

Universidade Federal Da Fronteira Sul - UFFS  
*Campus* Chapecó  
Circuitos Digitais  
Adriano Padilha e Emilio Wuerges

# BRAILE

Gustavo Tavares Meneghini  
Isabela Maria Da Campo  
Lucas Bueno

Chapecó, 2017

## Resumo

*Braille* ou braile é um sistema de leitura com o tato para cegos inventado pelo francês Louis Braille no ano de 1827 em Paris. O Braille é um alfabeto convencional cujos caracteres se indicam por pontos em alto relevo. O deficiente visual distingue por meio do tato. A partir dos seis pontos relevantes, é possível fazer 64 combinações que podem representar letras simples e acentuadas, pontuações, números, sinais matemáticos e notas musicais. A figura abaixo mostra a condificação das letras neste alfabeto.

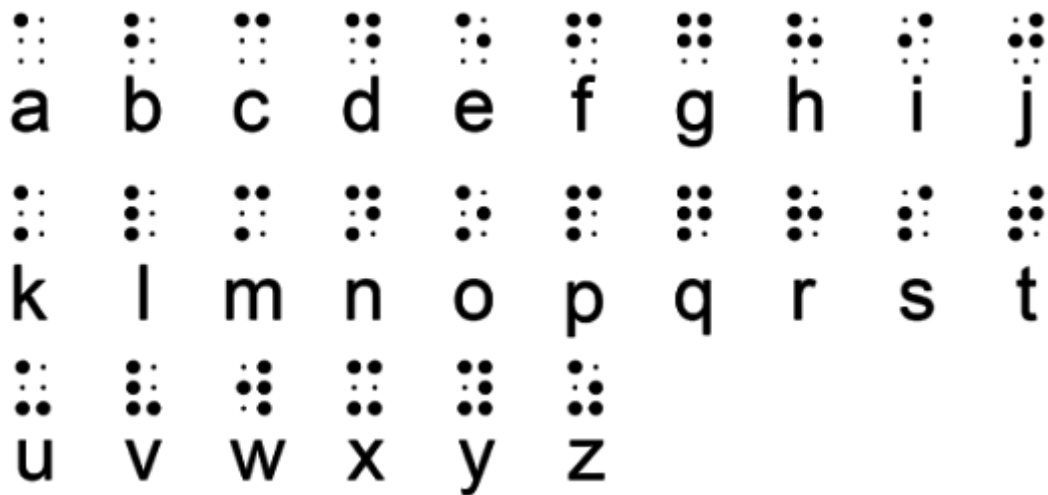


Figura 1: Alfabeto em Braille. Fonte: <https://brainly.com.br/tarefa/7387450>

# Sumário

Resumo.....	2
Objetivo.....	4
Introdução.....	5
Metodologia.....	7
Protoboard.....	11
LogiSim.....	13
Conclusão.....	16
Referencias.....	17

## Objetivo

O trabalho teve como objetivo o estudo e implementação de um circuito em *braille*, em que os estudantes tem que implementar um dos seis leds do circuito na *Protoboard* ou *Breadboard*. E na ferramenta logisim implementar junto com um *decoder* oito caracteres, em que o usuário entra com os caracteres em ASCII ou com o valor em hexadecimal e o circuito converte para um LED e este LED deverá ser aceso se necessário, formando a letra em braile.

## Introdução

O alfabeto em braile é executado por pelo menos uma pessoa, onde a pessoa realiza a configuração dos bits de entrada em código ASCII (tabela 1), para que o circuito converta essa entrada de dados e assim formar a letra em braile.

Os possíveis posicionamentos são de A a E, conforme a tabela abaixo. Os locais em que os valores estão com 0 entre L0 à L5 o LED não devem ser aceso, se o valor estiver 1 o LED deve ser aceso.

	A	B	C	D	E	L0	L1	L2	L3	L4	L5
	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	1	1	0	0	0	0	0
B	0	0	0	1	0	1	1	0	0	0	0
C	0	0	0	1	1	1	0	0	1	0	0
D	0	0	1	0	0	1	0	0	1	1	0
E	0	0	1	0	1	1	0	0	0	1	0
F	0	0	1	1	0	1	1	0	1	0	0
G	0	0	1	1	1	1	1	0	1	1	0
H	0	1	0	0	0	1	1	0	0	1	0
I	0	1	0	0	1	0	1	0	1	0	0
J	0	1	0	1	0	0	1	0	1	1	0
K	0	1	0	1	1	1	0	1	0	0	0
L	0	1	1	0	0	1	1	1	0	0	0
M	0	1	1	0	1	1	0	1	1	1	0
N	0	1	1	1	0	1	0	1	1	1	0
O	0	1	1	1	1	1	0	1	0	0	0
P	1	0	0	0	0	1	1	1	1	1	0
Q	1	0	0	0	1	1	1	1	1	1	0
R	1	0	0	1	0	1	1	1	0	0	0
S	1	0	0	1	1	0	1	1	1	1	0
T	1	0	1	0	0	0	1	1	1	0	0
U	1	0	1	0	1	1	0	1	0	0	1
V	1	0	1	1	0	1	1	1	0	1	1
W	1	0	1	1	1	0	1	0	1	1	1
X	1	1	0	0	0	1	0	1	1	1	1
Y	1	1	0	0	1	1	0	1	1	1	1
Z	1	1	0	1	0	1	0	1	0	1	1
	1	1	0	1	1	0	0	0	0	0	0
	1	1	1	0	0	0	0	0	0	0	0
	1	1	1	0	1	0	0	0	0	0	0
	1	1	1	1	0	0	0	0	0	0	0
	1	1	1	1	1	0	0	0	0	0	0

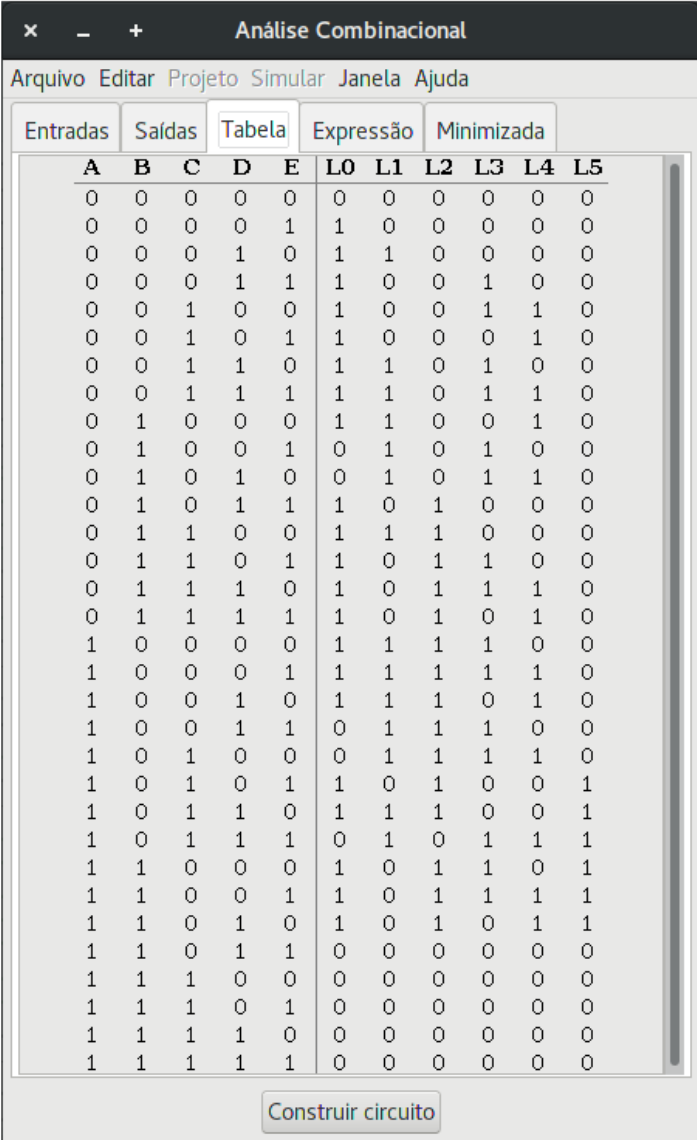
Tabela 1:

Tabela da verdade: alfabeto em braile

## Metodologia

Após todas as entradas possíveis serem definidas (conforme mostra tabela 1) na tabela da verdade, utilizamos o programa *Logisim*. Com este programa, é possível analisar e montar o circuito.

No programa, ao selecionarmos a opção “analisar circuito” damos os nomes das cinco entradas (A, B, C, D e E). Essas cinco entradas nos dão 32 opções de saídas ( $2^5=32$ ). Nessa parte também, damos os nomes das saídas (L0, L1, L2, L3, L4 e L5), para as saídas é necessário entrar com os dados manualmente para assim, o software gerar a equação e o circuito, conforme mostra figura abaixo.



A	B	C	D	E	L0	L1	L2	L3	L4	L5
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0
0	0	0	1	0	1	1	0	0	0	0
0	0	0	1	1	1	0	0	1	0	0
0	0	1	0	0	1	0	0	1	1	0
0	0	1	0	1	1	0	0	0	1	0
0	0	1	1	0	1	1	0	1	0	0
0	0	1	1	1	1	1	0	1	1	0
0	1	0	0	0	1	1	0	0	1	0
0	1	0	0	1	0	1	0	1	0	0
0	1	0	1	0	0	1	0	1	1	0
0	1	0	1	1	1	0	1	0	0	0
0	1	1	0	0	1	1	1	0	0	0
0	1	1	0	1	1	0	1	1	0	0
0	1	1	1	0	1	0	1	1	1	0
0	1	1	1	1	1	0	1	0	1	0
1	0	0	0	0	1	1	1	1	0	0
1	0	0	0	1	1	1	1	1	1	0
1	0	0	1	0	1	1	1	0	1	0
1	0	0	1	1	0	1	1	1	0	0
1	0	1	0	0	0	1	1	1	1	0
1	0	1	0	1	1	0	1	0	0	1
1	0	1	1	0	1	1	1	0	0	1
1	0	1	1	1	0	1	0	1	1	1
1	1	0	0	0	1	0	1	1	0	1
1	1	0	0	1	1	0	1	1	1	1
1	1	0	1	0	1	0	1	0	1	1
1	1	0	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0

Ao finalizar o preenchimento das saídas e clicar em “construir circuito” o programa monta automaticamente o circuito. Conforme imagem figura 3.

Figura 2: Tabela da Verdade

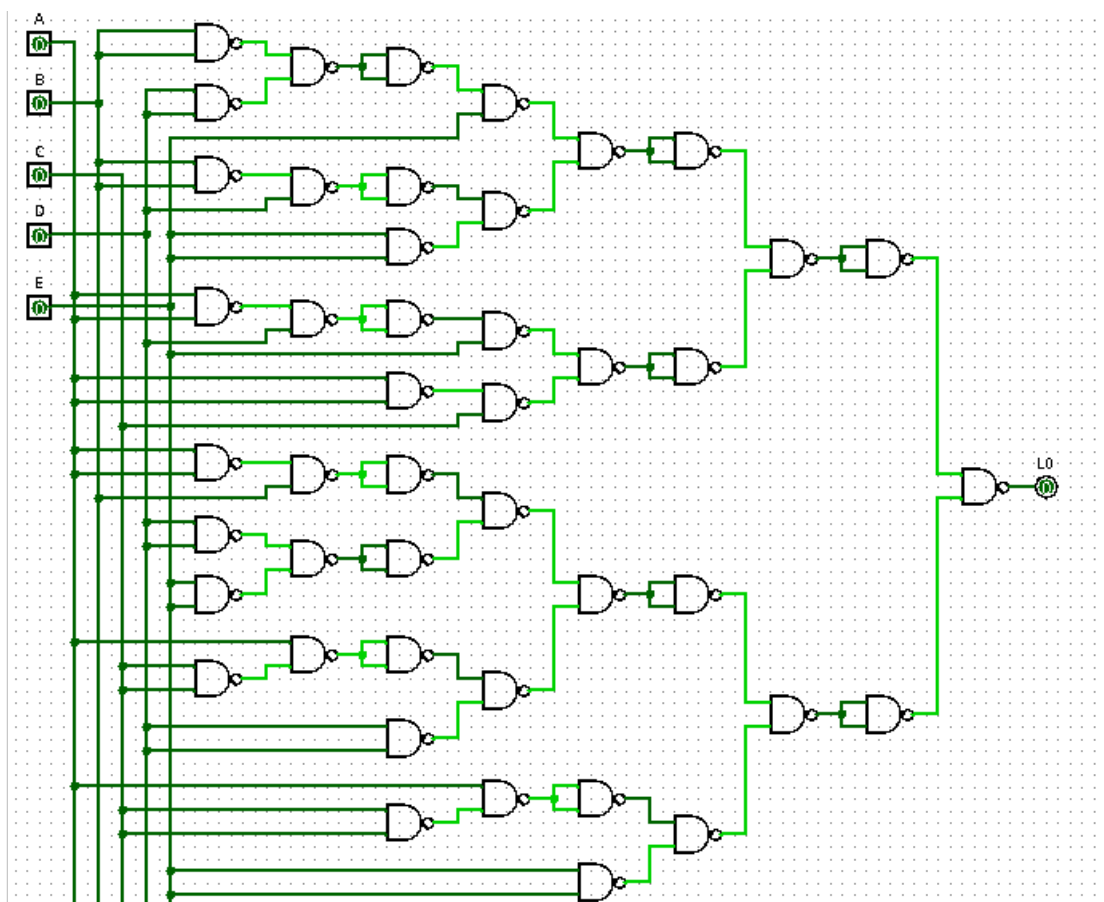


Figura 3: LED 0 - Circuito de Alfabeto em Braille

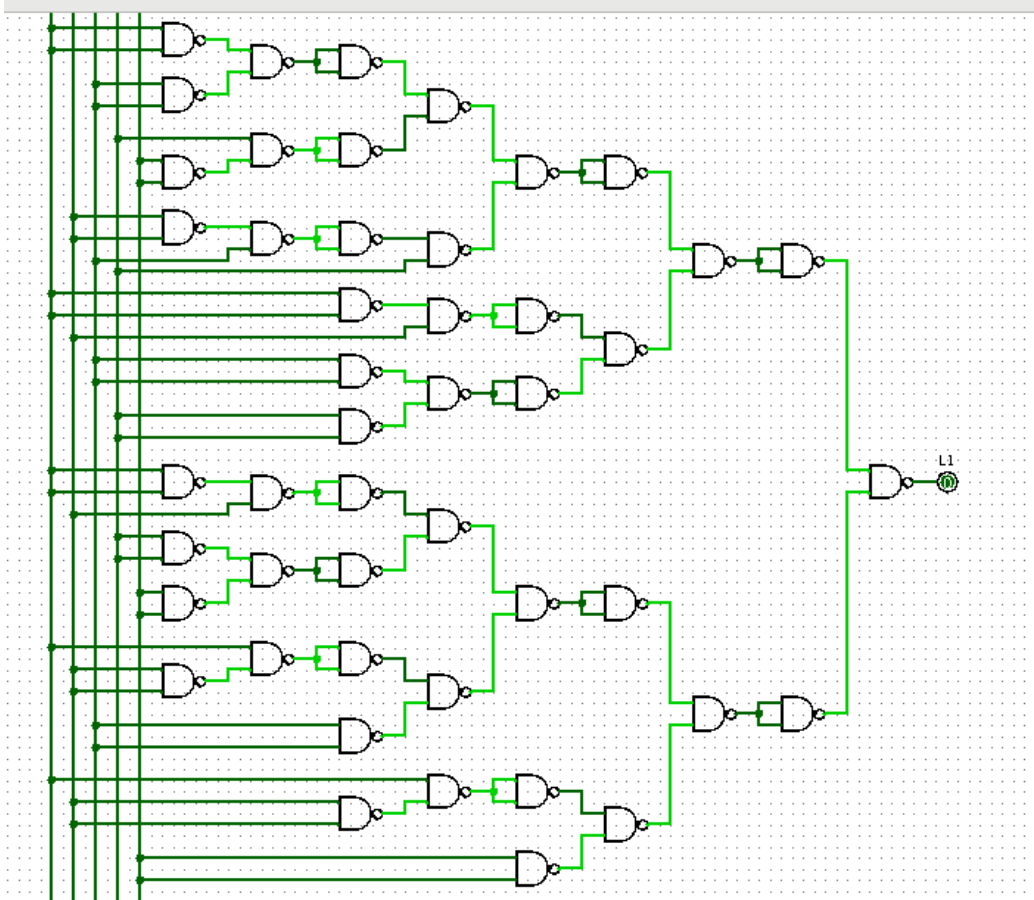


Figura 4: LED 1 - Circuito de Alfabeto em Braille



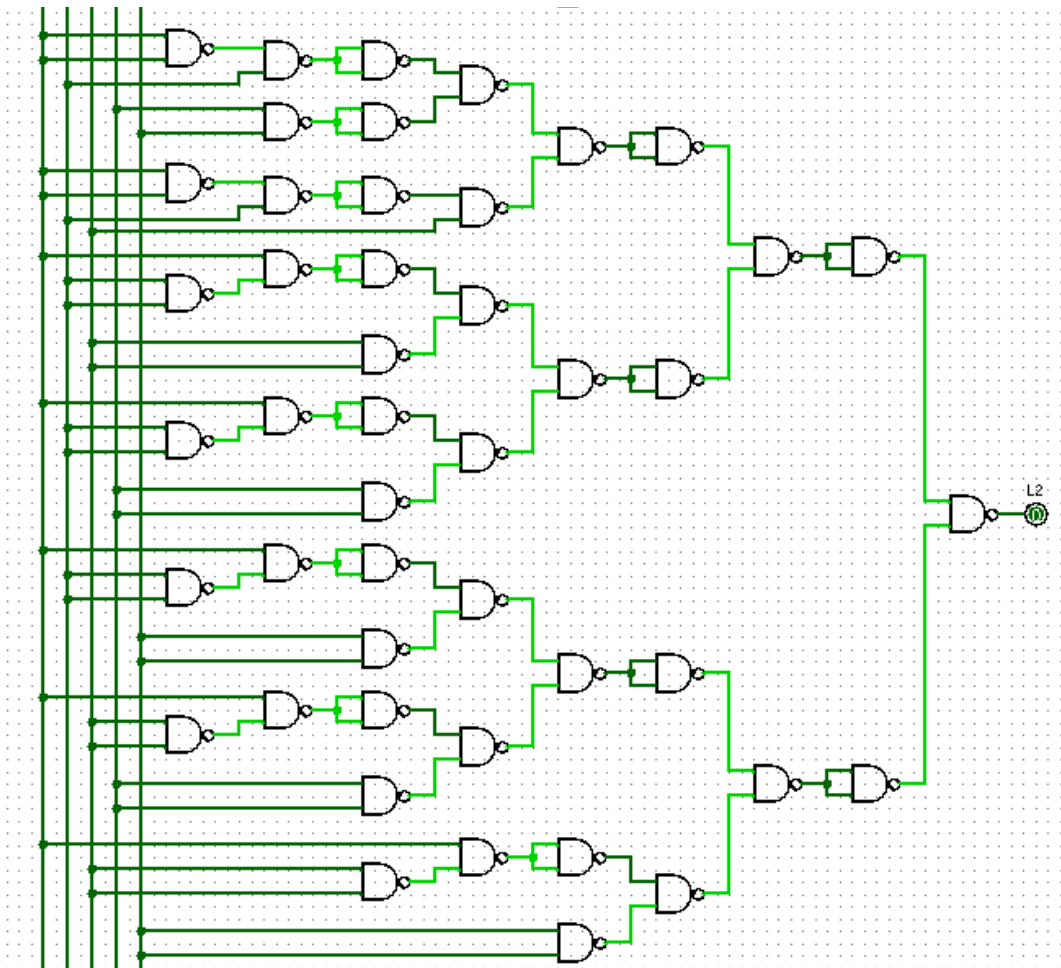


Figura 5: LED 2 - Circuito de Alfabeto em Braille

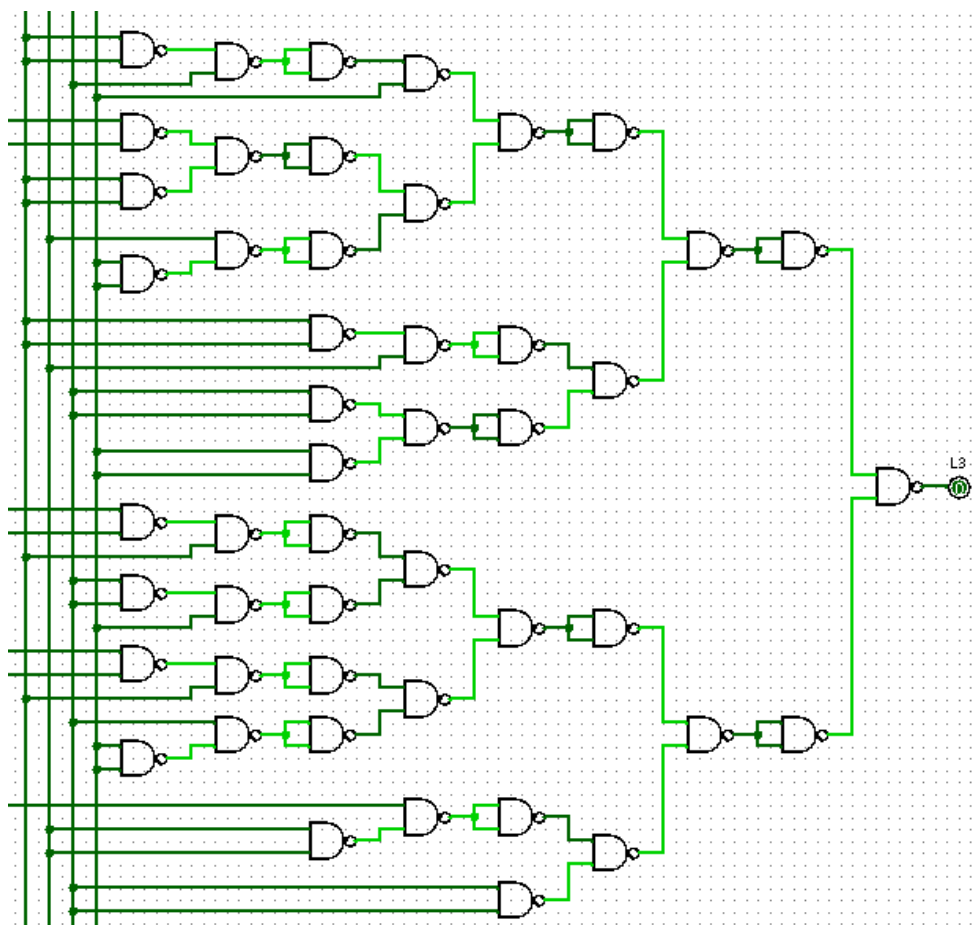


Figura 6: LED 3 - Circuito de Alfabeto em Braille

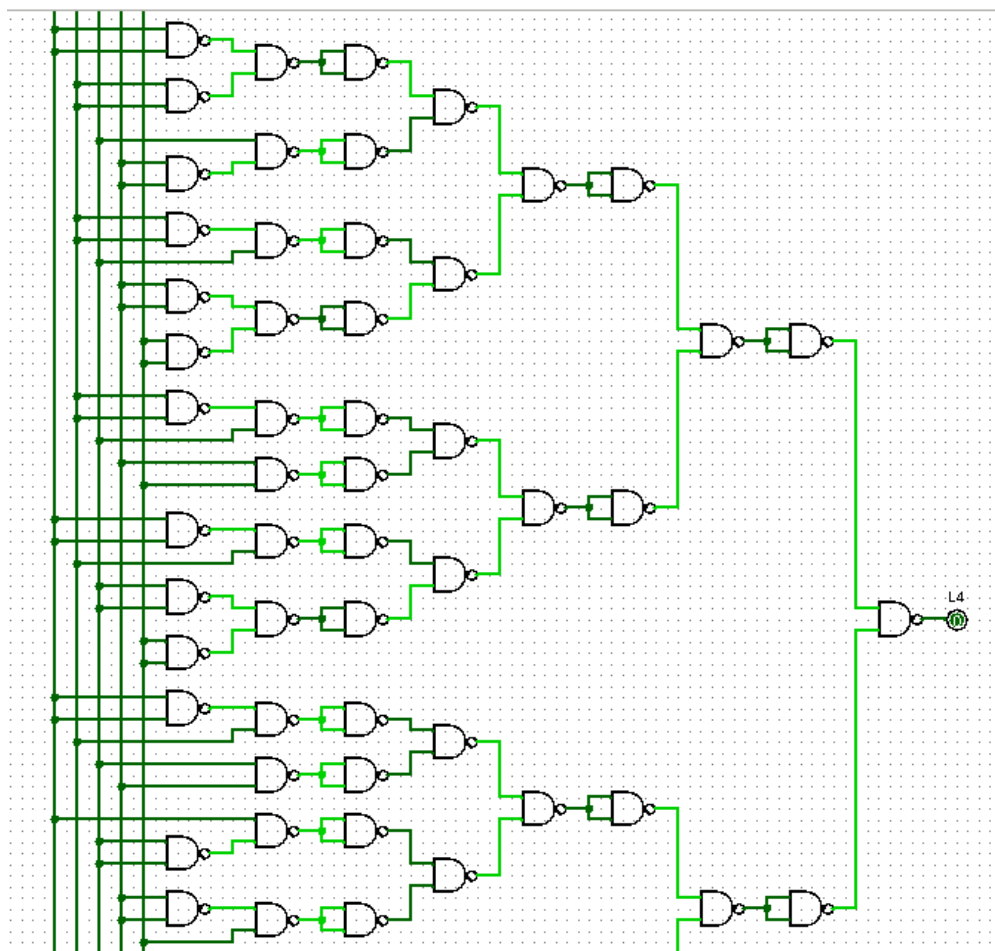


Figura 7: LED 4 - Circuito de Alfabeto em Braile

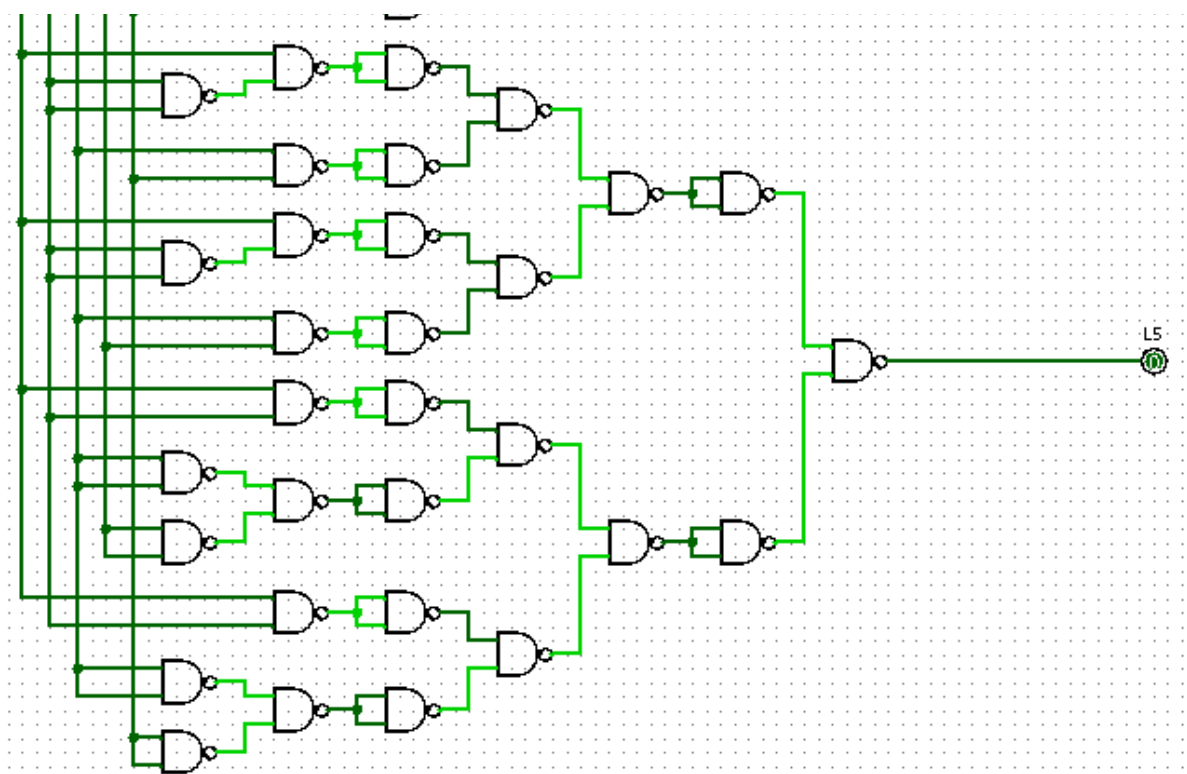


Figura 8: LED 5 - Circuito de Alfabeto em Braile

## Protoboard

Um dos desafios proposto pelos professores, era montar um dos LEDs do circuito na protoboard. O LED escolhido pelo grupo foi o LED 5 (imagem abaixo), pois o circuito dele é menor, e devido o tempo, se tornaria mais fácil de monta-lo na matriz de contato.

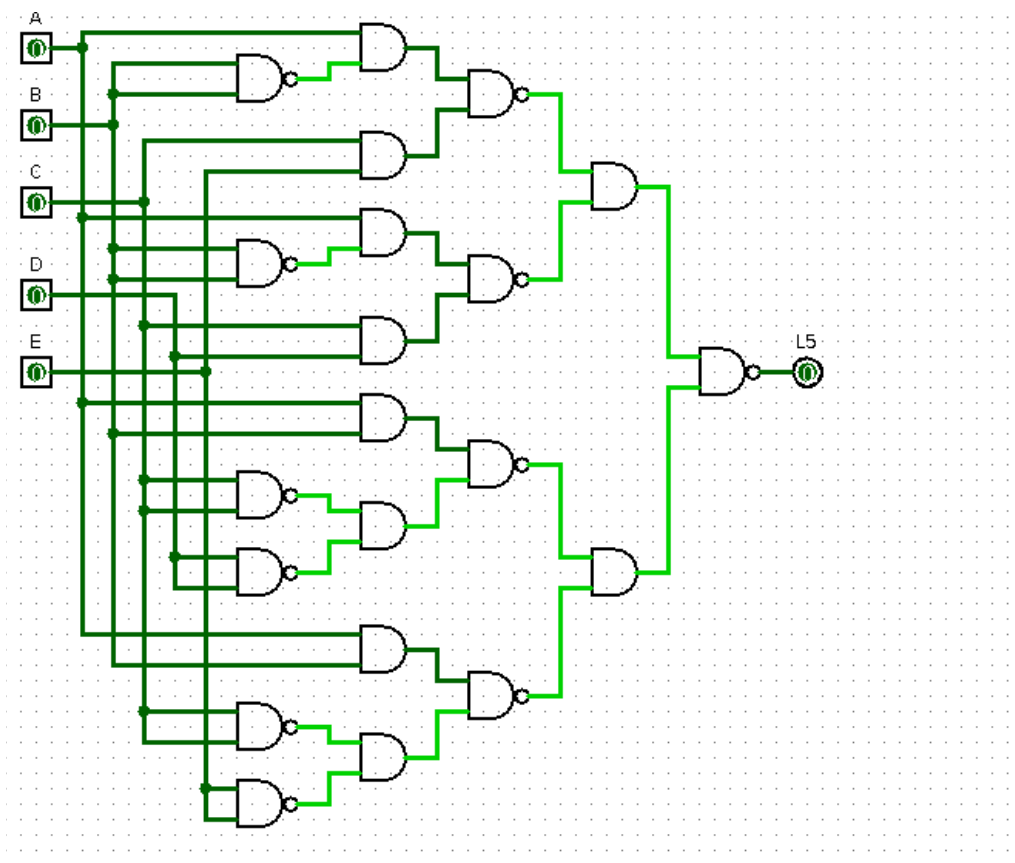


Figura 9: Circuito escolhido pelo grupo para montar na matriz de contato

Com a implementação do circuito completa, fizemos uma simulação do mesmo em um simulador de protoboard (tinkercad.com), para testarmos e verificarmos todos os componentes necessários para realizar a montagem na protoboard física. Como demonstrado nas figuras abaixo:

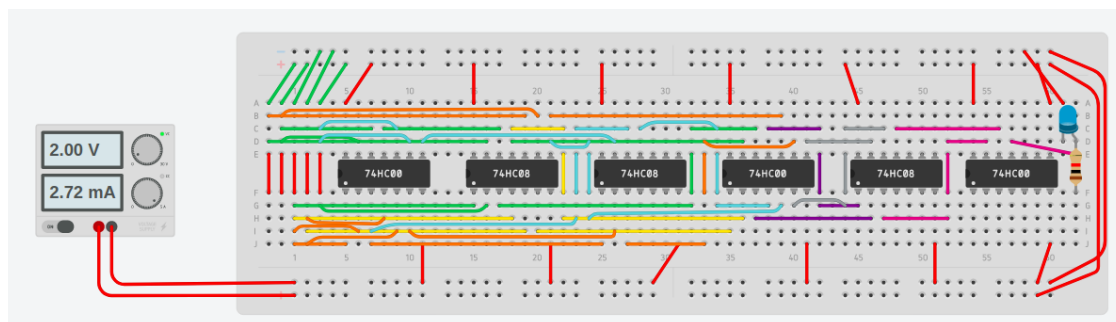
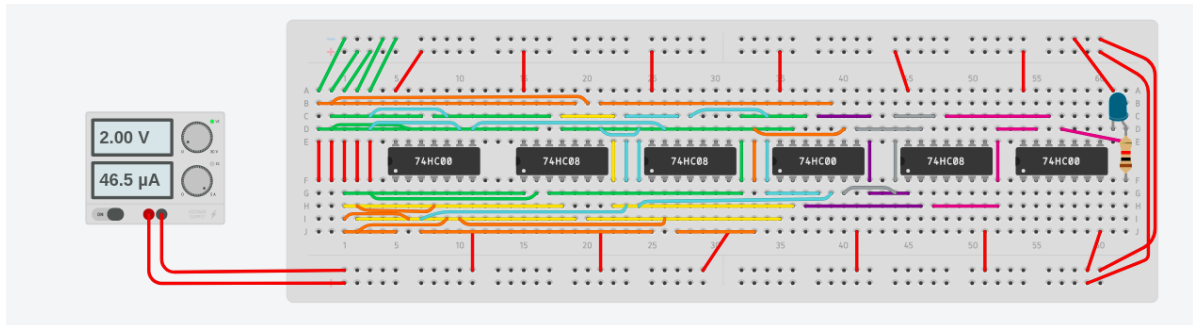


Figura 10: Simulação Protoboard: Resposta correta



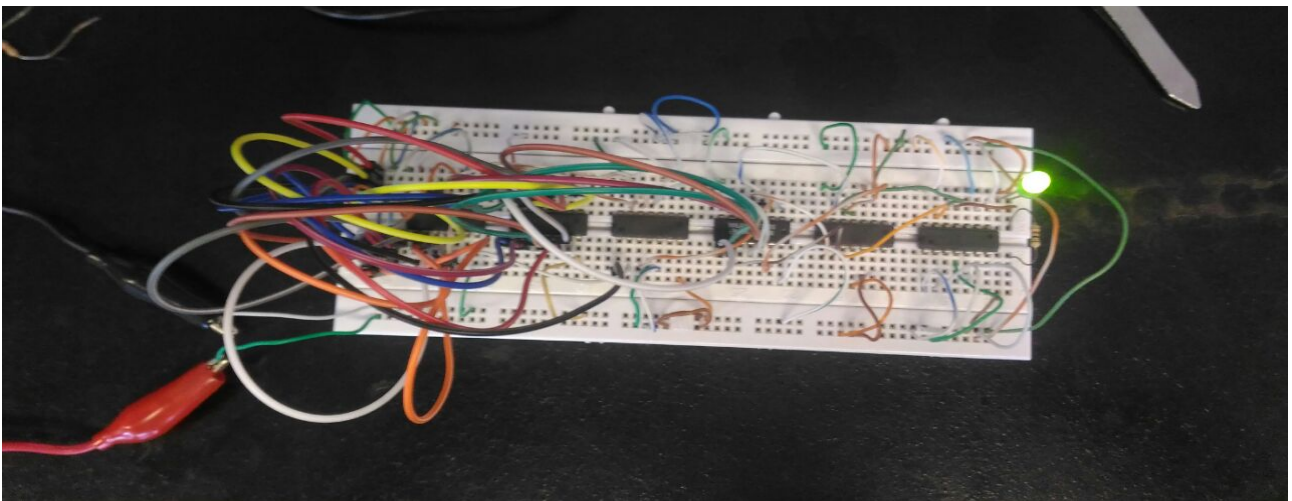
*Figura 11: Simulação Protoboard: Resposta incorreta*

Materiais utilizados para construir o circuito na matriz de contato:

Após o desenvolvimento teórico e a construção do circuito no simulador, o grupo se dirigiu ao laboratório de circuitos digitais, então deu início as atividades práticas. No laboratório foi possível obter os materiais necessários para a aplicação. Os materiais utilizados foram:

- Fios;
- Protoboard;
- LED;
- Resistor;
- Circuitos:
  - 3x 74HC00 (NAND);
  - 3X 74HC08 (AND);

Com os materiais em mãos, foi realizada a construção do circuito na protoboard, conforme mostra figuras abaixo:



*Figura 12: Circuito montado na Protoboard*

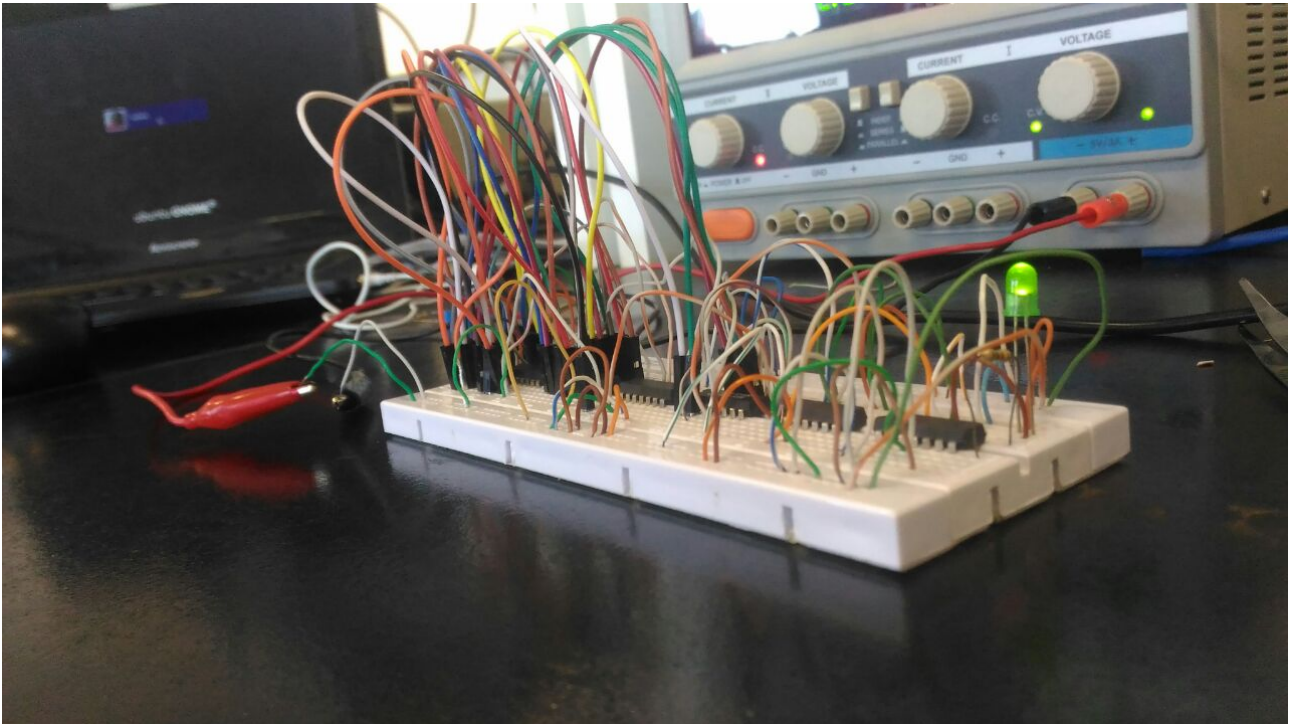


Figura 13: Circuito montado na Protoboard

## LogiSim

Outro desafio proposto pelos professores era construir o circuito digital capaz de ler oito caracteres (em ASCII) e acionar os pontos do código em braile (representados pelos LEDs), formando assim, os caracteres.

Para isso, é necessário termos o circuito principal, o registrador e um decodificador.

**Registrador:** Um registrador serve para guardar um único valor multibit, que será mostrado em hexadecimal dentro de seu retângulo, e emitido em sua saída Q saída.

Quando a entrada de *clock* (marcada por um triângulo na face sul), assim indicar, o valor armazenado no registrador será alterado para o valor na entrada D naquele instante. Exatamente quando a entrada de *clock* indicará a situação para que isso aconteça será configurado através do atributo Gatilho.

A entrada *Reset* levará o valor no registrador para 0 (em todos os bits) de forma assíncrona, ou seja, enquanto essa entrada for 1, o valor ficará fixo em 0, independente da entrada de *clock*.

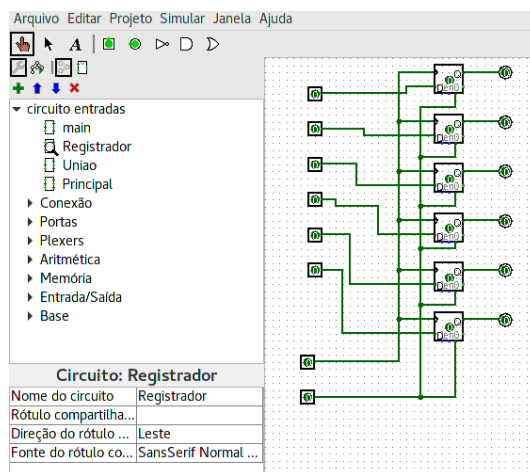


Figura 14: Registrador

Decodificador: Emitirá 1 em apenas em uma saída; a saída em 1 dependerá do valor corrente recebido através da entrada na face sul (circuito abaixo).

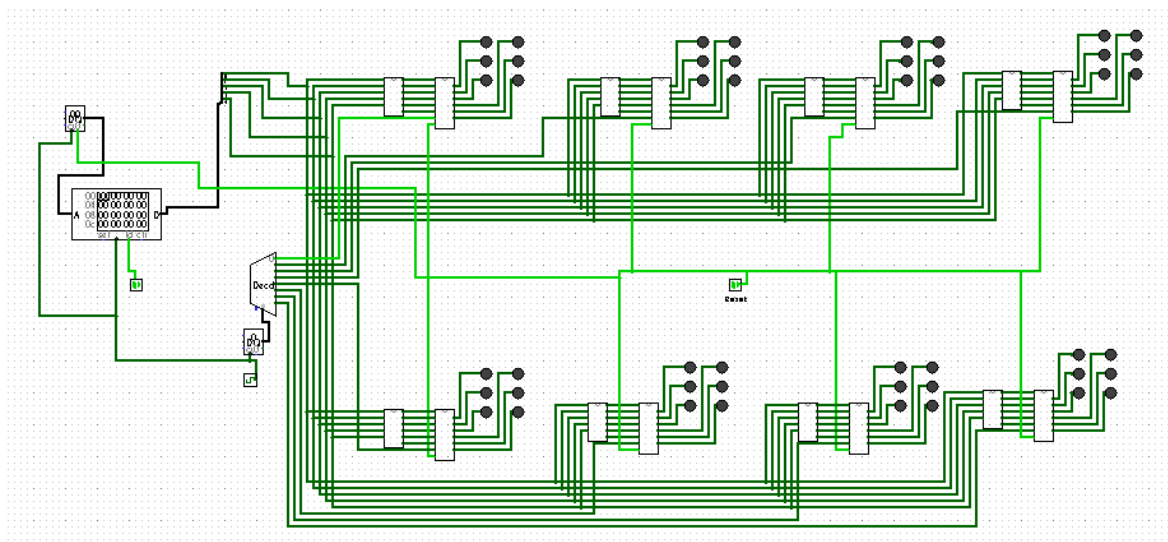


Figura 15: Circuito digital completo

No circuito acima, o usuário entra com os dados com os valores em hexadecimal, para assim, o circuito fazer a conversão para o código braile.

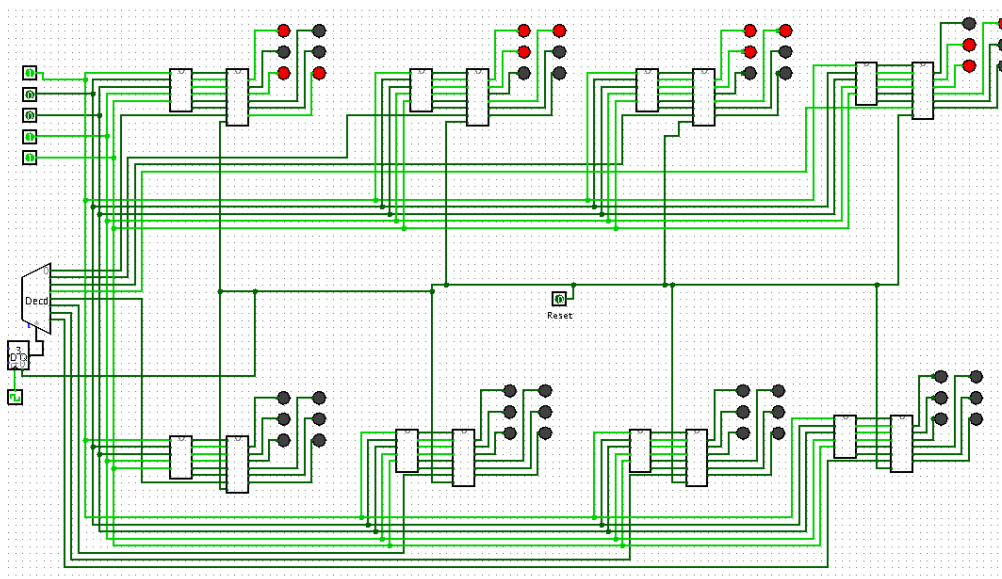


Figura 16: Circuito digital completo

Conforme mostra o circuito digital acima, o usuário entrou com os dados conforme valores da tabela da verdade e formou a palavra UFFS.



## **Conclusão**

O circuito montado na Protoboard e no LogiSim, mostrou-se capaz de resolver o problema proposto pelos professores, em que o usuário entra com os valores em código ASCII, e o circuito aciona os LEDs necessários para converter o código ASCII para o código braile.



## Referencias

Tinkercad. Disponível em: <<https://www.tinkercad.com/>> Acesso em 30 de novembro de 2017.

Logisim. Disponível em: <<http://www.cburch.com/logisim/pt/>> Acesso em 30 de novembro de 2017.

ASCII. Disponível em: <<http://ic.unicamp.br/~everton/aulas/hardware/tabelaASCII.pdf>> Acesso em 08 de dezembro de 2017.

Registrador. Disponível em:  
<<http://www.cburch.com/logisim/docs/2.7/pt/html/libs/mem/register.html>> Acesso em 10 de dezembro de 2017.

Decodificador. Disponível em:  
<<http://www.cburch.com/logisim/docs/2.7/pt/html/libs/plexers/decoder.html>> Acesso em 10 de dezembro de 2017.