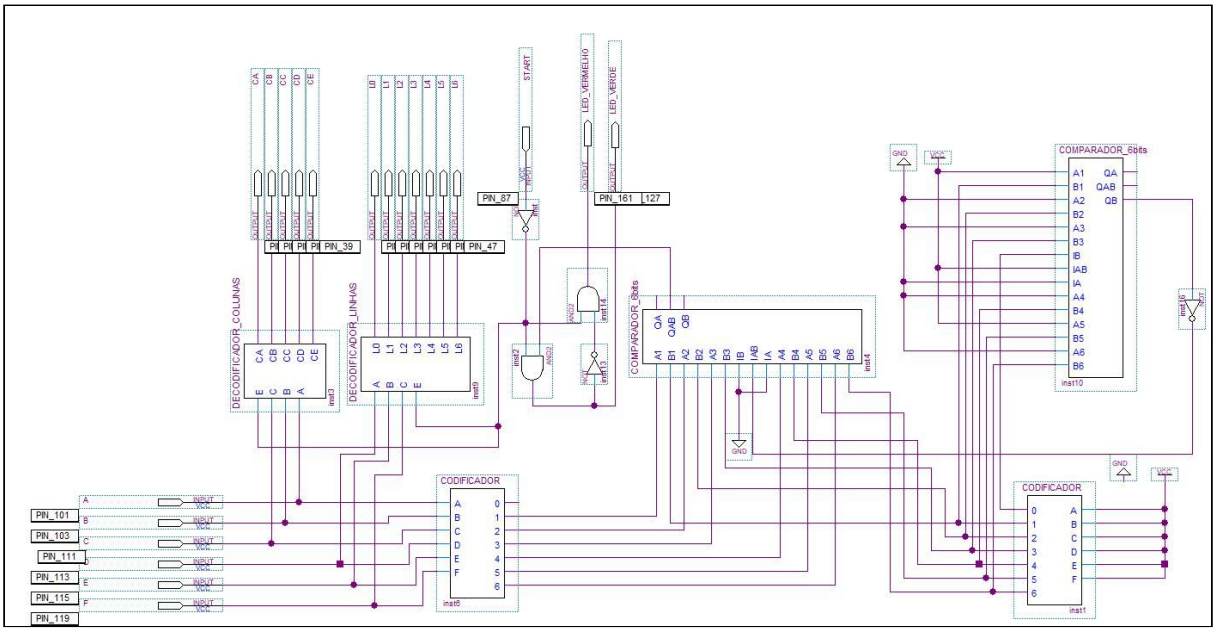

Figura 20: circuito completo
Fonte: Próprio autor.

6.Conclusão

Ao finalizar este relatório, conclui-se que o foi possível compreender e entender como se utiliza o software ALTERA Quartus II e as principais funções e aplicações da placa de hardware reconfigurável *FPGA*. Além disto, houve a utilização de conceitos e usos de álgebra de Boole, tabelas-verdade, portas logicas variadas (seus símbolos e usos), códigos binários, codificação de números decimal/binário e binário/decimal. Ainda foi possível notar o emprego de circuitos codificadores e decodificadores, multiplexadores, somadores, multiplicadores e comparadores.

Não somente, também, a utilização de conhecimentos lógicos e matemáticos já existentes. Houve a possibilidade de manusear técnicas de simplificação dos circuitos feitos, tais como o Mapa de Karnaugh. Este método poderia garantir um produto mais simples e "limpo" visualmente. Porém os autores chegaram à conclusão que seria necessário refazer grande parte do que já havia sido feito, o que implicaria em mais tempo aplicado nestes processos que poderiam atrasar o andamento de etapas futuras desta disciplina e de outras matérias paralelas.

Informa-se, também, que todos os requisitos apresentados foram cumpridos. É interessante ressaltar que, mesmo com dificuldades iniciais para a criação do circuito, o desenvolvimento e evolução do produto foram ocorrendo simultaneamente ao ganho de entendimento e compreensão dos conceitos. O acesso ao circuito digital desenvolvido é possível através do link a seguir:
https://drive.google.com/open?id=1HkAkkcUIXDutPnG-zOvTVCbcOaaH4I-7
Dentro da pasta do circuito, o arquivo principal é denominado 'continuar.bdf'.

7.Bibliografia consultada

FLOYD, Thomas L. "Sistemas Digitais: fundamentos e aplicações". 9 ed. Porto Alegre: Bookman, 2007.

IDOETA, Ivan V. "Elementos da eletrônica digital". 30 ed. São Paulo: Érica, 2000.

TOCCI, R. J. Sistemas Digitais: Princípios e Aplicações, Ed. LTC, 7ª. Edição, 2000.

WAKERLY, J. F. Digital Design: principles and practices. 3rd ed. Prentice Hall, 2001.

MANDADO, E. Sistemas Electrónicos Digitales, 9º ed. Marcombo, S.A. 2007.

IDOETA, I. J; CAPUANO, F.G. Elementos de Eletrônica, 41º ed. Érica, 2012.

GAJSKI, D. D. Principles of Digital Design, Prentice Hall, 1997.

PADILLA, A. J. G. Sistemas digitais. Lisboa: McGraw - Hill, 1993.