

Trabalho de circuitos

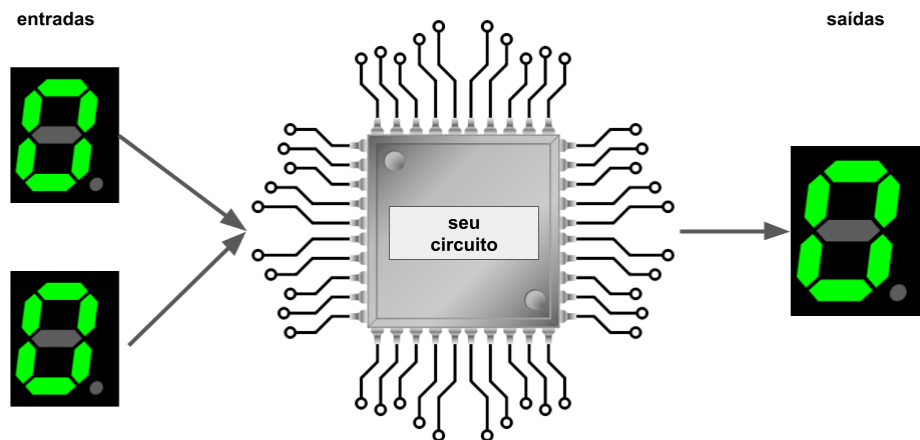
Tentar executar um projeto de soma de 2 algarismos em base decimal.
Aplicação de circuitos na linguagem C e na plataforma logic.ly

Equipe:

Érick de Brito Sousa Lima

Igor Torquato

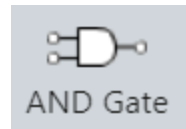
João Marcelo lobo



Será feito de tal forma: o usuário deve "desenhar" através de displays dois valores, que serão transformados em 4 bits e será mostrado um valor de 5 bits em decimal

Para iniciar, primeiramente vamos testar se é possível executar este projeto em C. Utilizando teclas do teclado para alternar os valores do input individualmente, podemos facilmente fazer uma ferramenta que faz a soma de dois números em binário. Mas como isso é feito em C? A linguagem tem ferramentas lógicas através do comando 'if' que pode fazer testagem com && (AND) e || (OR), diferente do C++ que possui o tipo de dado "bool", representando verdadeiro e falso, usaremos inteiros que trocam entre os valores 1 e 0, e usando constantes globais, vemos que: x_premissa = VERDADE;

```
int AND(int A, int B){  
    if(A == LIG && B == LIG) return LIG;  
    else return DES;  
}
```



```
int OR(int A, int B){  
    if(A == LIG || B == LIG) return LIG;  
    else return DES;  
}
```



```
int NOT(int A){  
    if(A == LIG) return DES;  
    else return LIG;  
}
```

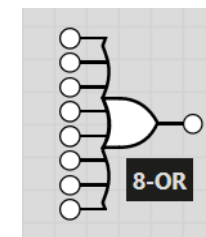


```
int OCTOR(int A, int B, int C, int D, int E, int F, int G, int H){  
    if(A == 1 || B == 1 || D == 1 || E == 1 || C == 1 || F == 1 || G == 1 || H == 1) return 1;  
    else return 0;  
}
```

```
+ int W = 0, E = 0, R = 0;  
  int S = 0, D = 0, F = 0;  
int Z = 0, X = 0, C = 0, V = 0;
```

```
V = SUM(R,F,RIU);  
RID = carry_out(R,F,RIU);  
C = SUM(E,D,RID);  
RIC = carry_out(E,D,RID);  
X = SUM(W,S,RIC);  
RIM = carry_out(W,S,RIC);  
Z = RIM;
```

```
int SUM(int A, int B, int C){  
    return XOR(XOR(A,B),C);  
}  
  
int carry_out(int A, int B, int C){  
    return OR(OR(AND(A,B),AND(B,C)),AND(A,C));  
}
```



Não há necessidade que eu explique sobre bases decimais, binário, etc.. Mas, vamos falar sobre como funciona em busca do 'Eureka!'.

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \quad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \quad \begin{array}{r} 1 \\ 1 \\ \hline 10 \end{array}$$

Podemos observar que existe um padrão para a soma de 2 bits. Como as entradas devem ter 4, representaremos o resultado com 5, então podemos montar uma tabela para descobrir qual o padrão:

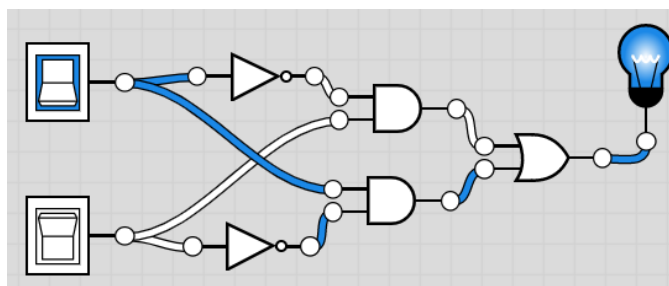
x	y	Unid.
0	0	0
0	1	1
1	0	1
1	1	0

SIM! Um padrão existe, podemos observar que quando as entradas são diferentes, a casa decimal em questão será verdade no resultado. Na lógica, já conhecemos a bicondicional, e sua negação é a disjunção exclusiva. Como descobrimos qual o circuito responsável pela D.E.? Conhecendo a porta AND e OR podemos usar a álgebra proposicional para descobrir.

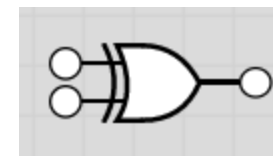
p	q	<u>p ↔ q</u>
V	V	F
V	F	V
F	V	V
F	F	F

$$\begin{aligned} & \overline{P \leftrightarrow Q} \\ & (P \rightarrow Q) \wedge (Q \rightarrow P) \\ & (\bar{P} \vee Q) \wedge (Q \vee \bar{P}) \\ & (P \wedge \bar{Q}) \vee (Q \wedge \bar{P}) \end{aligned}$$

Essa expressão comum motivou a criação, essa é a porta XOR, representada pelo símbolo \oplus



```
int XOR(int A, int B){
    return OR(AND(A,NOT(B)),AND(NOT(A),B));
}
```



Agora só precisamos resolver o problema da casa decimal, que vimos anteriormente e ignoramos, como podemos representar isso? Bom, devemos prestar atenção que na verdade, o valor que sobra da casa anterior faz parte da soma dos dígitos da casa pra onde ela foi enviada, já que na verdade $1+1$ é o mesmo que $01+01$ e a na casa das unidades: $1+1 = 0$ e vai 1 para as dezenas, e na próxima casa teremos $0+0+1 = 1$. Espere.. Então no caso todas as somas devem precisar de 3 números, os 2 a serem somados e alguma possível sobrecarga da casa anterior.. Vamos anotar isso. Agora voltando, como podemos entender o problema da sobrecarga (em inglês overflow, ou seja, transbordar)?

1
10
ste nndemo

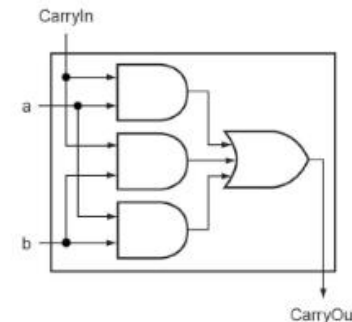
X	Y	CI	Of?
0	0	0	F
0	0	1	F
0	1	0	F
0	1	1	V
1	0	0	F
1	0	1	V
1	1	0	V
1	1	1	V

Podemos observar, que dos 3 valores que serão somados para nos dar o resultado daquela casa decimal, basta que PELO MENOS DOIS deles sejam 1. Podemos representar isso como um circuito? Sim, valeu Karnaugh!

AB \ C	0	1
00	0	1
01	1	0
11	1	0
10	0	1

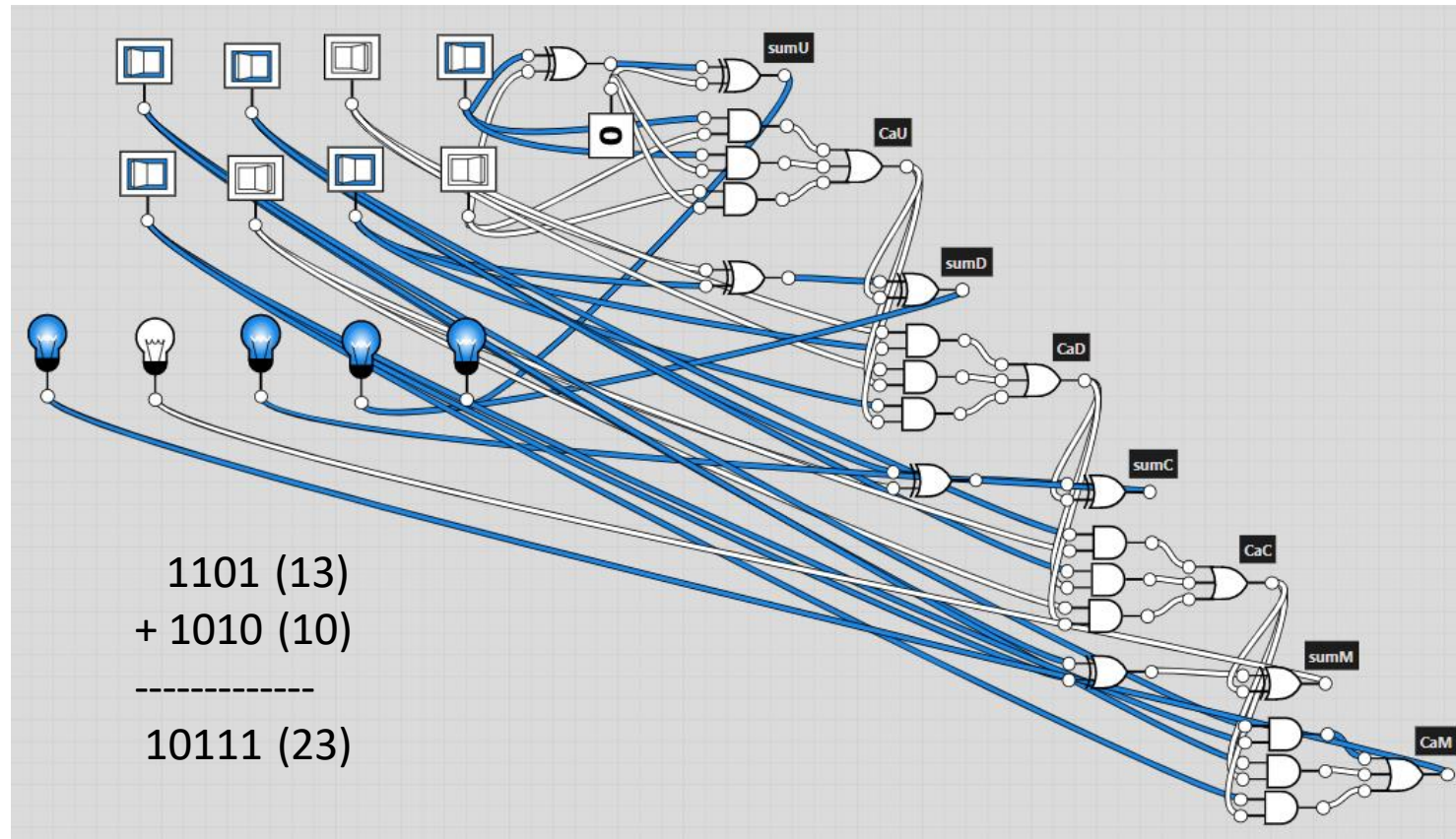
Carryout bit

- Actually, it can be simpler $CO = (a \cdot b) + (b \cdot c) + (c \cdot a)$



As duplas são AB, AC e BC, se uma das duplas tiver 2 1's..
Então: $A \cdot B + AC + BC$

Lembra que todas as somas precisam de 3 entradas? Bom, nós já podemos começar a organizar a cadeia de somas e carries que montamos. Lembra que anotamos sobre as 3 entradas? Então no caso o resultado da disjunção exclusiva entre as duas entradas, deve ser somada ao bit que pode ter vindo da casa anterior, e como é feita a soma? Com a disjunção exclusiva.



Também é óbvio perceber que nunca a casa das unidades receberá um carry, e vamos fazer o resultado com 1 bit a mais que as entradas, justamente para não perder o possível carry da maior casa. Teríamos enorme economia de portas se ignorássemos o carry das unidades, mas para deixar menos confuso, ele estará presente mas com uma constante BAIXA. Deu certo! Mas importante notar: a plataforma permite personalizar um circuito, onde qualquer sistema pode ser simplificada por uma caixinha com as entradas e saídas, então próxima vez que a soma aparecer, não será feia assim!

Bom.. Isso foi.. Menos de 20% do projeto. Vamos lá para a parte mais difícil, e vamos representar o codificador, que transformará a resposta de 5 bits em valores decimais. Eu não sei nem por onde começar.. Mas vamos tentar. Vamos colocar as informações que temos num papel, e ver se conseguimos pensar numa solução.

Handwritten notes and a small logic diagram on the left page:

Top left: $\frac{1}{10}$

Top right: $\frac{1}{1}$

Left side: $AB + \bar{A}\bar{B}$

Diagram: A logic circuit with inputs A and B, and outputs C and D. It includes two AND gates and an OR gate.

ABC	DEF	Result	ABC	DEF	Result	C & F	F & D
000	000	0000	100	100	1000	0	0
000	001	0001	100	101		0	0
000	010	0010	100	110		0	0
000	011	0011	100	111		0	0
000	100	0100	101	101		0	0
000	101	0101	101	110		0	0
000	110	0110	101	111		0	0
000	111	0111	110	110		0	0
001	001	0010	110	111		0	0
001	010	0011	111	111		0	0

Handwritten notes and a logic diagram on the right page:

Top left: $\frac{1}{10}$

Top right: $\frac{1}{1}$

Left side: $AB + \bar{A}\bar{B}$

Diagram: A logic circuit with inputs A, B, C, D, E, F, and outputs G and H. It includes two AND gates and an OR gate.

ABC	DEF	Result	ABC	DEF	Result	C & F	F & D
000	000	0000	100	100	1000	0	0
000	001	0001	100	101		0	0
000	010	0010	100	110		0	0
000	011	0011	100	111		0	0
000	100	0100	101	101		0	0
000	101	0101	101	110		0	0
000	110	0110	101	111		0	0
000	111	0111	110	110		0	0
001	001	0010	110	111		0	0
001	010	0011	111	111		0	0

Right side: A logic circuit diagram with inputs A, B, C, D, E, F and outputs G and H. It includes two AND gates and an OR gate.

Bottom right: $AB + \bar{A}\bar{B}$

Si quissemos 10¹⁰ DE, teríamos 7! = 5040 maneiras diferentes de montar o display.

Como fazemos pra descobrir o circuito que transforma um número em binário de 5 algarismos em um display de 7 segmentos que represente o número corretamente?

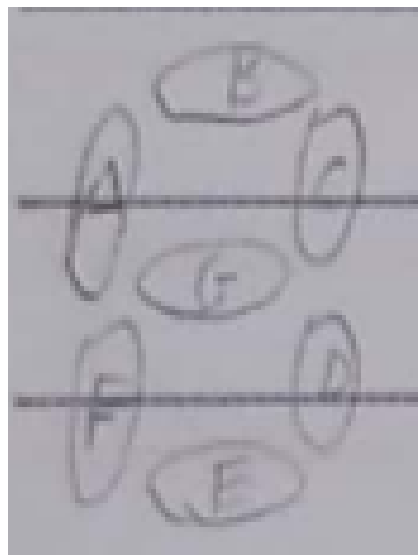
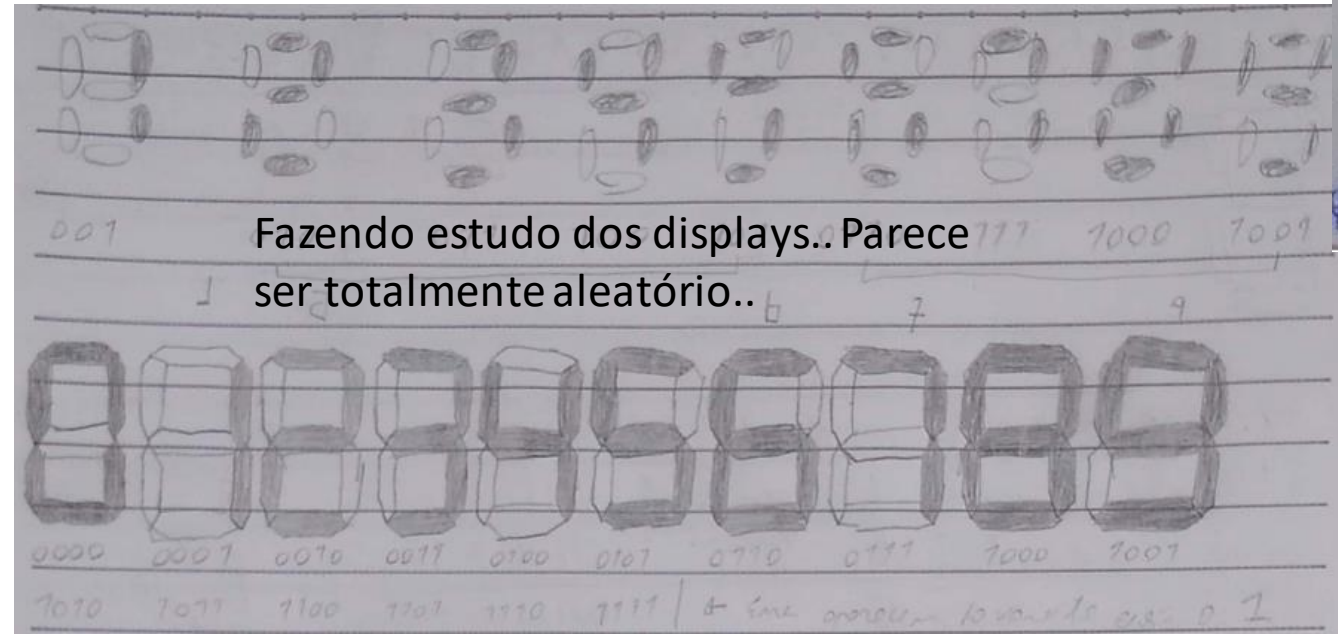
00011: 0000000, 0111101 / 00100: 1011001

Tem o valor de 5 bits que é o resultado de 100, 00000 é o número representado no display com 7 bits de valor verdadeiro, e depois do valor original 10.

	A	B	C	D	E	F	G
0							
1							
2							
3							
4							
5							
6							
7							
8							
9							

Não consigo achar um padrão que me ajude a poder aplicar essa tabela em circuitos.. Parece totalmente aleatório.. Como posso fazer isso?

Podemos perceber através das tabelas verdade que de alguma forma



0010	0	
0011	0	
0100	1	→ N0 A W A N E A N R
0101	1	→ N0 A W A N E A R
0110	1	→ N0 A W A E A N R
0111	0	→ N0 A W A E A R
1000	1	→ U A N W A N E A N R
1001	1	→ U A N W A N E A R

podemos ser capazes de organizar os circuitos..

Usaremos esse padrão AB..FG nessa ordem ->

Usaremos as nomenclaturas dos displays desta forma: vertical/horizontal - esquerda/direita/nenhum - inferior/superior/central.

Podemos perceber, que já que não há um padrão, as 5 entradas devem se organizar de uma forma que consigam representar as configurações de cada display, mas como observamos que todos se apresentam de forma diferente, teremos de fazer um circuito para cada um dos 7 displays.. Aff...

Relembrando o que foi explicado em sala, para isso devemos usar "a soma dos produtos" para descobrir as expressões que transformam as entradas nos padrões de cada display. Mas como isso funciona exatamente?

A	B	C	Q
0	0	0	0
0	1	0	1
1	0	0	1
1	1	0	0
0	0	1	1
0	1	1	0
1	0	1	0
1	1	1	1

Se Q é falso ignoramos, mas quando Q é verdadeiro, devemos guardar de lado A.B.C, sendo que os que forem 0 são barrados. No final, somar todos juntos.

To write down the Boolean expression that describes this truth table (and therefore the system that the truth table describes) we simply write down the Boolean equation for each line in the truth table where the output is 1.

The output for the first line is 0, so we ignore it.

The output for the second line is a 1. The Boolean equation for this line is $\bar{A}.B.\bar{C}$

The output for the third line is a 1. The Boolean equation for this line is $A.\bar{B}.\bar{C}$

The output for the fourth line is 0, so we ignore it.

The output for the fifth line is a 1. The Boolean equation for this line is $\bar{A}.\bar{B}.C$

The output for the sixth line is 0, so we ignore it.

The output for the seventh line is 0, so we ignore it.

The output for the eighth line is a 1. The Boolean equation for this line is $A.B.C$

We can now get the Boolean equation for the whole system simply by getting the equations where the output was 1 and ORing them together. This gives us the output Q:

$$\bar{A}.B.\bar{C} + A.\bar{B}.\bar{C} + \bar{A}.\bar{B}.C + A.B.C = Q$$

P	q	?
V	V	V
V	F	F
F	V	V
F	F	V

Exemplo, aqui: na primeira linha, '?' é verdade, P e Q são verdade, então temos P.Q, pulamos a 2º, na terceira teremos !P.Q e por ai vai. No final então, somamos tudo, obtendo PQ+!PQ+!P!Q agora vamos simplificar para reconhecer essa tabela verdade:

P	q	?
V	V	V
V	F	F
F	V	V
F	F	V

$$(P \wedge q) \vee (\bar{P} \wedge q) \vee (\bar{P} \wedge \bar{q})$$

$$q \vee (\bar{P} \wedge \bar{q})$$

$$\bar{P} \vee q = P \rightarrow q$$

Deu certo!
Realmente era a condicional, Pois Vera Fischer é Falsa.

<http://theteacher.info/index.php/fundamentals-of-cs/2-logical-operations/topics/2642-deriving-boolean-expressions-from-truth-tables>

Então esse é o método que usaremos para montar as expressões, a partir das tabelas verdade. Outra coisa que estamos adotando: como as entradas do usuário vão de 0 até 9, a maior soma possível é 18, então trabalharemos apenas com a casa das unidades, mas de 00 até 18, ou seja, os números de 00000 até 10010 serão representados por 0123456789012345678. É muito importante ter isso definido, pois a tabela verdade precisa estar completa para o método funcionar, por exemplo, mesmo que 11011 seja maior que 10010, ele obviamente não vai nunca ser um resultado mostrado pelo projeto (pois o máximo é 9+9, não há como obter o 27) mas mesmo assim precisa estar definido nos circuitos. Como são 5 bits, a tabela verdade terá 32 linhas, que serão: 01234567890123456789012345678901, de 00 até 31, mas sempre estaremos representando apenas as unidades.

Já não basta esse slide horrível, vamos pelo menos montar uma tabela melhor né! Para as próximas etapas, eu explicarei apenas a inicial, pois as que vem depois são apenas a repetição do mesmo método.

0	1	1	0	0			X	X		X	X	X
0	1	1	0	1			X	X	X	X		X
0	1	1	1	0		X			X	X		X
0	1	1	1	1		X	X		X	X		X
						X	X		X	X	X	X
							X	X	X	X	X	X

Bom, agora vamos montar uma tabela mais bonita. Perceba que não é necessário seguir preenchendo após o 9, pois os números de 10 a 19 e de 20 a 29 tem sempre as mesmas unidades que os de 0 a 9. Então a sequência repete 3 vezes e falta apenas 30 e 31 no final. Agora nós pegamos os valores das linhas de cada coluna. Por exemplo, para o 'A', que representa a vertical esquerda superior (VES), nós pegamos esse valor 1000111011 e repetimos 3 vezes e adicionamos os 2 primeiros.

z	x	c	v	b		a	b	c	d	e	f	g
0	0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	0	1	1	0	0	1	1	0	0	0
0	0	0	1	0	2	0	1	1	0	1	1	1
0	0	0	1	1	3	0	1	1	1	1	1	0
0	0	1	0	0	4	1	0	1	1	0	0	0
0	0	1	0	1	5	1	1	0	1	1	0	0
0	0	1	1	0	6	1	1	0	1	1	1	1
0	0	1	1	1	7	0	1	1	1	0	0	0
0	1	0	0	0	8	1	1	1	1	1	1	1
0	1	0	0	1	9	1	1	1	1	1	1	0
0	1	0	1	0	10	1						
0	1	0	1	1	11	0						
0	1	1	0	0	12	0						
0	1	1	0	1	13	0						
0	1	1	1	0	14	1						
0	1	1	1	1	15	1						
1	0	0	0	0	16	1						
1	0	0	0	1	17	0						
1	0	0	1	0	18	1						
1	0	0	1	1	19	1						
1	0	1	0	0	20	1						
1	0	1	0	1	21	0						

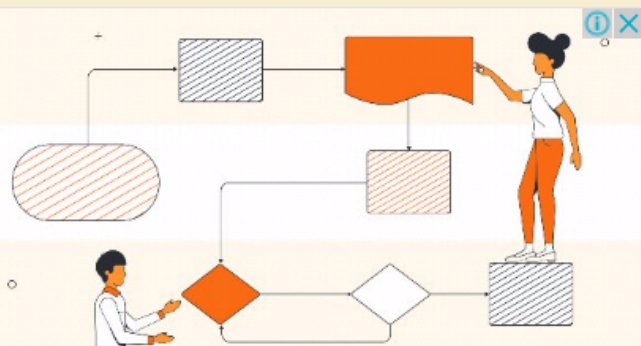
Faremos isso novamente todas as vezes para cada uma das colunas, que representam os displays da saída. Então não precisaremos repetir esse passo para 'B', ou 'C' e por aí vai. Mostraremos apenas o resultado final, que é a única coisa que difere.

Devemos também pensar num método para representar as dezenas, que apenas alternam entre 0 e 1 (00 a 18)

Lembra do método que descobrimos para derivar uma tabela verdade de volta a expressão booleana? Bom, agora nós temos o resultado das tabelas verdade de cada display. Já que o professor liberou em sala, não precisaremos quebrar cabeça fazendo a álgebra e as simplificações, pegamos o valor de 0 a 9 do VES, que é 1000111011, repetimos 3 vezes e adicionamos ao final os 2 primeiros: 1000111011 1000111011 1000111011 10

Results

$f(a,b,c,d,e) = (a \wedge b \wedge \neg d) \vee (a \wedge \neg b \wedge \neg c \wedge d) \vee (\neg a \wedge b \wedge c \wedge d) \vee (\neg a \wedge \neg b \wedge c \wedge \neg d) \vee (\neg a \wedge \neg b \wedge c \wedge \neg e) \vee (b \wedge \neg c \wedge \neg d) \vee (b \wedge d \wedge \neg e) \vee (\neg b \wedge \neg d \wedge \neg e)$



Online Flowchart Maker

Drag-and-drop interface & extensive shape library. Free 7-day trial.

Truth Table - dCode
Tag(s) : Symbolic Computation, Electronics

Share

TRUTH TABLE GENERATOR

★ LOGICAL/BOOLEAN EXPRESSION
a xor b

★ DISPLAY

- ☒ THE FULL TRUTH TABLE (INPUT VARIABLES + OUTPUT VALUES)
- ☐ THE TRUTH TABLE WITH THESE VARIABLES a
- ☐ ONLY OUTPUT VALUES (LIST OF 0 AND 1)
- ☐ THE ASSOCIATED MINTERMS $\Sigma m(\dots)$ (0-INDEXED)
- ☐ THE ASSOCIATED MAXTERMS $\Sigma M(\dots)$ (0-INDEXED)

► GENERATE

See also: [Boolean Expressions Calculator](#) – [Boolean Minterms and Maxterms](#)

FIND EQUATION FROM TRUTH TABLE

Indicate only the output values of the function (the last column from the boolean truth table)

★ OUTPUT VALUES (LIST OF 0 AND 1)
1000111011 1000111011 1000111011 1 0

★ TABLE ORDERED (INPUT VALUES) ☒ FROM 0,...,0 TO 1,...,1
☐ FROM 1,...,1 TO 0,...,0

★ BOOLEAN NOTATION ☐ LITERAL (AND, OR, NOT)
☒ LOGICAL (\wedge , \vee , \neg)
☐ PROGRAMMING (&&, ||, ~)
☐ ALGEBRAIC (*, +, !)

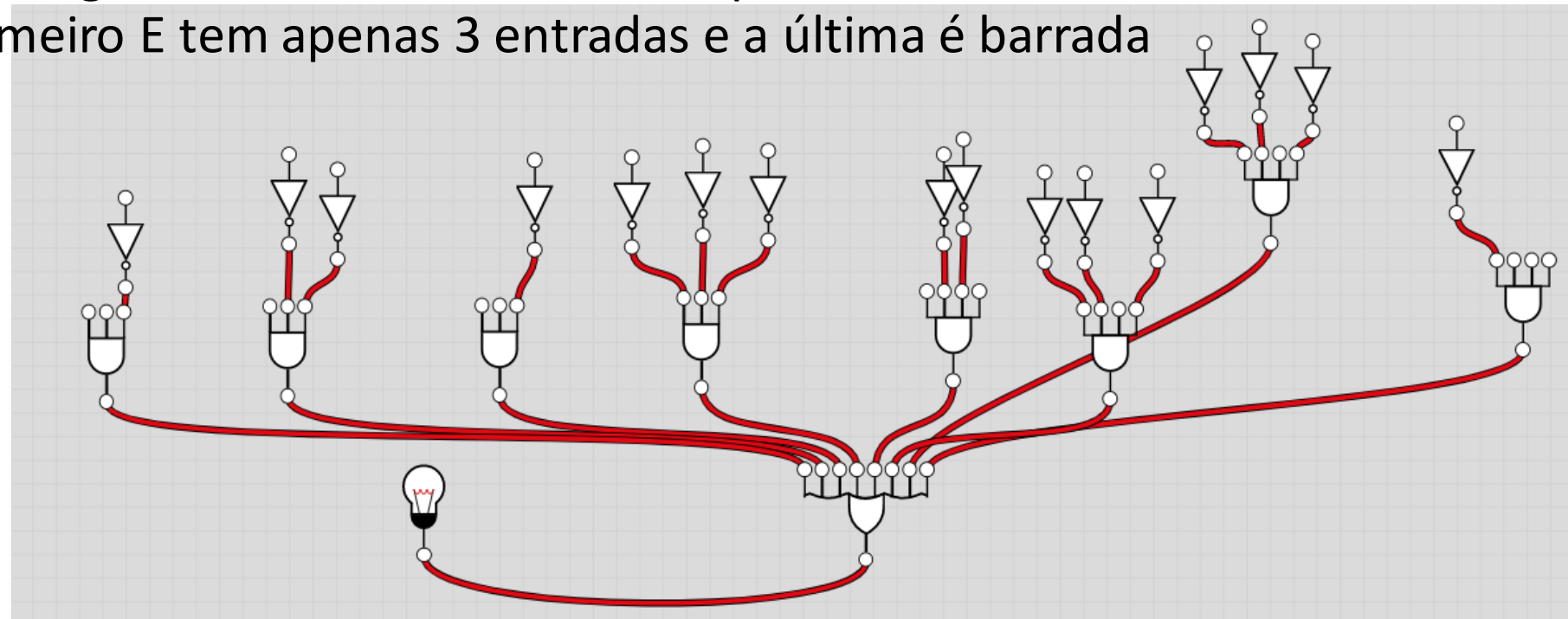
► CALCULATE

e agora apenas colamos isso em um site que simplifique por nós. Os resultados do cálculo do site, ali em marca-texto amarelo, representam a expressão proposicional, onde 'A' é o algarismo das dezenas de milhares, e o 'E' as unidades (em binário). Onde quer que vamos aplicar nosso circuito, precisamos completamente delas, que ao total serão 12.

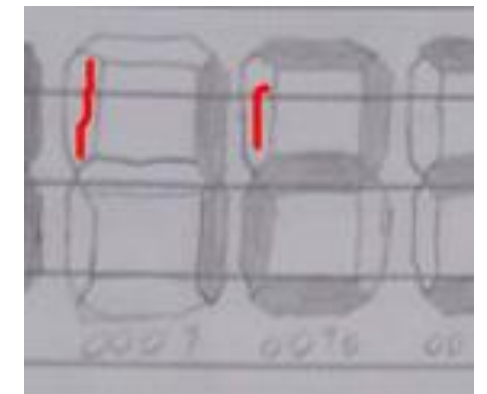
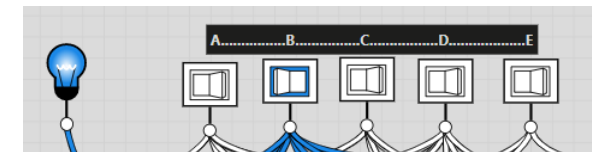
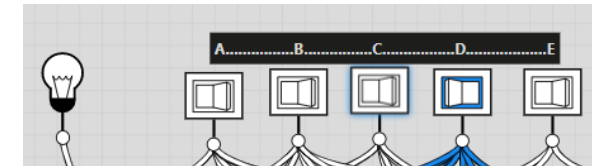
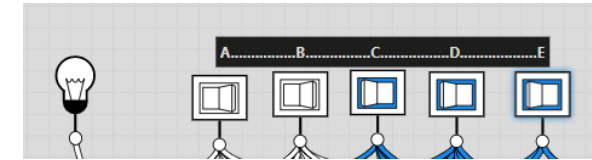
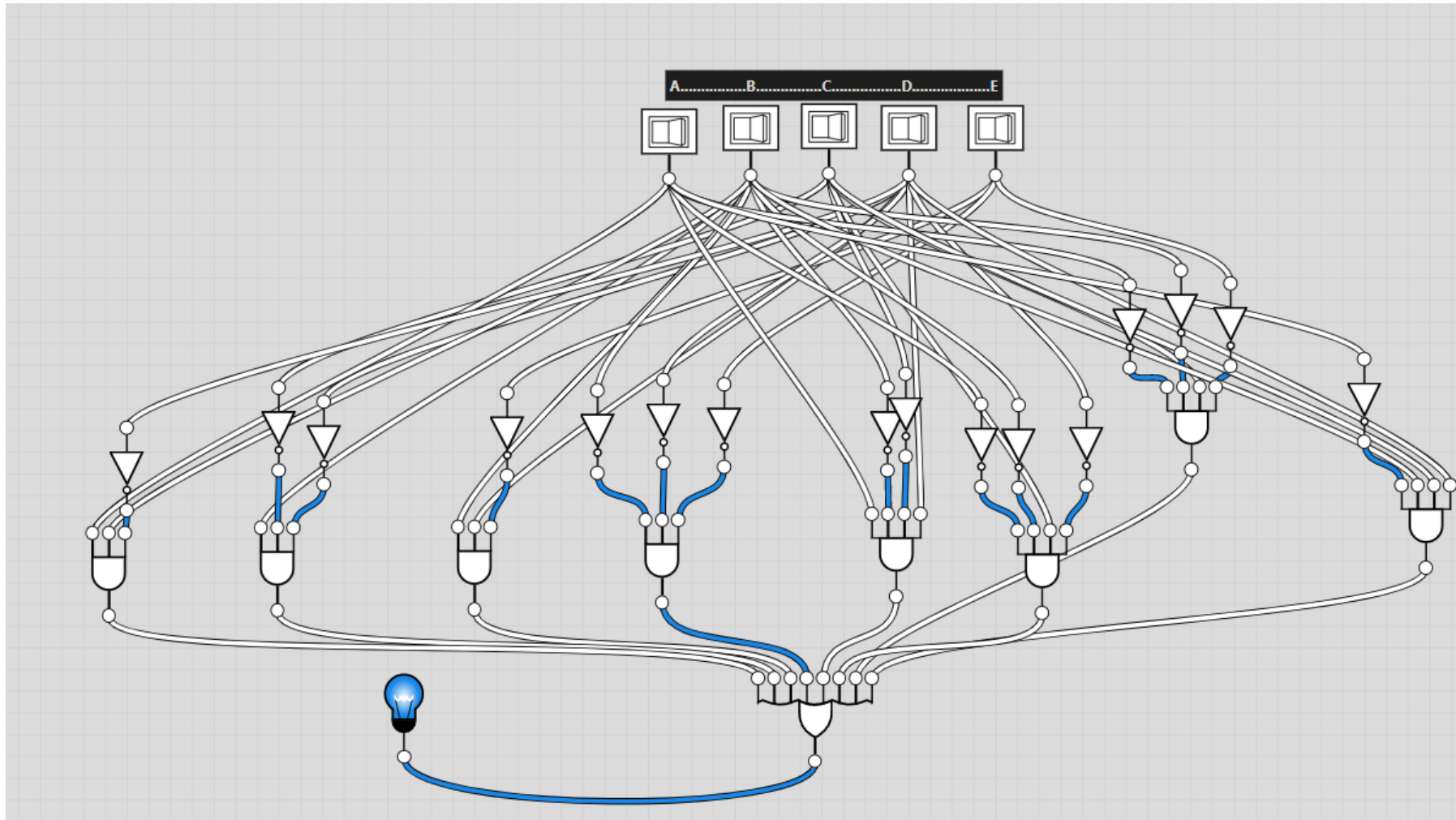
<https://www.dcode.fr/boolean-truth-table>

$(a \wedge b \wedge \neg d) \vee (a \wedge \neg b \wedge \neg c \wedge d) \vee (\neg a \wedge b \wedge c \wedge d) \vee (\neg a \wedge \neg b \wedge c \wedge \neg d) \vee (\neg a \wedge \neg b \wedge c \wedge \neg e) \vee (b \wedge \neg c \wedge \neg d) \vee (b \wedge d \wedge \neg e) \vee (\neg b \wedge \neg d \wedge \neg e)$ vertical esquerda superior (A)

Essa é a expressão do circuito que transforma os 5 bits do resultado da soma no display VES. Observamos que são 8 parênteses entre OU's, e dentro dos parenteses existem apenas E's. Sabendo que A, B, C, D e E estão em ordem, podemos também ir organizando os barrados, exemplo, são 8 E's como entrada de um único OU, o primeiro E tem apenas 3 entradas e a última é barrada vamos organizar:



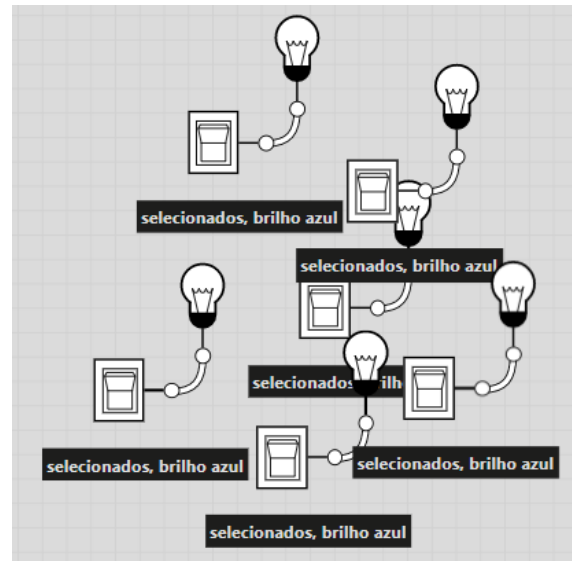
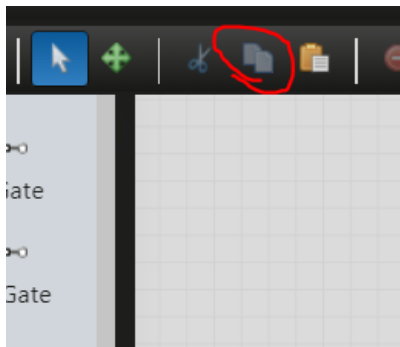
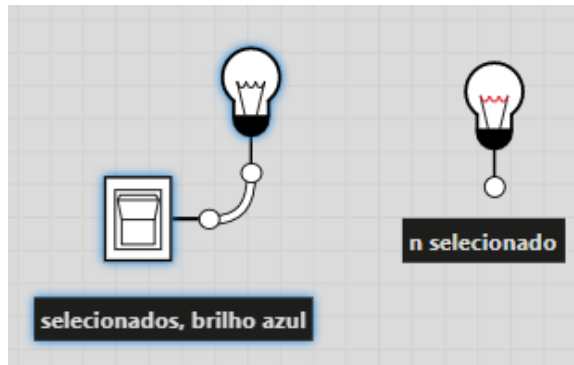
Após organizar corretamente em ordem temos aqui o circuito funcionando. Mas como sabemos que está correto? É bom testar um por um, pois as vezes os erros são discretos. Por exemplo, o 2 e o 7 não tem, mas o 8 tem os display VES aceso.



No site logic.ly/demo que é muito útil para a prática, podemos perceber que ao selecionar um circuito inteiro segurando e arrastando com o mouse, é possível copiar com ctrl+c, que trará um código bizarro para a área de transferência. Esse texto, quando colado de volta no site, replica totalmente o circuito como ele estava, até com os fios conectados.

Esse é o código dessa pequena demonstração (é tudo após os 2 pontos):

```
data:http://ns.logic.ly/1.6;base64,jdLNbtwgEAFwV7E4BzwMw1cVRz3kkgfoOQKM66ovYqxstunr+tNcnAuuTlz4sd/uC/zy5jKtbn8KdPSsV  
Ot5x9tOy1iL4hybaUw7OF+jr9zqk29nnPHyvyqs9xLfHnRxtr1rHvmHEQIGXgKQ09J5cjjwqJJ0Dpkld90Jo1l44hCq/BgkRptDVGE2uuHZPaCF  
DeW6PAASmNrHmda6jjPHUMWHuQLG9jTaejgmK0mQxw7aXkJO3AvTaKy0gDupSHIHFXbEJB1jkrwYOSUiuZM1CRIAQ0IMkq8ObAaOa1  
nte6BTaEsuQvrBjILkeVDSAjaMezU4pTsobHSJprTcE7RNQ03FQeBDnIAD2g27K5oRRYYQGV81J5jWQOppovdUskl5y2o9DPy10TX8dym  
pvdY3/kWmepk25ld9f8Ovp8btX3Qaepj5f9uvG6WP+e0vf+z/HN0z7/vce/gE=
```



Mas por que estou explicando isso? O que tem a ver? Bom, é que eu salvei o código de todos os circuitos que fiz, então se alguém quiser estudar, analisar, melhorar ou até roubar, bom, estou compartilhando com boas e ingênuas intenções. Qualquer pessoa poderá copiar esse código e colar no site, criar um "custom" (versão simplificada do circuito) de cada um dos displays e ver o resultado final na prática (eu também salvei o resultado final). Bom, agora eu vou mostrar apenas as fotos e códigos.

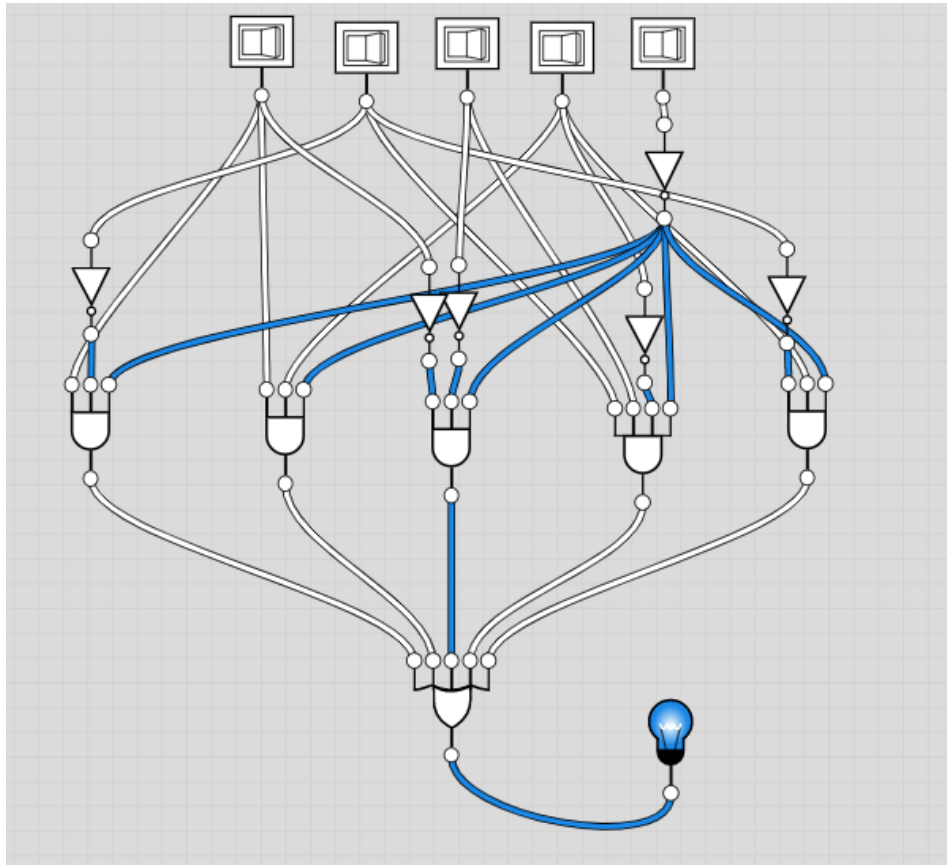
Esse é o código do VES:

data:http://ns.logic.ly/1.6;base64,vVrRbuM4EvYVwO/UNNnNZnOxs7iHe9kPuOcDKYk7OeSSwY4HN/P3V7I0wGlzFmhYSYLklbblcnd1VTWVX59e/ngcn74/fPvv0/OXj6dP5/PnXz58eP4yXB4Ynr5/8IOefvv1pf5nHs8P5++f54+nL/97PI+f/vHjKaeHr4/Tx1NgphySOYt5dpJLdjXGyRm1VK1ZSb6eHr59PLHy4MUi4ekayPLp4TteTkOkZMIasqeU/enhz5dzOT++PH88ZTo9vHw9f/56BshWnr7Mpw9doCZfdMw8uuJTcxJ8cnVu3tVMIGoutZpeQAnbgPfGu6uKMvEFIE9DAklyUfHJoh4CKmqZZxmLCzGyEx9nZ4nN5clPU+DSfA4XUNHTIJo5Ja/EOUZZUenAHGM0b1E8m8RDYBFZm3QaXa6lApZGZ1aaa+b9bDxbqX6FZTyYJZQks2R8iDeslU5FpjazizFkZjWoK4WTi36q2mjKyY8XUGjbEDh4VIU4pHwlpvl8/R1Qy1RHy+LyqABkpbhiai6EiWpplUKIF0AuDJwzhWgYiCA5UbpGekJTvc8+KEcxL3/H9Pi8QuJXcl5fzq8l3kYLuamzaVlnrQJOqN7xWJLV5seobYWjQ1TC+wFPyiQxrGOX/BCzcGBTBcPCNTS+Bw1rDZny6KZ5KU4cxVWailPO0goVs3HtFseByOM7RA8i6VqaYDKQJAKpNOXM/Vh+0qgiLc3TmF2BDDkh7x3GrrpaEk3THFEvumDxKQ+UJRigBCO1lc9o3iALbyz7GCjqfY2iGZ+rTMHNqRilbNVIMXZcdAqUJpmDXeAEQhFCAG4jrx5jvzVKBw+9lOgFW+aIzHdUh70m0NWchyw7kTa5HAo7xeyT5yal6tapMIRIMaFfxll5K04YUBRcJAYoUKBXk3VTccqkFpZqFag/itMypCeQqy2gJT5XuMVaHEuDDZzEUKBuHHH4Uh5dhi5i3jFbFq63qYjFNQUopGKlaUJsSmjPgckGk5iPcJPqy1kZsgH95L3xpTdIVDQwugdnJUB2LMd/Qqp/AgQZXg1+4UKs4vJV35ith2PFFNME10+phxPBQ9oAknFCdbao0DfBV1AzTz3RLcX5CHKqj65QmylZHWdWJrvA0uwxVg0gjudVcHxM4AjYGqlm8pnipn8yZEsSiJiCvjavm5hTZS6QutHJFBLkuBWkDiGHnzEY+yloZYMjinn2i+BAk2P2G5MJPYS9KiYcchxe5Y6bqlOTzDFAdGCWeRkrdtbGhLEyqSMomutK5Jgi7CqnyB61kR9gFnqrZPQQyUghydfQSE9x0I0wEzLG2BLMoRjMYUkb48xxzjSjBqtXYWIGMVu0DlPkW/pZvCHgF9IYHuPrneqicUvoBVwGyYlGgFybg/4Ex+RngnVitKetNH7wHg2MSRi6s4Ye9GxIajlZJ5Qs8X3uYKlkxvSatwLe8Bwhx3NwEbRuWZth1i5oEuVBA4hjPgSOLDcSB+hQkLTkMMCOu/o0za14VNhrQEGkxYoQDf2ZK/JOwrTFukZnFRpQwmWSF9OkH7YJa8iSgVohgkp3yh+XQhgiJzOChPgWkZkrEgZVAbUbGrWhAYcJcovEZZFl8a/vq0RHuDcCtpItLbwlJarOMWOEUmBUbTHY5QijqvDLWutlcyUiRnIsWV4Qo5lft9qE4ek2SBEGCiVq43qYk20mQsjY40oEPKNoiqzqJu8iFWCXW0mjsA5XALyggRE3sAkLDnMumRBLExPtBm2FaMYOPYAiIJCr8cZ3leStao1Ta1jcJpEeK4GIAGLBO6ZeMFlrRHTVJEm+6LfvPcKmhZweGEpQb5EsqXBNKtaUoVQ59WrYHRY6CS+iUD06fbShOGjKikWAUCUL7TMlPBGIUtO4LtORk9UiCE3wUqPGvBRujzCgYdQfaFO0FpPPO2YCVYIGQEa2QPgYvfQOGfsGY0rAAxwsAR2yDDCKHmJ4SKqpC3wrG0tTSL6g+mALTIMMcNDitokyRrBPvRxavK1yc2AdJLaA8vjBGqSH6MDDhKgQw3RTRdNwYE8EX5sF6iFnoR3IXDgvZxQj+xroM910n8ujYvf74Kfgw191KdjrocGIREQJTcpbHyCM+OGtcYinSFGmB/x34FldwODZCEB7QyC0i97DZXsdgrLE+Pf3w6/7t+faqvhrxga2htxNbil7yFSS+QfshOZEWrMF1rwgFLOA4gw36HFKhrt7CSY7e5LFMoHaLhXzHRgmN8eX4GEPxh2z//9fs/eze59QW/P0/zt+Vq6we8vL5vMb08f3u5vw6mb5E7EsxOZfoY3nUwfZG1E0zfHncdTN9a2gmmb2+6DqZvC/wrmLBTma616UgwOwTuW5qOBLTPtr4cfh1M31bR2aa+HH4kmJ029R0aHznaO5XpOyw+EsxOZfpOY3fa1CWanQS+tzK328FOm/qOz6+D6TPaXju4s019RvtObbrdm/bs4M7K3A5mZ5r6biPsjHaX679Xm7pc/50l3Of6nWDubVOF6/e69p3edLtR8ttxpi+CdFbmXgXuivyDvRODb2/SGnOk7CXwnnek7CHynNvWdA74TgftOj3u9qeuTHQlmr01dPT8SzF4g75qGI8HsKPDdo9117tY72l0Xuw6m70jynbzdjBvuB3cDuYNI7jwbwexwpu9Q/DqYvpsfnWD6bhccCwaHm313Uo4EszNN9xL4djBvuDf1ce6dwlVfm98pXPUNQCeYpp3Y2Si7bsP0hqsuAh4JZu8cuCsCHAlmR2f6Vo0jwezoTN8J95FgZOdGRtdhz5Fg4s4Jeddp5ZFgdCdCdB0qHwkm7bSp62l73tR1a/WVznzY/k39t/8D

Agora, 'F' ou display Vertical esquerdo Inferior

(VEI): 1011011011 1011011011 1011011011 10

$(a \wedge \neg b \wedge \neg e) \vee (a \wedge d \wedge \neg e) \vee (\neg a \wedge \neg c \wedge \neg e) \vee (b \wedge c \wedge \neg d \wedge \neg e) \vee (\neg b \wedge d \wedge \neg e)$

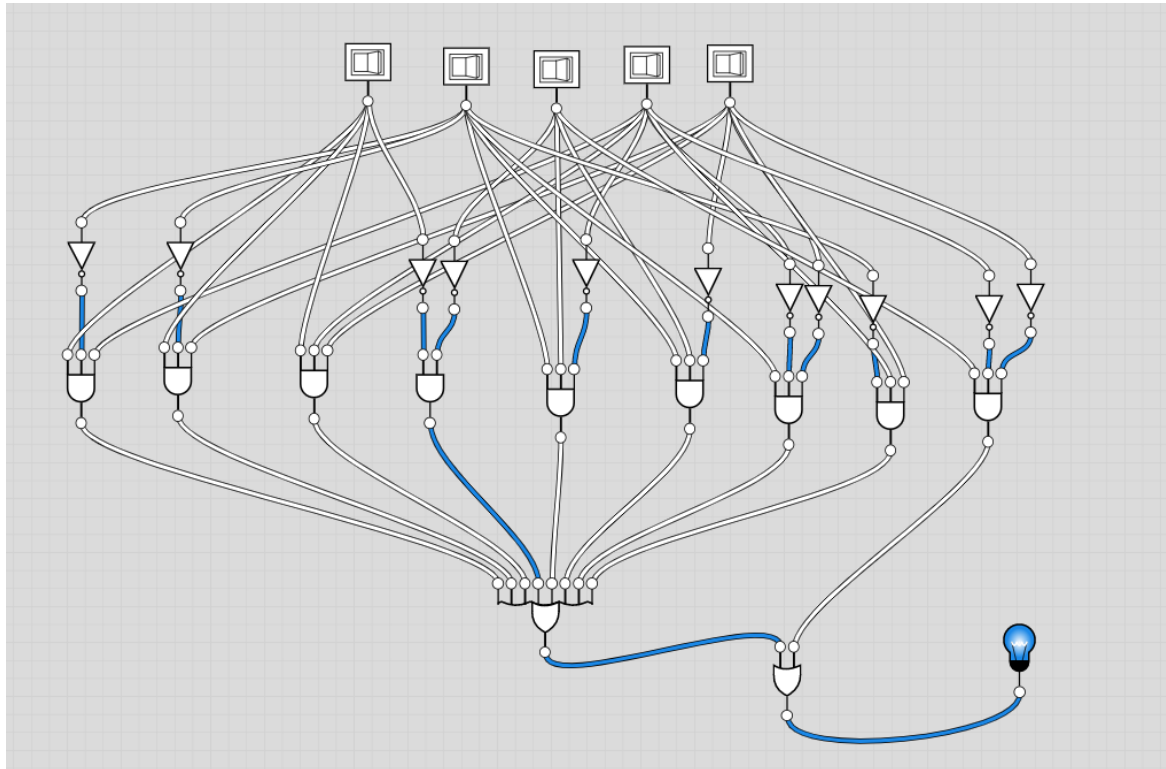


data:http://ns.logic.ly/1.6;base64,vZnNbuNGElRfReB96PnpmekOVoscctkHyDmY37UCRTL
WMmK/fdqiHFPJTGCZOsoi+Ln6uqqfIRlu/++SduX1fM/293jurs/HB5+u7vbPfbHP/TblzvVu+
7rl338u6TD6vDyUNbd47+bQ7r//e0j3eppk9dd1io7g0VlgipAxyqwOicKYQo5QlZldqvndaen
7uX4ZbrVy7ozqqfxy3erH/tDOGz2u3VHsIvtnw4PTwemrGH7WLq7NiQJlxyovoiBbhsRMySh
FcBIHNOa++PVNa68/sPUKbXht7f1DeBSinKEmUQLIUgK9SIWlBaGwlpKlf9REKvO3NFMr1M
IZSN4EqGkqJEARYaQQYclKHBglSSapmvk+NRyhD2NtPUiqWFGSVVkB0bCqyWWBySmAG1j
D6mHBQikf8q/FdqVTY5Z+JMFLFXIPQ2hLbHEmg1FnkVMBaEzQldSRCOhsdnoi06XHk/QnRZ
jcAmRYYWXw0NiPb2WWemWOYIFE4aXVCZwFsOsI4R70cCYH/w7DF3mHgKpqMYAOVKrS
RmqVRSlAGErZkKfkuIuk5JAC43o1o3qRBttA7zcRBF8GUZJOMmmEiFZaG2DmQvZBeoQsR0
WAenIPqTJo3O/sex9JcBaOqC8VSErxgPKfKHOSqEzHlml01kGmwsQXT++mcQPf0zvlamL+G
gQnMbn/4GaZ6GYoqRaRseUwyaxGiJ0HaGcCiHFEZHAznew5HGC39WXyrSVC/0agWGgaxiZ
0hKhWOHe3ZNC4hz6llyR4yFdwgDdfGNHaOVhrPyV5FY1UgrFxiKjAlRN5zTJE7rUbKqKp1Ya
glB+yPgUDBOChsdJoIAuYUDHuD84W7QXH2cY1yDFcNXLGeBXrze71QKB94xq3TcRK52U
mYSiyBpiKiJa3OqNOxJVAINXJrL8qAqXM+eZcp4f2KZqKScSgWI/ALU48HAGQqjHEe0WDP4z
FXk39oTn4xjSunWb/Y1JHwCElOfKdfK0jJCMc4tNODWAxOlOsPsGQO8/byXcszC6Lndx+u/+
f/grPm3j5PwARVMpICsfWvhQE9kj2UghoySlk6tQTocahT2NdnXwi+XFhrkOkq8cab/bMQi/c
WriP7/90RocwwXfdrk8v37b8A8er2+rzupnT5ereZi2g908TJvLxzALyrSZdB6mrzGMHoepi1
BbgmzMka2QJ2HaTvhNCRt1jULnmq8VYDNz0v3XKbFpRpO2fPw7RFQ+M2ta3mLZVZgLI2
TG2H4sYxtT043hJmwcDXjulymA8cu1uCN8K0PurPw7Ql+Cdt0+V1sABzbc5cnsDm48Z0Oc
wHrnZbt32SZy5X5gM903YEaYS5dkyXn2c+MIEvh1lQppq1152Hanq3GMLBQB03dckuYpZxpS
tBbwiYdgZtW85YwCwZuM+AtYZZypunL5mHansonMHen32C+/gc=

Vertical direita superior ou 'C', VDS:

1111100111 1111100111 1111100111 11

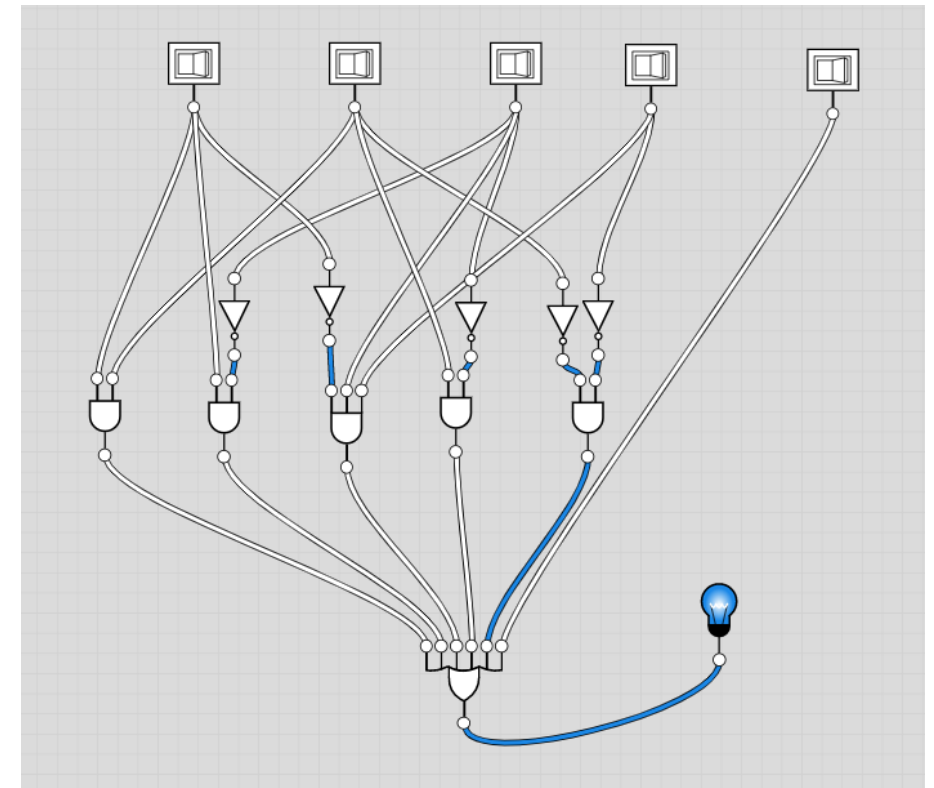
$(a \wedge \neg b \wedge d) \vee (a \wedge \neg b \wedge e) \vee (a \wedge d \wedge e) \vee (\neg a \wedge \neg c) \vee (b \wedge c \wedge \neg d) \vee (b \wedge c \wedge \neg e) \vee (b \wedge \neg d \wedge \neg e) \vee (\neg b \wedge d \wedge e) \vee (c \wedge \neg d \wedge \neg e)$



Vertical direita superior ou 'D', VDI:

1101111111 1101111111 1101111111 11

$(a \wedge b) \vee (a \wedge \neg c) \vee (\neg a \wedge c \wedge d) \vee (b \wedge \neg c) \vee (\neg b \wedge \neg d) \vee e$



VDI:

data:http://ns.logic.ly/1.6;base64,7V3LrhvHdv0V4YxPtev9CK6CDBwgnmQJSJNOgntcKZMmwjhm7X59V7D4PHpJ1d6ubzB1QhmXBksjF/Vpr79rV/Mvnr3/9ID//+eGPxZ5+/fbx4eenp1//4YcfvnybDr8xff7zBzHzh3/8S/7929PXXz58ib/Ujw//8m8PH57+/BW/4soZkaJlIbrMtFcNv2qceVmD9FHrovfrp7L1/RfNT8tr/3tfz495Z//6fmPPHz4/VP5+JCOqUIZz5y2nGmdPlsxcNaszSaLmGyMDx/+PiguX/48OfHB+EePvz29Sk+ffr65eND4A8f6h+/fv3t6V8PH+V/Hz58/f3p19+fAKvFz9/qww8kGMHfImzQzNUUmTbGsVhlZNa11rjnLil5gGGkmUQBiJxEePvDDIH98Z3AvNFWG1IZTElynV1h0UnBpFRF2OKNjfkAzBolm3dgTPgjaN7bEbL8nci089E5p1kNyjEtbWFBa4ROjiU61Usq/IDMefWCTK5B9t/ficxExqR3gRVZE9MiSQRW0yIzLwUpTmXDSiCEvM1iBLfxNZ/FJODKaM0y16hOchK0U1QOQlay0GxHw2AH2AJWYwPdgOpxP3PCXH/I9qkfzSHKgsBVIXPMMoUa4TLfEvPvaqZUxrurmk3Us2G8nfyCQQRJO7MTAqtCsdFnJJYOTFnmFehMZPxn1C417UdEHjVP/IMQPMdEdQ5JC1VMINQ2HT2iAsJpOQJRMCT75GxnL1j7AgNtIrsfGomb3sUohG3xhWinU1YwLmpvTW0Wu5DLUQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCRnttlNwzOm/TMW8Jlbuw2R/GKz4VCzqqDfAODQW9rdH0q2iKhXKVDU11rYzpojkiUSF1rknMYW1FjsJnUnTFsyt2wrcbgsJD4MgC1rwAgoY8DUmG1xgy3yaijMTZJlVnT6NRLfJl4wrrIUuWcFiFYNcsl2hwqZuEH1sbMHIOReEcM9VJqk56ddkvixAniD48vXpPYKmlt4zctCGALJtYNwgfEJYNNOVrllouyDQM/dj7Sb79v0vAhAUAMIK1SsSi3NBOvwqcsarRzUoOeewmCDoSb2tjrM5ZLC0cSh4Sdc4KPiXPbilHRk4N3CVDatJc4FgmQpj2Yyb5l3AeMn/RaM2YTG2OqUspZBD1SEh5MID22YsQOJEitvC+9Kfw6NAtPtZ5qki7O8KRaNgaNkCcwoC00AxkDVCiWpsOQrn9wZMOI4dU+4bRUaz6OpNeHtue/VlwcUT4g38H5IVXInvZrrF7fnwkYpdZzHehMc8Akka7WQbuB47bWEXYxgtiQZVlxGRTMX9HDM8Ytx5HFIVZvAOARpqgaJBF0LUQSfHRIkk1Ik5Z0zXpoXdlGnYNSuQdySqsqBbRN3CnGD8oY6h1Ircu3yNvE0q1qr5Vk0dkfT5Oy9Ri4zITzy20SJ3sJYJopCC2CNa2UOG2/49U3jXJTgGMiAhBjqj1fMC9eLL8LT55JYylsUEmfz2x3F8ArTfP3tpPmQTgSPeJG6WNThCld1y8hqrCvVay/DEjRhEsdq8M+u/Pkk3rDCRSz+BMvnT3/9+ek/0++f00n9AztP4QNlZfe+jCPdhUNWcTQi1ddibZ1LjnRH7z5jciHERzXvnYBGywks+euXlWCDP7DI6X//6UcqEcX/4acvpf5xePXdhzz8fZrOPvz55a+Ly2BoHlknGH4ZDI2ULoOhqS4iGBoPQZDE6FEN9E46TIYmiZ/COZeBkMruwMwpBaFlCbGSHuCGbiJxtV7ghkEMIOClOOhddNEy9DY8TIYwMNNtAytM7kMhtZbES1DmxDuWYEH2USBEw6KHonbiG6izQYvg6FxFg5kOtrlpPTcN3LTVMuvBDaKYNo4cZBOJ9W/kpvWsfUu30SQI0TK0efaelrmim2h66EZ1Zr24umI2rQdzRTfRlOKN3ERTitRWZaObaEqRCGYra68XV1cseuvBDLJpa9GjydYbuWm9IB+4aatlaN3FjblJpvWwU6CZad3Ej1l7fqozAkFqNy2Bog8C3YNYg1yZp2D3B2MFMjySO9gRjLoOhSYA9wegBGNJIY08w6jiYGrfsCWY0uSivRT3BjGQnqU7sCWZUGUkvNih6pOH+CZjPX/NhmV+hHxD81o8ODOcF34gv+P4VPtf28glUif1IIZ5Hirx6pLDGI4I08aF+WNbh8Kt5re7det1//PPLfh3swb0tkQXkE9Ouf4KEd0DDKq0T0hTL1992vg9rgCBfPvAkQ9SHCR8YU5nlzyTeYQcyn6sqqSaD+cfxxK7lftTX5ZLjzWlKzBHD8XGP5rohoc1C5xwzLQQRHuM2CpcDh0x8T8ss65j8DBFArrdVW9Z0uw0bg5AAyXQkx6ODGa+zhlWYq6TkyaQw6NYHk8055UCqCJlVslwLgoJPEDjKOWG5CsboGaSdlDLGeIEAFcrrk+PLPXbyOLrYgsEVY09ZK1h3seDBEAaauqRFDNKrybvHQwWIA74TOst+b3beQh/XVpVzBiJHEHqATEiC40oyTZeghPLRqPVk1RSwGZcSRfWQ/yuNb0W0EOEqbqkZlFvigDFj01qOAFJ5QglhcjvDZ4cycS+ORW1IHx93Lzp4GeQtpldFeXfXvON1TOrfe0KAgQ7PMI17rGypfIEkwaPzqYls7Lwx+XkLMf7AY8LHDVwzmDXz6+1kspb3DctomiLNPvDtlEajeOATsnjnoG4QDhy6P3eXaeMhPnAv9IIXBmdjaN9Hri2nGEnHUhrDg1P+OoiNahlj7ZQkvjmgSBOkClTtHxUqo5FNPDwpALEw9aekCR

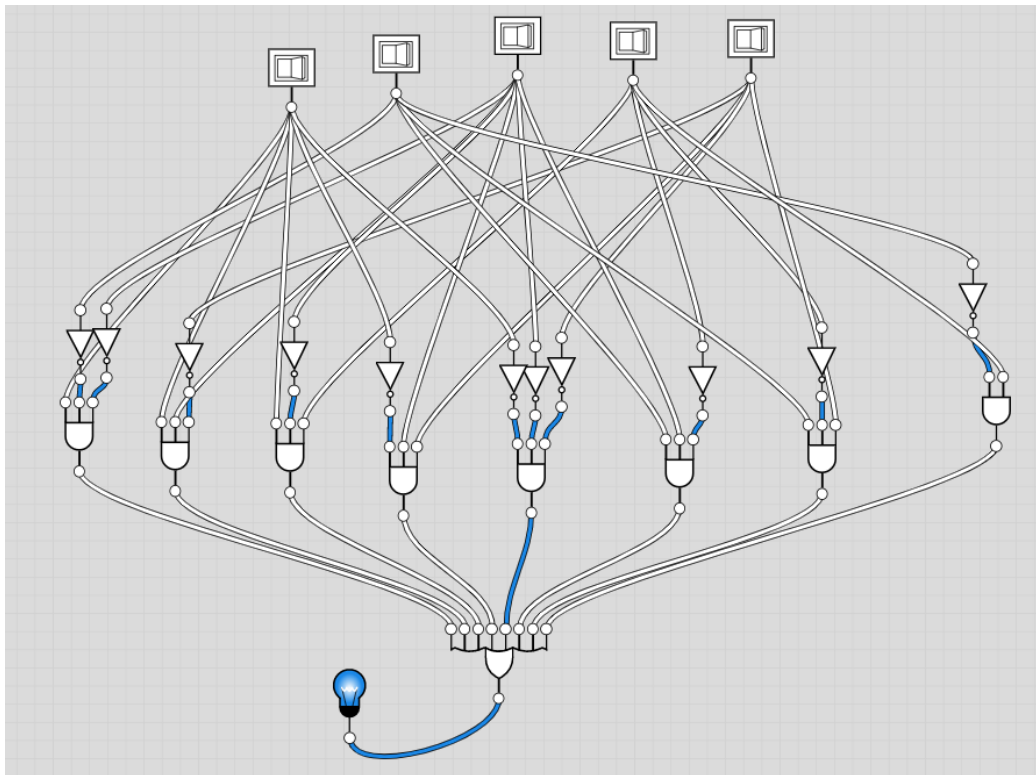
VDS:

data:http://ns.logic.ly/1.6;base64,vV3LjhzHdvwVYtbMUj5PZhqXhnc24LvxxrC3Rj6vaFACQRzall/ekV01wxl2d91TrOqmiGogznRHn0dEnKxTxb98evzbx/Lpz3ffv0+cuHh1+fnn7/h19++fxlOv3B9OnPX9RED//4l/L1y9Pjb+8+p9/ah4d//feHd09//o6vpPFO5UQjil+Eabjyq5F0C3qkKytZvw0910e83+38rS89pf//fhUfv2n5295ePf1Y/3wkk1nryrgvCUprM1BpBSl6ETFFZ UypfTw7tuHByvDw7s/Pzwo//Duj8en9PTx8fOHhygf3rVvvz/+8fRvp4/yfw/vHr8+/f71CbB6+vSlPfzCghF96pmiFb7lJkxzXqSmkyDfe5dB+mzOCyB TZtInlHpS8fUvtwrs208CC86SdbqJlJUWtvqgktdKaG2qohocpXICRo4Q8wFMqPAGWgi0hqz8JDlrQ/LeW9Gi8cJqjilai9lpqSZvuspNnpD5YF6Q6S3l/ucnkbkkqw4+iqpbFlizJlQLuVuRSZLe6du/zCVIU35G5Lcjy30WWPtezgBnnbU9BoOCRStUcEAUtek8RNV8cQJ9gKqYqTdidcVspJovnyS/+I6uPnGYXhIPBN6ZKKFbElpCz0LElJk4wpeHfTikt2DkygKSwltJ2CdwchIBMrpVDQWDYKS96LYGlXruA/scpgWz8hCGZ88BmBlQciaDU6qk0LV0FstgTuhUZSkIEq4l1KoTkLRN9jYN1ExyHoeB8yBt0QqrDgGF8TEPhkg3NN+lrqwjX0PQbhwBhl0jkn0FttrQsbrUQlarQuuaw7qZyqLaSrXipxEN9hCKqMUalbRA8KCDr6NFZbRxeSsjNFuJjTvzb2eyWqA7PgfpW5SCVUrw69QF2AYfMliiY2UenR67gWl/DNLGK8m+70ZryPQZwg+Pz79iKbBwm dGD1KMENsOxY0qZJRFN8GUVPsIBYGdtU9bP9Hr978KQHEAKFjMMJJIMyGdvkpSyBbABrWUEpcRDuZ1+w4h0NH4hUlc42E4EbJfJb6kF02SUB3qzDZF92V6lCISHr7d5F4r7wlmTPY1GLcLJaPmjSes8AMK5NE5E1ysM64ajjQVKTFdFhQ6hMVC640KfTbUkuxHJOSSqRi+ilXGCKAZYK9Zs4tKvcvIXwKi3rXumbZvQBjIcaxlvL8NgjXlBnjBv0P2Ym5ZeBzPl6RLZWOMedvHdhcc6AksayNYN2i8DVYjLk4JqlHk5Izyc2EHt9q/Blc/ZZZS4wHkWam0MjwdfCFCFnscYqdFFHBO9d0O5FXcw5GHNoEfdsMvFQ2yy9Qd2A3sBzoFpV2rC3WebZ1ZLVF9HQgaEpJQSLXhZKBfS3SxqzhSOhqS EIQM73OpdNcPL2ofE+aWgMbEAGx9gWjAjKD/JfeyZSi4llsUlqYn/7NzW8ITSPf5wNH9qrGFav2IYCDyckBERGN0e+tmCDjKvRxEm9dYN/DucvJ/VKFa5iCWdYPn38269P/5W/fspn/Ad1sipEkXsYc5IEuyPPrpIYRPolajNlKP9m3efMXkQORvO+8FAY+QElvL4+TPA4BsWO/Off/1nrhDMP/DXz7V9O7366UOefp7ns0/fv/y4ug6Gp5FHgPHXwfBE6ToYnutigUGJ0nUwPBPKTBNPK66D4Xny12D0dTA82l0BwXpRmGB4inQkmJU08bT6SDArBcyTgOteNM0Mzl8dbwOhjdYmYPdM0yug+HNVSzI8E4lj2TgIW7inROukB5L25hp4p0NXgfd0za2HOxL03ZtWknT3shsB7NSwLzjyJVuYqn+ndK0XbVvmCaEBWFGHneefWRkbpgmnh+6E89sN1c37KbtYG6YJp5TvFOaeE6RO6rsTBPPKTLB7Fxt7ebqhqS3HcxKN+0lPZ5tvVOathvylTTtjQxvurhTN22PzA3TxJsu7qTa20eVNTCsUeM6GN5B4GswfmXWZnnYl8HQypkeyxwdCcZdB8OzAEeCsStgWEcaR4lx18HwtOVIMGsnVyzSOhLMmu1k8cSRYNYYmPViK6THOtW/A/PpsZxO89+NCwR/JEsHp4sFX5gv+OMrfGr95QU4Rvo9R3jec+zVe45qvGeJlJ7UL8s6HL6a1+p+WK/7z3952a9DPGSgmkREPwnxyfIeAcMrJq8UrwSPHa/Dm5DolyCCC7C1MeEHDIXRZDd59ARBjVfVTdkJgX+k/h20jsV9Xl5KQP1pCOSvqobrF8V1WiEk0ZNQNx0MqL3LoSOUrkNKacw7KOMq4BAihpIktmXOk6bQROHoBBTqhBjwTfyg8PZYNOLOlo5zCpKTRf8CZAUpE0bVJXcVkJVHKCwTHeK5lM0mdnKDQZ45wLCgWqTLBnly+P2MmTmPYrVRIqnEbJkhMhpJMFQBuafTaUM8pgphA8AhaNjfhM2yP5s9t5KH9bezPCOY0eyHABKaELnaqZuqzRq2WjkeykYValmTtax+0Qf2pNrOCJowaWiYWAD85ApEABBFJlBgFjVzVzCF6dr5lK7gN7SNnrpX3b2LMRbaTlOBnV1veF8T+nSekoHg4ydRKId6zuYL+mshCnJh9xvCTRzyAmHEm8H/D4KMGBcwf3f3aGm0CEcpX3yaKIYzfE9a9thEcB3X0EnwG4wDjK1MIZU6ecZOUCv9op1BmNldGBztJ6yVKjnyMG66aX0hUwujjQZAJMJK5WCHIcpc/Ky1uZOZHpaGPJsktHqACg6SKf2yZC8y66pZyWjvYlSjZ8LhC5aB72DQoGv20x9ZIEVcNT2rasd2qJlGgPHEGOI/jolkTRpEAuEuRiQJR0tk+8KTpGkUe5BgEdBhXYDonTyQgCM0hluoWmLpnSk3bSeeQLqkRmCY6eEBS8iNOgKy2v7gyxggN7H/SiRoKGLDg9gpg0hoCukRIVM4RnWafyKzI2IEDQSB31c3DMaDaHfotlldu3MgRdytklNFGGhbNJdyi710ln35WDEjk1bx8CyAqPVMqaU2r8svgGrfSobB8QHfYwMcUOqLi1UgVUCxEVgRrKYR1DIQWXYoD58jawQYL+sdxnIsVGAZl1HdJaulj9BohEzdL+RW4JzaTkOe5PrsE2gXmCwdRnlibaLihYlggJKa93SEch41gmrVsJ1SxeddVdR3DN5qKY3UdK50myon25bK6W6Eqk9TQIKBwfilf4sORIVX0wLHEvpZDcIBJ7uolkqWyCGOMNDhoGN9ZmE2RSd725wG6UBK42grl0lvHm0VbC5jbSnPhey8g3phgjMKsbHPYEZ5k43IluwWgZKvobGc4CABukkYktl9xCGFcQUP1qQ041qUDTGYtQodM9kQBteBi8LiniY2aPwGY4c/M9cxzds288FVAa+Q4IATe4C/KOFkapJSCdae9lrxtsqhQQ6P05M5OyQkLPJWxRTMB4h82afOgSfokH3oG4t6sY0BzoeYzbKumsVAnptvj1Bxok0CicozAhOxqWINXhIWz9MG2CGXXmqrSeFCaupERDbHUUZZB/5pGfbHo9tczssinpWQwTHJQzTls2xCGqKNQE0gQZl76c+kJNFewjYYCau6g9/OcBgYW5R2R6IWNKhChbqFAQvOWL1slaOGHdQb5pxkGCnchYYwG7qlFvLxdGgWtYgOQpWhlZlj5naLpfDGw0OHCE2lDibwOTZu8oRpWCs0FNmriWjVjQvNJAOPVRAg+BtCVBoGw6qsDVlCrhYRh/+cTvZ5IEEH2A85iVjaHhBzF1uX9U0TDrBuA6zNeZdTepQREmhwGRd9trX0rYDyJ2QwBIY/KgXsoPiOAeCpg0kaZ91q+1nlGWGTU8rlXBXL5MEIXGObqM5rez1wDoUdDeVLDA0eJl/IHTx FB8ZAKGKC4UzJ9wsyVQxTSjAOHouACQfxCy2QaJnwQ7LwyGZARef+oE5hGGBnMYx4SZSOMNbyHhYpvKOEVLVMCrgDnlOCwbaBhmNCgKkxFtBbMi710TSD9adAADRo2LgFjiGUjbEHRKofWbzKfDyy0aBeifSYUTFWZjg/Aw9YbAiNd411nScGGPDBfJhFEQs6Ee7pJgOkz3jkE5M/qud6EXN2W5MBmyubBRUa5w8ZFICNi5SYTYfmhuxmGwp3hRh9gsd8BZZczh/ghCekMloU9ZhtrmLZtNuAMDX0XjC1qOG30OkJ1A/acYaQKnTX7HBQJUghKGg+gwukOVuYy3zHbnlYphA7WcGXPDz9XD+hZ412KxdSWJMqF84CTN9kdCWb1ujrH9K3sYrA8LBMNb7C7D0Y3pLB8Aap62B4YyHzhJ43RxDZm0FmDVFHQLmJU08Y76y2cQaM7j7yCxfjiSYIT7dQSPbO2YpPBOno8EsxlZ3uHt2jYchzSZBbw3MtvIYCVNvMP3lWUiltBy5WBNmnhCe6c0bdeMNTnYGZntYfa6iXeZYaW1Wap/rzSxVP9OBcxTfSaYvWniqT5XtXdq03ahXNnF2FszPAvCXUDbycA8C3KnAt6ephvWDO9o8E48wzsZvFOaeAeDdypg3nEyV5tYn+xlMGtpYuX8SDBrhpzVDUecWWHg3a3NOojtjbvxY2tVnlHfSpu1gbjgdbAdzwyFuO5i1O1xZp+Qre4KsqyFMMLzrB0eCWakZ3qWVI8GsdNPeAt4O5oZzE6/m7mSueGm+k7niNQATDI8nViZK1nUZrliFeCRYNbOgVkw4EgwKzDGzWOBLPCM7wT7iPBrNz3wDvsORLMYh0hvnPKI8Gs3SvDOIq+EszKXUS8F1vRjta11jOeWbnVgPOCa7cacDTIPYfr33N0+z3HEL5nie7lWw049wBYqqGLOu67OuEcpYUJz/uWazl9eFq2wrWaUGLSeFL4f6SXZ6NKP0kVlpGn4MZq0o9X4X9qEXzseJ4KOyEJUa38lvHeoihfXvpXKmHemohJoSSqc1hu8o7sC6yztYmfgtViq3XcwZKVRuPjdhEDKRFVrFzVR6RttyyNK1RQuwbKTeg3rbOHwp2BFUh6ND5bEK4MVkkcdVSc0S5SO8wTT560OMnLSl2Dpyb2GdbZD9nOwCoJlRbV+EFW0opsoLu91l9GCPu2aVAtOQ5rBgnsydYl55yW/A5Y0KQCogKKYKojtscB1lgs4pLo92U5Em7Yx7gbQscRk9DTTGaFCMd1cXRnlPewVblnGz4emGepi+JEtBU2ekLJavaH5xhGI0WbRfo/Qso7jDECSPAAniTKMu3yEMm7cexvBVh5eqQx6zzkVVPi8+SzNGzDLzQToRPsazL5nsBpb4MlRhLEDg1sz7gpNmBWybaBAVctp4XowwHj5jkaR2/R7HskbY1KRxC1lBgYLBQYvtGr4a27q6gp87LEhaJVr9CoZQtbt+EwNAkhkUYGUWncKQbdh/ZELXl1tdceorypa/UeWhsODJRroUFMyaMgRP7XtHoirkqteCCoY8+rKs27k3aJYaNuHA0CAgJFu2QvqO0CSPaQMfAmJMCgagypKWdXnnLtawn+g1mKvbdRwaTr1ZCUHXZyirrCDAqkA+loymhCGkpOWWK/mWbS6SzfXdp9YzhWsj1SzqJLuugcZgxRsgo67ksA2Gm7hvh+kfoYz+oyTKNaSJgZ+20qEZFGRSBQ6KlmaGmNhn4lLSnZlmfrsBEv2Q2swkPU7Wmu9EvO+5tSo6FG1xQiYeyGo7xh9MHA5+spXSgLYucJyc6BWCvQ0ajV6j2fnwZZdr83acb7s8LldmkapCX3nlprbjDIWz0l23F9D8GJt9D2cNRpaQZBQ96zroBn5VZ7h+Cl5aOay17NM6r99karkFU/6AZl9skqIOy9yQGgsFB8HAUOMzFYpY+VYabnSG2Wmcf44Bpr5Gs6+fXmjZVMew4Bt4+JlhNnPQHlGZF9gFKOvZ4/NVxfRHfHo4GFO6l9Cfl+N6JTDWBFhm0WfowYzXa4XJmZXBv6+ZLKqnY+8LhS8bHCXXWMLaPZzqogh5H45MvysTlNrMQL/WUGvcDHhaaGsad8XCfTY9DMTQRPqrqmrBj/6NR5Hk2i5sQFWy14lDb1F569sgh5nGBgkWlaYbyrm+By7F9XWpvroecOXr/W2bNQzFb+p4Q1cfl6GLYMuusaBJZdBxShhjO6ioudLr7o1SUvRQBdeqDQcGYx8U1QMS3y+hn3hcccoi9AV5r6k0d0W/iarPv6mGd9CthkKNo+j0Qco9Xcsy/q185P5jsVGvmBeXwlpkDxbLIY+Bydhqy8iwOuMo7HTeglpK5+lykzGfw/Fy5q6omgt5Bxm5+3dQmtHyiy5XjlsZ4063INTvr2uLLaxZgsuGJYgrYBh2TlmgB6vHAlm7Xo9i/9XjgdZkwX3CiOL/leOB1mDBRMMT6dX0sQaR5k1w/Mwr4JZiQzPbK5sMrBmY+Y1Ed5YsHldjXWiwkWtbyg4Esza5U7WQe2RcrD2qGPWgePaYhtH27jLA6xD2SMjc8M0bRfKG6aJp/rMyPCOI+Mza3TxDtn4i6c7Czg7WBW/7aEfWnaDuaGaeISRSY3uWMFdjJOuUuMJ2tvd3prfHMzshsB7NSwHu7ieeh75Sm7R74hmnADuaGPMObLu6Upu2G/lZywBt17qTavFHnTmnijTp3sp3b56YbdtN2MDcUyu1z0w27iTer3qmbEbmP97CIdE57HQzvrJar2qycHwlm7eSKNRGuWAhWZlJbdry550gwK/uHPHd/JjiVzUze0emRYNZ2Vlke9kgwa3cNsJzakWBWGHjnj04Es3ZczkK6l8Gs/m0JHNJam5s4V6bOwLxsiv4/

Horizontal superior, 'B' ou HS:

1011011111 1011011111 1011011111 10

$(a \wedge \neg b \wedge \neg c) \vee (a \wedge c \wedge \neg e) \vee (a \wedge \neg c \wedge d) \vee (\neg a \wedge c \wedge e) \vee (\neg a \wedge \neg c \wedge \neg e) \vee (b \wedge c \wedge \neg d) \vee (b \wedge \neg d \wedge e) \vee (\neg b \wedge d)$

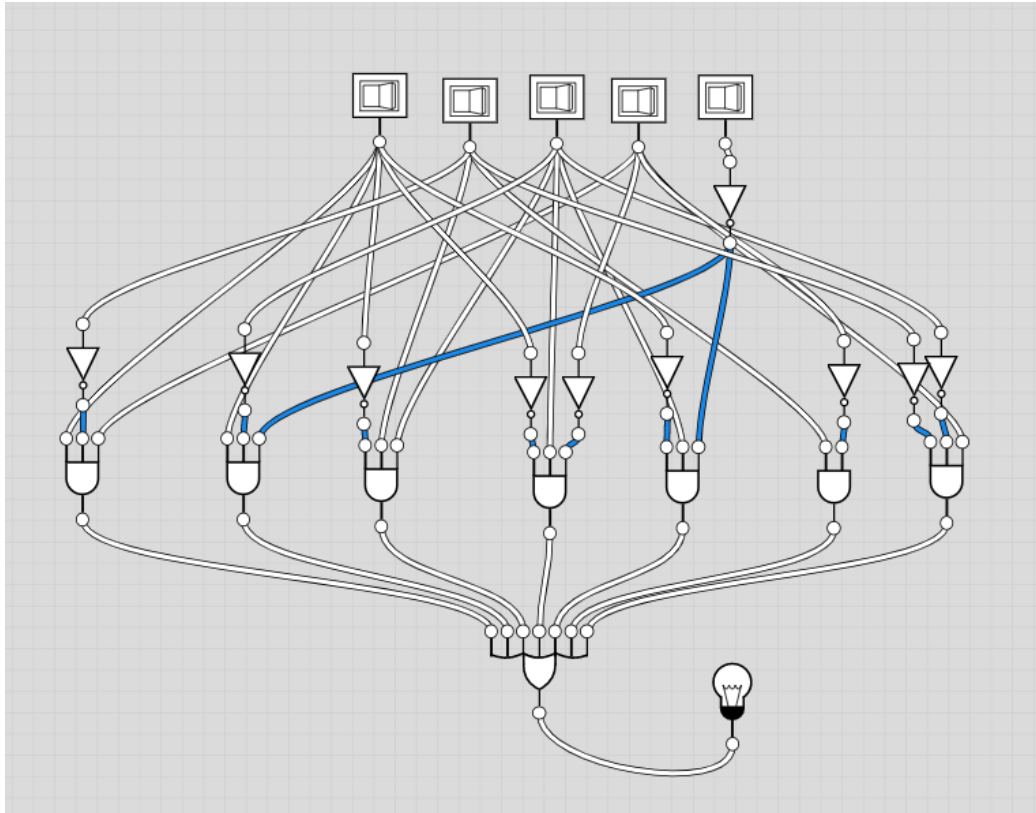


data:http://ns.logic.ly/1.6;base64,vZrbbhPJElZfBXHf4z5WdUdxtBd7kwfY61UfY69YiGKsjd9+a5iJDcEzKsRgBmxhoD/q8P9VLX/e7L495s3L6ue/m+3T/fphv//+6e5u+9Qd/tBtXu5UB+svn3fpn5r3q/3L93q/fvrvcZ8f/vj1kvXq+bHcr5N1VRnnBVqQwtrkRYxBigQXVYxQYzr1c/7tZV+vXq5Xytrc37s9nH/uNver4Ncr3bP++/Pe+JocfNU13escwPGliBYgTVFYZ1DEauOARc1Jr3EZPTXKdNpw8n606F44dbhsQ7C9bpKmJSWtiMRUTUSmhtioLiHcr8IAEHFNWeRCh/wul9LIJi0UdEtKIGg8JqKCJYCyLkWCkPlKVBxT05hVF3wTFRVm0xyCKrklYlbTwulmRcpbN6tlQ0wElqDcUdyVK3JazkBiHtkUvqEopO6o6QvBatBYDFWp2RHngUBA67Q4gVsPOOvn60L9jPG4HCsMhwKp0jtmKUCMIxbckfAUrjMI0uqnZRTtEwkPnRwJtO49uIQIwoUD0mZrDBmEBUXgTmnCZfoQiva3tQOBN/8UHAisXJKgIOChVC1caqUT2VAiakhKNUoFOyRmGLAC8xcC6DpYjaHQOGEP17ouwXjhlxFgtN65KrHkMuofvMXALxgDCTqlSBJVam3CBiupEjU1J7ikG6gaC4xKqV4rsRevxQiKDEFSN4jmFRE0asxQbBHNSUjOZOOnCoNUa7VsIqgWz4LBgyIIJ1YqjXoAmSDRTHxibLbg9NALSuEvWTCoo/vWjNME+oxgu9v/TtAs0HemHoQQMsUAswjKJyqLZrjNSsLI4EdDEtb7OD4/EkAxQFQoEyyvSCIognT4LUohqyc1KDnnMIYg2M4cy+EQDh2AV5QsGkmmGbAaIXUvTsIGQd5ZhMmutSSloiIZ5dF17tg9Rxf2WMydxWNg4rGAajy9ErlgZrKwzrHoFGDxCrbaKXav0djyMuWC02yBUE2I6JzIKgSUAQDZPPKGRaoSQTxn6VHb4Do05b98zMLqLxMrpaEx0vfa8eOZB40sRFzh5S1RK1N4N+SXivblwXp31sr8IhP1GUi6DxiOzdeqspLk4JKEkHE6Mz0Q2CHk5NfQyOPIVWcxUMUpGm6qiRaBiIsYdyFkooQmeVjUdOxrtXdzHnMGbRIm7JVIPkkmioBoheSOdl6IvufYjapJpmEzB6ndpYMHQ5Oy9pV4WSnnqbxex1CNGBUMXQ3A4OWxnKxjt5+9AgRk0eQ2NAl02x1RvhFfbis+Xpc8km5HEUuu/2N57U8AWH2f04WyA0quCpXrQtQDocKWF9ZHR1gKV663UYiyZ06nQafOmHedmpl1eYZPFnLjvHbw/7v9PzJp3pH7mTVT6I1Hy/TELqd4XUVZJ2i+prAaiD5Gg8OX1gQhKiE82T/dl5t93S4fTEOD7/9fVPrvAPb/i6LfVn/2nDlZq8nzdxH14/vl1Nw/A8cUmYmcjwTGgahjdIMWF4JjQNwxs6mWniedA0DG8GP4bR0zA8mZ2BYa0kTBieAy0JM5MmnjcvCTNTwDzJn4bhbc/MyPDccBqGt0gzl8PbRKZheLsUMzK8a7wFXimm3h3ezOix/I2zpp413vTMDxvY9vBdWm63Jtm0nRtZC6HmSlg3n3jTDexXP+D0nS5a98wTbwRhBkZ3g31kpG5YZp489AH6czlw9UNu+lYmBumiTcplCaeJMid1W5Mk28SZEJc61rXz5c3VD0LoeZ6aZrRY83tn5Qmi4fyGfSdG1keNvFB3XT5ZG5YZp428UHufblq8ocDGvVmlbhXfwdw+DMrs2aYZeGZk7PdZwtCSMm4bhjQBLwtgZGNaVxplwZhqG5y1LwszdXLFEa0mYubGTpRNLwsWPmOvDZkSPdZl/BnM3/nPWI/8B

Horizontal central, 'G' ou HC:

0011111011 0011111011 0011111011 00

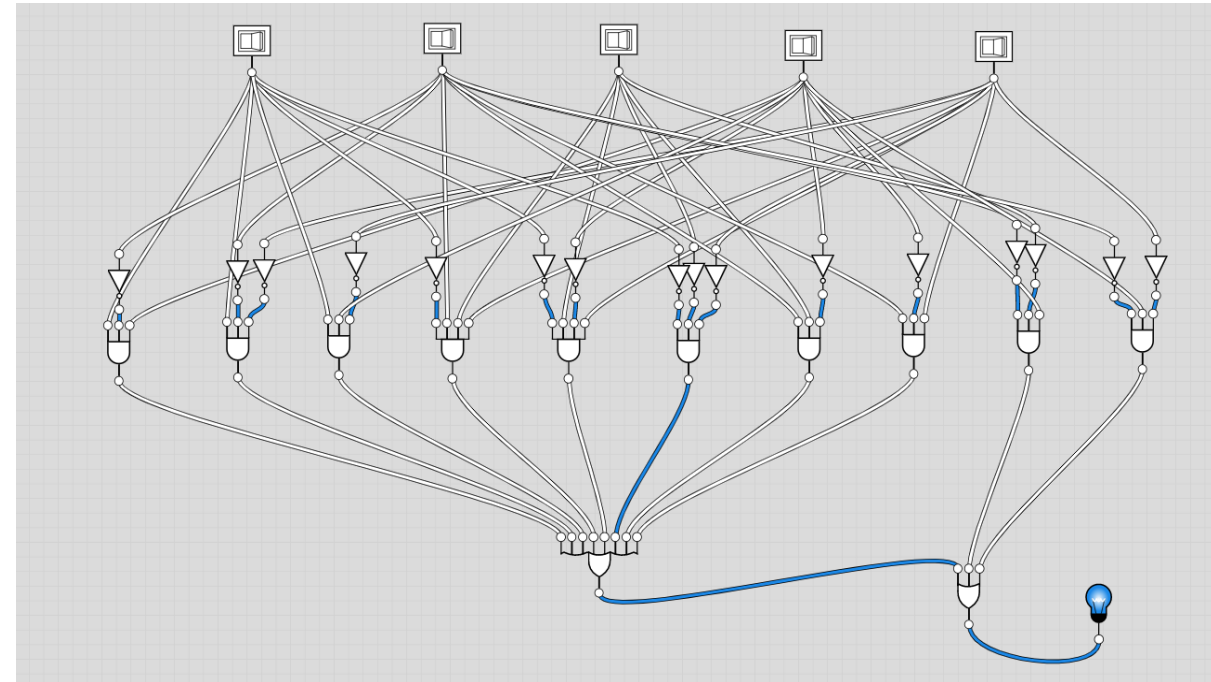
$(a \wedge \neg b \wedge d) \vee (a \wedge \neg c \wedge \neg e) \vee (\neg a \wedge b \wedge c)$
 $\vee (\neg a \wedge c \wedge \neg d) \vee (\neg a \wedge c \wedge \neg e) \vee (b \wedge \neg d)$
 $\vee (\neg b \wedge \neg c \wedge d)$



Horizontal inferior, 'E' ou HI:

1011011011 1011011011 1011011011 10

$(a \wedge \neg b \wedge d) \vee (a \wedge \neg b \wedge \neg e) \vee (a \wedge d \wedge \neg e)$
 $\vee (\neg a \wedge b \wedge c \wedge e) \vee (\neg a \wedge c \wedge \neg d \wedge e) \vee (\neg a$
 $\wedge \neg c \wedge \neg e) \vee (b \wedge c \wedge \neg d) \vee (b \wedge \neg d \wedge e) \vee$
 $(\neg b \wedge \neg c \wedge d) \vee (\neg b \wedge d \wedge \neg e)$



HC:

data:http://ns.logic.ly/1.6;base64,vZpLbxvJFYX/isC1q13vRzAaDJAEiDfZJdugnmMFMmlYVC Ln1+c0u2U3SbHn Ut2SFoJgS+TH+zj3VN3+5X73+12+/37z9OV++3C7+bzff/3Tx4/bh+7wH93994+is5 tff8mPD/vdl5t
t/FJvN//89NfNzf77V/zlg1ZB8cRc04bpFjyLSlimchZGJ9WEEvhz6tvs0r9r3o+v/fDfu33+/Nvzr2xu Hu/K7aZIUazyleGtG9My Neabta wGn2OJOqVsNjdP txupZMenX2 pz8/12o0QXpl9uc/Ntt4/7u932dhP45qY+fd192/
/98En/t7nZPe6/Pu4B3eL9Q918pEFy70Ufh+YqZ9oWxVLhgTkRtb DWSuncAd IYe4wzMKpOOv3zH+UcY3oIY86J18Qjsykkpn0fSB8kC16ZGF3mrskDo3a mU+eMttN TRjHH+J9XMIapa006Mm24Ylppy5BhzXS uWb
Sct23pwKiC78yyOOZXMQaal28c1Z8sCjKYwny2gvmiEeHkUvZDHFEP L+X6m jg+/SFj3JZTQJ9C86VFJq UJ6Bgf mOeysJK rNkZF Gbl4APp wlgC/AkrV+UkbnQH e bQc gRYHh1SVlikc r2IKMWs sBE7pnlRmZvj dYmH2CsDR
2fxMX/gEE9/oTRi2iK1yaG2phUXCI0QrBQdGCMFs7xLjFzPoiJtp2d0DyHxqpAftKc1ddVMDWbzJMETAoVoQkoJF0c4054G5P3ypehKLw4Cs1zsbvOT0OzCEY0G6sJmaH9kKcGjmCbZS mXVJxRuoShqo1Wn TvPk5Zd
+MnSa+/LMPoMZrvbn8I0x2MVtbJcDNLEi2QxucCCTEr7Kmwldah gfawC+gAjuTuaBOJM859pBIUGICajMlg LFalkHYrGZo88Vc5RQ6ppO4QGE+hclA6INM2TWURjR Ay+YR6KCBcd0Oc+J4zHlkLxohkbbh1FjNepjIBD
6mOBidkgE0ZccFWoD+oJBliCFmMgQ6SY1prVDgJ7 rVnZyIJCO2Ma0jBhuHS +BqZAAQ58rSwZdXbzMAQMJaC7GYn1pTAihjjtn WTykyzA+P rMUBeIR4QACKsO0zk2pgL4KQ30o4ztxXh8SwjelsXSa3bez6aQhUhySb
3k/nXzgL AoYpxa18cmqauQIE+yx3g69Y/TFZjFnb39/9/vn/b/S4306Mxu6ylCrZhyOB4YooUaK4ownHo TMtuk6Gi LhuzDp1aFeDBpbX5pBJzOyd6cAy7vtFmT4jXfK/uPTX6hKMvzBp22pT4eXP3ziw9/TZunh98c/F5
dhaC7xMgyt7Kcw/DIMrWovw9Cm1xRGXoahScqaMDNpoin sZRIA5SF GhjZ8ZmqGN NepBUw6i63ZTTORofnwy zA0aSB2E60114zM DMzSNNFcMjFNtGPn mjAzBb w0TdfDvG GaaApOhKGdvc/D0BT8n brp+nEwA
7NUZ65XYPV2aboe5g1bmzbb3qlmro/MG9YmZyIQYZ am6Xo/84YKfD3MTGRoU/cyDO2wNYXRM+OANFvWhJnTGZKCrgkz54FJrbkmzEwB0wpwTZg5nSG92GUY2jH9DOZ+lw9n8Zv+qP+tvwQ4HPsfiC94+gr3tf
14AYop+kA5SH+gyN4Hyjz9QFJzfKiP49oLPw37M9I+wReRRE6BZaHR95E7FlufSFeU5twaZYfLTVN5+XkymZccIXOTy9kz+5vXrflYDY4DVXMaHd8cFVZLL0UBG2 FM9XrNI4zGn5 ENVw5atGJ6Y312U35q6AkWizk6li
TDolA3JAtw1mp1kARsuRFD7dKNnRyGqoBSndqCnV2Y/46KB9Ss1BrgeJGHUXOvABj0apklyMXLg57tSA7PY2UfsNQqb/huVH4tVRUdxNoRAMqtERtRWQTzXBJC6v+ Uv4WhuryzWCwlbvsl+MS33QSDm1ZYfp4c
4WXWNW4hrRCdk4e7Tf6m0EEUV7 aepzcDP7tz6TNh w9Ba4xcKwqKCocmFjBqmSrRWTBa64b LZC2Pb2/HNYwzR+syuWxDFarzIYblam4QHfQ1S9JG5qKp0AfjqxvUAP Om8+dLIX+8L1u2oUI2 MGCKYK1K3ht0A5
01t huUgVM y6NZ3jKE2ys2YkEEeX68sldJIZI04xVVtvii rKWF eBM rYtleoiN0PHe6NeCofzR5uPZTCuGV2Ld UyWjGM2Jg2 LGDVMNpgC3 WJ1bthliqCOF3bmR6 mslxy YMutq30Nc eNRt01Do3kk3xYN3Iaschro9XR8+J2
rNZWZUvf+RIhkMBqbxxxXfFgFSQOUmXnRm7yPWb wvMu8sdL54swkrKRQZ/oUJNksByIT/RoaetgElCzTjkHmx3HHZk53TMc9kO2s1OYZRu7GjLOc8kyV2AWdSyGRZzhUM049koueGt83Gaq542dOyFYBJAV5pGE
ncoeaguTl1hymAUS9QKfPkQs41MVXHTjCtUbWheTABJsm+mNjYehA4CMAJzD0Qa+J7je7gyiZvhJcYw04aRxliVE4QghDBxX1TbDGeKbr/ASPBpvtMc3M9aqeXGh6zWtcUgwmislVbZ49wKfy7CpseInEaJ0OPTDg
A6iotzxUznjZtUSu5i2XC7JVgFJwxhDZBLGYEjwD4pbJ eBIYkuDpnjx4irTS5rckmC4KQ2OQDCjHE5aKXmWYs4Mp6+ABgFqalPAn e5V1ctVsyw00fmoK0olqt4Gp8RCwxmHO1VD4c2750e8vH0pNOjo9fJUZPDCysJ4
gtfQClMgSZxiKlfbGDRY7XfaTwdzc1w1z/qijnbOV9CCr5yzyLjyyEtRQSAy8C1JwZ8njkdYssXZYXx0pJcXfZan/pm4qQhfhHEzB2HSc eXyQZg2MYinclpLz1xFk hwp eWdHKZXLMDTTQ4ShTaKZ/QvJkFIXiKSpNFMzJAN
GrBma2K0JMxMZmtZdhqEdYogwtGk9s5kiOVNimmgDciZNPMMMTI0Y7kmzExr066x1hS9uc0U6fZjjikBac+aUO6tVo zMnMrj4Vpun4czKSJdnU2Mw5Is406tUk WYm3 lvGGaalOWukBc2E3XT+0ZnVlcM1fDvG
GaaFOfqjMLI3O9hZhL00LRo5mzd0oTzQ+9U5quN1czaVoqetfDvKHOXO/03rCAabb1nSJDS63vVMA02/pO5up6DzxnlUgvNjO1STcpRBhaN6wJM9N NNJ1YE2bGdtlG3ZowM0/a0PzImjAzz yDRPOyaMGbmsr/Su
WdNGDujwKQXmxmUpLX3WWv/eKtk/w==

HI:

data:http://ns.logic.ly/1.6;base64,7V1LjyNHcv4rgz5PlvL9MfAGDzKwuvhi2FcjnysZoxlBatmSf72/ZFV3k00yN2qqyNWBWqX2sDNDfh3PLyljMv/y6cvffsyf/vjw+0+fPv/67dMPz88//9M333z+dTr8xvTpj2/EZJ/++S/5t1+fv/z04XP8qX779Nd/f/rw/MfP+BVXzogULQvRZaa9a
vhV48zLGqSPWhfV/zb1W76k/675efnsX//3x+f8w7+8/JGnD7/9WL59StpUoYxnTlvOtE6exRg4a9Zmk0VMNsanD79/+6SSf/rwx7dPwj19+OXLc3z+8cvnb58Cf/pQf//5yy/P/3b4UF7v6cOX355//uOZsFr89gt9+oYEI7jYkg2auzoi08Y4FquMzLrWGVfcJ SUPMlXukzAwkZMlx/+
YIbDfvxKYN9pqlyuLSuimsyssOimYIKoIW7yxMR+AWWWMh8w6MCX8CzXs7Qpa/Epl2PjrnNktBOaalLSxoDdPJsUSnmkiVH5A5r16RyTXI/ucrKZnli/QusCJrYlokCQNumqWcedOyNOfSAVKbQb8jMGmTp7yKLn8uZwJRxukXPYPBQpagGjLxkrCUAm88GoA+whA2TNAcdcmVOJG
/76j3yP6sfpMwpFQeCqkDlmzUKNUJlviVfQNVmQk49tVzSbqWTDen5BIPXkndkljGVWh2OgzHEshPq1zzKvQmMn4n1C417UdEHjVf/AZgeY7lqglGFuqZKYqsOnsYRcSSolKilBvydnOWrD2TQbaTHY/BA3fY5WCN/jCtFKlrEXINTemMpdYWJfNZBn5HgXARU4olb6XWxnT
QHJYo48rWJNMsmqLHYJeiKUVosgW83BIWwOENrHkBBAA+GoorBluk1GZmzChfen0myWKHbVgXHEpc8FEkwa+YBTdHe1dJDrY2IKRsy8I4V6ihHJYUo/OeB2BPEFpw+cvzveRNW/2m8HEEbaPtJ08YNwieYRVN5e2qFtgsCPEC+qdk1j7//KgBBRAsUD0isMrz6M0lme8eKs
DknMOiwiCntRxdJzFIYOIGSUJDUfCDa4qxmUPTkIHhrxbmMqmtcS5gJEs4dFM5jjzLmD8pl/BmE1ojK1OKWsZ+ECFeTgJ89CGGdvglLHytrd6S+hUch0+4km6elsb4pFY6CoEhwLyoLTIGMgaoVSVFj8lU/uAhhx6rpnUW0VGs+jqTXh67nv0SMHBE+QN+T9krKrtNo1xy9uL5
mNUurUj/UomOMgnoKzVgrhox2uvJeriBLMlyaBiNcqaOaCH0xy/CEeeRla1CYyDkaZq4EjgtS8FOFkooTCZRVbeOeOlec0u6hyM2tWlW1VOWTbXJ2C3SC8l4h1pco71NPM2s1mp5EY3dUTQ5e6/hy0wId/82UaK2MJaIoIACWONamc3GG3570TgXJXIMaEBJNHHVK+aF68
EX5ulzySrkHqQli/7TImx4hWi+/HJWfEgngoe9SF0s4nCnEwrpkZDXWleq12ExmjCIUzb4R2f+fBJHWeEqFn+G5dOPf/vh+b/Sb5/SWfxDdtLCB5aa73UZh7sLB6/iKESqr8XaOocceU6+fcbkEiHOyt47A0zSE1jyl8+faQZ/YKHT//H9d9REMP+F7z+X+vvh0w8/5OHv03j24cvf11cB
OPLKXuC4dfBOJLsdTA01kUEQ0tK18HQSchRTbScdB0MjZMfg5HXwdCD7gAMqUQhgqFlpD3BDNREy9V7ghkYMCOfXADq6aJkqFlx+tgaiU1UTK0yuQ6GFptRZQMRUO4ZwQeeBoTzgleqTcRlQITrTd4HQWtt5HTwTY1rc9NAzVtlcx6MAMDprUjB95Eyp3UtP6rH1DNdEo
CFEYtH72npK5ozpofOhOcWY9ubqhN60Hc0M10ZjindREY4rUUmWjmmhMkQhma9ZeT65uGPTWgxL409agR6Otd1LTeki+UNNNWydCqjzt503rJ3FBNTOriTlI7fakyAKMqNa6DoTUCj8G4Qa1N4rB7grGDnh6JHO0JxlwHQ6MAe4LARAZCklsaeYNR1MLTcsieYUeeKFLT2BD0i
naQ4sSeYUQQmfddg6JGa+2dgPn3Jh27+h35A8Es/OjgcFvxK/MD3n/CpttcPoBdpj5TE85FCrz5SsZHUtLFD/XNMg6HX81jde/G6/7zX1/n6yAP7m2JLMCfmHb9J0j4BhSs0johSrF83/k6sA00c/HMmwBSHyJ0ZExhnjeXfIMYxHyqrqyaBOIfxx+3kvvVJ1PhjuvJZVbCBfELYbvioG
2B5W7zSA5SOFYqk2wFDh0GmJKfhIH6WeAAGqltdqqfT1mAicHAAjOMEGHRR8izk8mA08PUcmjUGlUuB83imPIAqLSRWbCMuAooATC15yTIugjF6BmknpywxSbAhl67PhyJ5k8jmq/2AJClW13WWuY9/FAAeCGqno4xYzSq8l7B4EFpQN+pvWS/NrpPji/LqQZoyEDySwg
BjhHuaU2BSvswYlIotHqS5opIDoUpAvrIX7VmF4LHKQNUZSK3xgBix66xACK8IQFwuR/jscGbOpPfHwLmD4+51Zk8jeQptldFeXB1vOJ9TujTeOMAGQ7PMLx7GyJfIEkwaPzqYls7Dwx+xlkLMF3AY8LHDFw9mDXz6+1kspbC3PCNomilMpvDtpeaheOQXZOHPeMxAHElOfv86
w8ZSbObf4jjYzC2Vv00uuJa8dhctaFSOLU/IKilqkHWNpnCkXGNaSA4EOgEidYqOI1LNlIzgeBoZcmHjQ0qOK9Nwnu82RQ3qTnvcVM5MZuUxSv+LkYqL1oG/IYODBgWfirZEcIpdL/FoySE6YDD896CD2ZRIJ0EgEctHcFE+KHs2T7xKOkpYB3P1DhKYoaU3aPdGZFbOIdFYoppFTF03J
SRpuHPsFRCTVlhw5QJ4ECMRsiY/OjNEeg7VozdGHE5BXMpAYFJoghoEiorSHXkLONUbhlKewJIOVIG+Slc123Nw8NCVW2jQwHr+Y15wogcLpKBSyu1MRTEQSYyYp4+BJAJQvAIrQ6qccvgG3KlgZU7D+mGvZagrVHVpoApRxsO5MNRIGVUIDNMlLBRRUq/hbQnYLeNd
CulYUCDSyke6i1dZNYqFQ2bwfsXXCOFSChgYKpVomxB6gUEb1CKlSoIPlthYEKlP5TocJ42AjsFYJ2sIfEiJVhX0H77TKXHFpzzPdKstUsZ82EYo8IARFRABYI/4b5ZeIwJKXOBK38WPeAgJpsgFkvc0CFysYVHlxLMwqzSjrJ6Wokgg5SaeHupZhv2cGtvE65jy2I2ZCNM8heqOCUgG
zOC5hu3lYH6BAkyykX00jKcKBNmTlIC5SOSSH6PsJHqhJrsrUwCtkMOCqeMykve+xDrHIL8yp5wajf4HY4ffUdU3RpsOcdlEsA97BEQBVagzxRzLReVinXdtZa4ZXysEFGhc75jwmSFBZ5PTMCavHETm1Lbs4F0MCt4Du9WwG1UNwnEvs2HWTQrv4WvzegIPk5UwHC9QIxge
FiOWiENSu07aANNvOlOpLQplmHEJgehmUMoaxJ+aQH8cvM2ktAzi8Qki7J7ckyZ/SZtIDUEHoLYlgpZvDH8qRg4nYrqCSGjRDPH2AsPgScO0GxS1olENc4RbEDbVJbLFDJs2CB7g5xb7rSkN6GxqA1NgAu5cGiaBcmCQaJKylcpoeY2C6VwyoFD+4CcEaxl4ItsZOOsqmEp4FBWX
1UUyWqMryoqcKwMAYHfWEilojAsQmufONLVksTBP6CdFe5IYMgLgId6SSnbuSDqLrPNaioqHa9MA9nq9a60vWlWOROIzI231EpbTNj1QGx6ArASlyddqHKHihNjPSPnQgKb9au1JZhlgg33syrwS0Q+INAZhLm4BKd3c6xBoodDOSs6Bw5KLvWPnAKE4pAqQIDCxpTPlmqU5
APjccsXALROBnkdvUoeEjwM4jKx4aQmdfDKkEYot1ZhditiBxBrcQyOLrZDhC1aTPUoaY5DAQqz5QhkfCvSggFckivEVlYptF06P+5COA9TCSzAJHWZiN08EamD+OeDYqYKNOIRjU/qfN5AvNT4IBZRA4ZhKfN54oBBLxHPTsikiU9BNzDkghQcxvOicof1nPdiCmzrVEhmguDM
M2j29s8SKKDOLzYcVEbO9MnmMNBtsCjA7Y/6CIFY6t+ACU9CZdAw6l7BXMWYarY1ompLoNaQdE1vwdMjQj/CjLEWqj0l3ZqHvHg1AJBnQ97BAO25tL9TqmOMXBdGBXy7mXHwV53qHnLADQ5SSUjqscFJqz+2ZBDM8V6eQvsESBonDksHQCPrvYGH1KhEMrZG6DoZFWFhl79L
Q6ak8woxFgUhwW1J5iBmmjEfDDZRCozqPPIJGK+J5iBmmgN6T1deyAZWud5TzADydCat6NpOErQJBrwVsmsTwcDNdGa74NhlIKipaaDjWqjIdo7qWl9bhqlg42SWQ9m4E20Y4aBa5Oy/r3URMr6dzJgWtYngtmqJlrWp2btjblpfalcZJstRkaBaEOoG2MwdQKkcDXq+mG9o
MrTV4pzhd6wzeSU20xuCdDijJWtqbmJtJptieYkZPlOt8TzliQk7xhTzCDCLZtZUmNOKprkz5sMKIN6lHeKtEtB3PD6mA9mBsWcevBjDZcSV3ywZwg6TSECIz2FrAnmIHNO15W9gQ28KatBrwezA3rJprN3YlcOdR8J3JfCwAiGfcqGFSUpHMZKrkGeCeYEZ9YBIF2BPMIM7QSo09
wQziDK3DvSeYwd4DrdmZJ5jBRgitW7knnNGuDKmpvCeYwRYR7cMGUyIO1noZWzarBpQPHK0aUHLKROqs/OjJ2x8phPAjKelSvG2+e101cLHp6mVjQsO1dAGACKBtqKl1rlbEqiGrnVQNTi6+x9PWugzi6BaOQdK6mElXzzi7D51JMsGRunHX4/6xcrmdlbulCB2ud9aZPQN1i2a
BPlh7cKUL1YFX4VwUkLOXlfegsZj/PaoQy+xlHEOUozOWP1K8qyV11E+ixUDzQS+lrNElAhdrykx+3ggU9RihbFCa7rck+qmPwllHSXciXfHixtFO8ofrHAIROp+ClEqugh28UwCbVcveh2jxpYHwF5CWUjLHKM/m2vYz3QgZotR0sQJcjdcs6RADYvQDKFtWnPtJ3AOY5OcoQ5vUW
ZR/f3PjOr28YV1ESLMCKO2j9GxjwjiAHKyxZjTchmdChASRpKXhYmY1ZJTbGEURFB06AqUdK1+WsT33GdC1aCgK25kvNcESR31FKTuvuvgJl36TJWqH1AJFRAGn5PmhK4L47vlJqYhN8WDoiYduw94SUUxswxh3lWm6sTMMV64WgH5H4PZdmu0hL1QgHn8G5vUw91oj
qlumKocK1KPoyl9+jRrwU9R2PsKZptl+iWIGRAYOccJY4GZWDTdaJXA80U2JR6HTuDDYsjNGKZG5eT3w1NHei44p4V23fbwFSQLYNkrkbtSotBjv7qZuLcNNrvqShTTPQC9FloiyrAtQZFFSTYVjlsGAnd5WVA0JygVWwY+R1FA4FYXpNm3DWlJjrURzAQhM0oQFkKz3EZ8Dfmog
27yR6DuToSLPh2KrmocAy9zzNC+JhEQg+2ippl8qmhJclMX4ab54Gm+vTiQRBkEu1omrYSTKdnXCFqliHgujCBae27nBL7HvHGmY43c8oiiKNILUZVdc0Bcc1mDkXBo6IGD1MahM8bjylhv4Q+PYFEvpMn6t1Ja/UWiLfdtCuGdfmw3iEgLTUwVBDjWTAAGvOFWjJLAsfwyR7B
mdHAAOewxm03XRZtUqtO9I29SP2BNLBVBv3LlOlLnVq1hBunl6B5L5tt18l6xbOPPLCWZOnhBgxbJtQp1juuWt/EMGFsnDzR1LlOyt+h2SabKGwDya9QjUYGR4BBFYCcKWDEwtVcwDKW2Qg1+XPDUciZx3C2TfgryatwKF907cetAeVIAgoli7KMZukl77noX1x5e6vhoLBqxiL8O
d5PmJzSzlNus1q8iiaXBPyZzWZajdXVLXxiuZiswfs7KqhCNSt30lHmnwcjm9dFiosi3E+XPlpOTcYdxnN8X2XH2S0yt7GgxOhDq2CtZ6/+gU60S6rGKBa+oJo7CmabWbjZe+6oQoSaA2KtoghTHfii5VtO7xki9c69RxsEsoPrHhFBh4fHCce5lICR1LjyOUw4Q9ivOCn8+tyFq5XYwG
eeEO582SQfU4BUgXgOeD4xcualZZ+CZQM0YJ79bqN0mO/jaOqz7phAw2V7bBeWtPqNyzLwLhxxk3rDQgM9YV4fy9leTPrvlgGTElXl5kH1+nNvMNAhRWav6GQXSn3JorXwXphgZ2I5yA7p/tN74FYxsNsZPY9+C8gf7UHUJ/IAmezOaRiqOqGFGKGGoAh8TlQFqg28KPMaOSAl
BAGHu5SqUE9JCXlgOGHk1RpEMHQEvdATaT6lGgzNFKZJ5iBZGjszcCMQ5QWVicc6tDphcBRIarEQ1USrEvYEMzqxJTWB90wHou9a5x3J0WweJbdR5x9lD09JXNDNa1PlDdUEy3rEyVDOxY/Uzi3VBOT8U5dmldowOBDB982Kam9WBuqCYaUySCoR13DlIeISlSWXw07fVMbx
RnNkpmPZiBAW/1JhqHvpOa1nPgG6ppPZgbxhladXEnNaOn5DdMB7RS05Zm1bq3EIntFlNTrRzfd10Q29aD+aGiXJ93XRDb6JVpHfyJlpFsm0WkRrB18HQmrFUrE3S+Z5gRpOrUkU4oBAkyRIHBW1z55gBiOUNHa/J5jBcCmtdbonmNHLYLnD7glmtPhAYmp7ghlEYBo52hP
MqF1PSnr7ghk++EAJWqO6iXJUtWlGmPSBwxlgQsH+KZIFP1L9ykKlxfIoKGeDvX2aAtW82xxiZ7PeW66lSCynw/g5EMzokCcN6zAAPZoCFMPZKCPgFJZ/Cn2cGODhzeQb43aTYP3YE2EPH4fz2c3UHH6s8zAuw1DeW2EeAdTVd8JN5f6iivZzGU+VD/e8LkgxMokhoqCsf7so
Trl0jUqlhVSxhTvQ9hmUPvq8ylns4lgg8eTD+b6xaWWNEjkrA3SmtuXHFV4P4/RUCM8F29uUUIy192K2iE5KdjOhsnOEp2VnJomqPWClAug6c9NIEgmfpJrVS6mr0/9U7zN7uw2udi3uZxDqA+h3/aRsOdhliKrtUgXUeBSZbmd07kLwUJlczqzvW2QCM7PneaBGR76PIV1CLBG
MonyT3un+/LQbM2CXRI574qzVKTOMAXQixPvFmYyDDYX4WmYkYUjmJAX9FYWyyiPRA432y5CVlF+rqEuz/gfWQqhZ38vpDm4/r6NdnyZ3oHiWtkel4fV5WRRtUHEVxXy3flkxBrECstw4eHq46qnyY6AKaUBpH8RIzAiGERwYVcWiOUg8XPDijVvuCuYcyrJ28hnNchG67B
dlc/2K5yK8QwmNucmWbPsbqV1106oC1to3B6u6qJg1ZrWMPKPDHDY7XlJfECzypYzRXbwslock+v4ggs1gKvLr1nmBFQeBSl7IEZLNvW2XK6NPcGO0LdHlslkx9HvP1D8L4/FHzAUQAiO0A5Spl+glUfGC6A4j7HGutH29W5s+hJQZ7+ygRw/VX2g2kLB7GXCvjzJhT7J5GW
1+51GEIKa3eSDC1036kvT8uxd+rL01LsvaZ2SEmN2PvYqqb1YEbdZ5rXgdDSylUoEdyzcHxCSmLEptlW21mwVRGNrN16GG1ZG54tL8ezMi1SRFOcJpO4mREMLQIOjg+ldFV6twkKYLuceaWExir1TSQDC2C7kmuBscntDixl5jB8QnNNfcEMzo+lel8TzCjqR2SN+wJrJwOCb76
kOCvr2c3YRUf7cdGoPcy7+GkwCZOEO6Od8PUNO/6ixAKTnZozOCuW/G+qOD/k90RqCMOelhvaDkpdzdwN14c9czAsGDvNDbAkzXJzokQCI/dAOHF0KO/T48b7n9iYHyl24zKtZQRXmXQwKhjT4/JTdm8GIW6ZTgr5RDgphakFzBQVKvVx/G9uC2kVYmYpBgHQuhWRaTfqo
pWleDwnkXCPqHORsqvp+1oG0jTzqQ9CVQUcXCIht9oqx7sJSx+K8wFNEb65Sr+hudpDpN2lG0iaYkmwKdVvKW+dKoD/gos3LiOizPLSZN+Ol0dgXjrhcIIdu8mN5nrmS/yV6JzI56tHDgdYp7b1FSWrecnyBamEm49w1WEfjkjxus2y6iQGqXBTvry5+tp9WdWU19zbgUQWR
Da1kOlwJ04oQl194fh606hGaFe+3XmqcthjHkOUVrrrFlfFA7SyrNgYbmrIdzWlJ9Mcu1RGcl2eo+zCR3C2GU4qqNgK2Q2czXB+wCixlMdgPHmSqJATNJbADAXz8fWde+Hed+U1omuzjhRi2U1rpY28SOrX2LQKyWZfK5flopEC1eclwvN1TtVaD/ciyyMGdy735kw4J1laGwB
Qbkwhq+qwrh/ATNI22cqaY2ouHzyMo56kgHpmAllo4hQoSWgoujFYbmlRdJzWlJ9qNjPqYV227+Aae3FwFnaq9QonKsGpN4P83HBfm1wOuy6Snn4E6AJ7i3gNDJ/4/gLm/CM6C2B0080YIVhp/TJY45GzgrFMRBDaqKoxUrOe6ohzRQMOYHOgJcX7ujQWVEXPCMC97ZHFHl
H+W0PiQZbNkBL7yzn3Vx+QbGvd40g4T6qqZtz166wlvyMJiikWp/r6qO7CrwbEdBJk2VODnGp/qK8Ozte8nzS3L2lqatWQ0JjrG9e1dBztu2TQv2BEZQUcO4NhQCzi33l+BPTursIND5yR+hua4p2mQoPPfnrPziY6X1OGw2DpDKLTfRdPxP73NkCNUffw5ls+r1W+6FpyIoVjsxy6
yOT3nsQnZZZE87jd/rLGksOV85ekl3xEz9Gs01RoTYlgmJG9WCDCMO8donV2g/tczA5v9xQJ1+SPVeUmex+ZgP+m42ElymBlKCLB70RUBcsQbyE4K4sz3rj1+HkjhR+Sta5SR6LzuNlktUhP6nKvNGlR+tgBjPgDhK9eq+0DHqmFqhHLhNEHuKbtXbXayq3q2RAKHDFY562pgSil
xK5creNTpWfbi3v7Evbdh0QmxNsTERDu8fwUe0RtWxKXkRyCaGhcXjEuhZp9BWNNEaC9sJQMraqDIRATBNCZ1YRliCXepBt27SLMIRu7oTsgqo3SopMy8lCQ5sDjLMOip5wbq5Jy6EZlJ5rsNi9NS5636Ft29QU+r8zPYJ3sqBlQxY33i2kxryLMhfzwUYdgVlUjUyMr/apaXZE0k0Aozl7
kzAsKvqVQQL7CQT1080N2Z55by/m4qJ0R+n32ftriXKqHjVdLeaupw8N4rKqA2+uFQHiS3tonDiSHAoeF7tab0hD1e4U06tTAdQsLrfewarRobstTrzw8ktowppkCWEEn1Rujw74KRUYse1L+7A7DYXQlDNISSRV68RDFbrdHkyXkcokKHiSe+OJZNtJDSnD66SC6SUO8R
CFlgUEYqGkEocIhpYb9wQz5sBkaa7gOhtbelcZYG3paE8xo4DnLPCEM3pbkCTBB+m1uLQkgqGj9nF80BPBNDRDXRCBDOI4l4igLTKtp9wZkqAzt3B9MgAJB6pUgYb9U3C9gQz2gIkUYABujwGRs2Mwg51a7DXTDe8Xgp5bWdN0TzFGaohYK8gQzGaQhZd09wvboCm37AlmVB2
QcsueYebpgJRB9gQz4sCkcl5nETfiM6QluieYW15Nt7qlu+HVdLSEK94e4+slc0M1ra8ob6gmWq1OndLeehcRqVa/lzeRavV7DbmUwLxvvec3YajA3vBhpPzhb2gypPUK1mY1qWt8SueUVmKT+zp3izHrJ3NBmaM2m09kMrdlOjzXRmk139isDrmOU3jMDrwdxwaYfWhrutmta3
OW4Y9Gg9wt503rJ3FJNpAbnYlerUF5r1JlDbfzhmqiNSjvJn13c5b0k5SH5l6KEEaOhhUlKQG5b3K29VgbbhiB14O55c3rJDWfqemx6EVd9HqsYD1WsB4rWl8VrMcK1mMF67GC9VjBeqxpPVawHitYjxWsVWgeK1iPFazHctZjBeuxgVYwXqsYD1WsMhnLY8VrMcK1mMF67
GC9VjBeqxpPVawHitYjxWsxwrYwYXrsYL1WMF6rGA9rAeK1iPFazHctYrmMcK1mMF67GC9VjBeqxpPVawHitYjxWsxwrYwYXrsYL1WMF6rGD9GVewXneN/jh8=

Agora que temos cada um dos displays, só precisamos montá-los em ordem. Pegamos cada circuito e criamos uma caixinha personalizada. Assim todas as portas lógicas são transformadas em uma só, que recebe sinais invés de interruptores e envia sinais ao invés de lâmpadas. É importante perceber que como fizemos só as unidades, temos que pensar em algum meio de representar as dezenas (que estão presentes a partir de 10 e até 18), podemos ver assim, que temos os valores 01010 e 10010. Como bolamos essa lógica? Podemos perceber que é fácil identificar se o número está entre 16 e 18, esses obviamente terão a dezena. Mas os valores entre 10 e 15, como percebemos isso? Vemos que o 10 é apenas o terceiro valor a ter o algarismo de milhar, se pudéssemos ignorar o 8 e o 9.. se o número tem a dezena de milhar é maior que 9, se o número tem o milhar é entre 8 e o máximo. Como fazemos para ignorar esse 8 e 9? É possível depois de perceber que o 8 só tem zeros e o 9 apenas um 1 (unidade), vamos seguir a lógica:

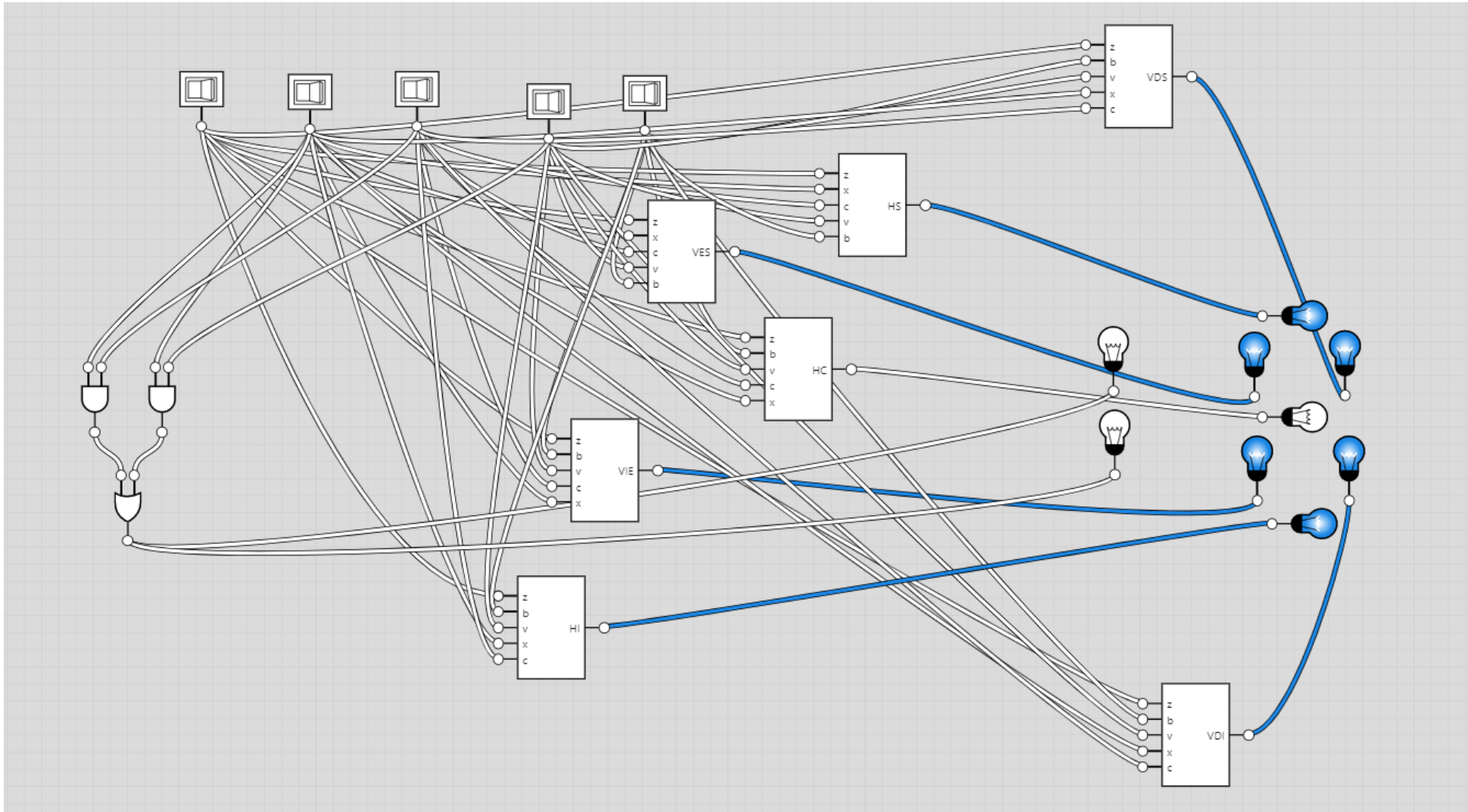
BINARIOS	DECIMAL	PALAVRA
00000	0	
00001	1	A
00010	2	B
00011	3	C
00100	4	D
00101	5	E
00110	6	F
00111	7	G
01000	8	H
01001	9	I
01010	10	J
01011	11	K
01100	12	L
01101	13	M
01110	14	N
01111	15	O

BINARIOS	DECIMAL	PALAVRA
10000	16	P
10001	17	Q
10010	18	R
10011	19	S
10100	20	T
10101	21	U
10110	22	V
10111	23	W
11000	24	X
11001	25	Y
11010	26	Z
11011	27	?
11100	28	!
11101	29	.
11110	30	,
11111	31	;

Eu estou pensando em um número binário. A maior casa dele é a milhar, que é 1. Se ele não tiver a casa das centenas como 1, a das dezenas será ALTA, e vice-versa. Perceba, que esse número obrigatoriamente está entre 10 e 15, pois o 8 e o 9 não podem ter nem a dezena nem a centena como 1, e meu número precisa ter ou um ou o outro.

00111	7	G
01000	8	H
01001	9	I
01010	10	J
01011	11	K
01100	12	L
01101	13	M
01110	14	N
01111	15	O

Isso foi simples né? Agora está tudo completo, vamos ver o resultado: (o sistema das dezenas não está perfeito, pois nesse caso não há limites para entrada, é possível inserir um número maior que 18)



Para o decoder, é simples a lógica que devemos usar. Mas antes vamos definir algumas coisas, primeiro: O usuário deve desenhar o dígito nos displays e isso deve ser traduzido para um número binário de 4 dígitos, entretanto, faremos com que desenhos que não existem darão o valor 0000. Segundo: apesar de um pouco grande e bagunçado, faremos usando a mesma ideia do encoder. Vamos lá!

Como vai funcionar? Bom, a resposta é óbvia, com uma tabela verdade gigante. Deixe-me explicar, lembra daquela ordem que usamos ABCDEFG nos dígitos? Vamos pegar o valor 9 por exemplo, o 9 não tem o display vertical esquerdo inferior (VEI ou VIE), que sabemos que é o F, então se o desenho do número 9 fosse ser um número, seria 1111101, certo? Ou VVVVVFV. O 9 é 1001 em binário, assim como eram 7 displays, também são 4 dígitos, então teremos 4 circuitos, um para cada algarismo. Então como são 7 variáveis, teremos uma tabela de 128 linhas, com os displays acesos pelo usuário representando um número entre 0 e 127, onde apenas 9 dessas combinações entre os displays representa um número real em binário, e os outros 119 representam o número 0 (apesar de 118 não existirem).

Deve estar bem confuso, eu imagino.. Vamos colocar em prática e vai simplificar.

$R: 29.0, 7.0, 30.0, (27), 7.0, (77),$
 $27.0, 7.0, 19.0, 7.0, 9.0, (77), 5.0,$
 $7.0, 7.0, 7.0 = 720$

$T: 29.0, (77), 30.0, 7.0, (77), 7.0, (77),$
 $27.0, 7.0, 19.0, (77), 9.0, 7.0, 5.0,$
 $(77), 7.0, 7.0 < 728$

ABCDEFGG --- WERT

[illegible][illegible]

as colunas em `pe` representam os numeros formados por cada configuração dos displays

Para o dígito W (milhar), a expressão lógica será $a \wedge b \wedge c \wedge d \wedge e \wedge g$

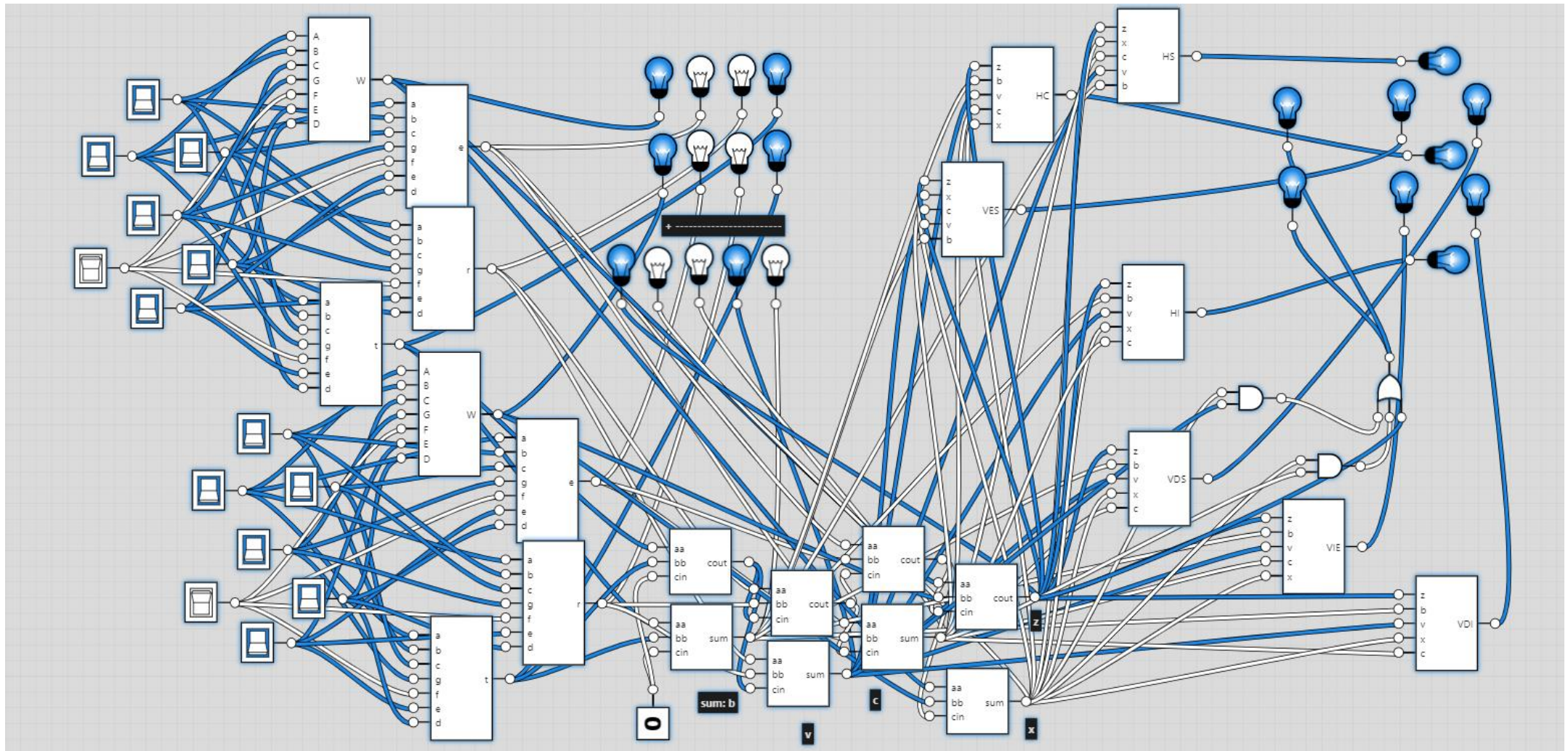
Para o E: $(a \wedge b \wedge \neg c \wedge d \wedge e \wedge g) \vee (a \wedge \neg b \wedge c \wedge d \wedge \neg e \wedge \neg f \wedge g) \vee (\neg a \wedge b \wedge c \wedge d \wedge \neg e \wedge \neg f \wedge \neg g)$

Para o R: $(a \wedge b \wedge \neg c \wedge d \wedge e \wedge f \wedge g) \vee (\neg a \wedge b \wedge c \wedge d \wedge e \wedge \neg f \wedge g) \vee (\neg a \wedge b \wedge c \wedge d \wedge \neg e \wedge \neg f \wedge \neg g) \vee (\neg a \wedge b \wedge c \wedge \neg d \wedge e \wedge f \wedge g)$

Para o T: $(a \wedge b \wedge d \wedge e \wedge \neg f \wedge g) \vee (\neg a \wedge c \wedge d \wedge \neg e \wedge \neg f \wedge \neg g) \vee (b \wedge c \wedge d \wedge e \wedge \neg f \wedge g)$

assim já podemos montar os circuitos. Como não são tão complicados, eu não salvei as imagens, e já imediatamente montei o resultado final. Uni o decoder, o encoder, a soma e o overflow, juntos, de forma organizada temos o resultado final. Antes de mostrar a imagem, devo dizer que o link salvo é tão grande, que upei no mediafire. <https://www.mediafire.com/file/wrz5lkyhlmtkkcq/compartilhar.txt/file>

Ai está o resultado de todo nosso trabalho... Caramba..
Esse projeto pode ser dividido em 4 partes, na esquerda está o decoder, na direita o encoder junto ao overflow. Na parte inferior central vemos a soma, e na superior central vemos a demonstração, onde vemos $9 + 9 = 18$ corretamente em binário



Agora que finalizamos, já é possível passar o projeto para a linguagem C

```
//AQUI OS DIGITOS DE CADA ENTRADA SERÃO CALCULADOS DEPENDENDO DO DESENHO FEITO PELO USUÁRIO
```

```
W = HEXAND(VES1,HS1,VDS1,VDI1,HI1,HC1);
```

```
E = TRIOR(HEXAND(VES1,HS1,N(VDS1),VDI1,HI1,HC1),HEPTAND(VES1,N(HS1),VDS1,VDI1,N(HI1),N(VEI1),HC1),HEPTAND(N(VES1),HS1,VDS1,VDI1,N(HI1),N(VEI1),N(HC1)));
```

```
R = TETRAOR(HEPTAND(VES1,HS1,N(VDS1),VDI1,HI1,VEI1,HC1),HEPTAND(N(VES1),HS1,VDS1,VDI1,HI1,N(VEI1),HC1),  
HEPTAND(N(VES1),HS1,VDS1,VDI1,N(HI1),N(VEI1),N(HC1)),HEPTAND(N(VES1),HS1,VDS1,N(VDI1),HI1,VEI1,HC1));
```

```
T = TRIOR(HEXAND(VES1,HS1,VDI1,HI1,N(VEI1),HC1),HEXAND(N(VES1),VDS1,VDI1,N(HI1),N(VEI1),N(HC1)),HEXAND(HS1,VDS1,VDI1,HI1,N(VEI1),HC1));
```

```
S = HEXAND(VES2,HS2,VDS2,VDI2,HI2,HC2);
```

```
D = TRIOR(HEXAND(VES2,HS2,N(VDS2),VDI2,HI2,HC2),HEPTAND(VES2,N(HS2),VDS2,VDI2,N(HI2),N(VEI2),HC2),HEPTAND(N(VES2),HS2,VDS2,VDI2,N(HI2),N(VEI2),N(HC2)));
```

```
F = TETRAOR(HEPTAND(VES2,HS2,N(VDS2),VDI2,HI2,VEI2,HC2),HEPTAND(N(VES2),HS2,VDS2,VDI2,HI2,N(VEI2),HC2),  
HEPTAND(N(VES2),HS2,VDS2,VDI2,N(HI2),N(VEI2),N(HC2)),HEPTAND(N(VES2),HS2,VDS2,N(VDI2),HI2,VEI2,HC2));
```

```
G = TRIOR(HEXAND(VES2,HS2,VDI2,HI2,N(VEI2),HC2),HEXAND(N(VES2),VDS2,VDI2,N(HI2),N(VEI2),N(HC2)),HEXAND(HS2,VDS2,VDI2,HI2,N(VEI2),HC2));
```

Para a parte gráfica, simples, caso um display seja verdadeiro, imprimir um espaço em branco onde ele ficaria. Os displays de entrada geram os valores a ser somados.

Primeiro declarar as funções básicas, como vimos na página 2, a de soma, de carry, e iniciar a Main. Inicializar as variáveis, os 23 displays e 13 dígitos, assim como os 5 carries e 2 mecânicas ou gráficas. Num arquivo separado, criar as extras, como TRIOR (3 or)

```
//ESSAS FUNÇÕES DESENHAM OS DISPLAYS NA TELA, APENAS A PARTE GRÁFICA
```

```
if(clear <= 4){
```

```
    interfac(W,E,R,T,S,D,F,G,Z,X,C,V,B);  
    digital(HS,HC,HI,VES,VEI,VDS,VDI,overflow);  
    digital1(HS1,HC1,HI1,VES1,VEI1,VDS1,VDI1);  
    digital2(HS2,HC2,HI2,VES2,VEI2,VDS2,VDI2);  
    partes();  
}
```

Após a soma, o valor obtido gera os displays da saída.

```
//AQUI OS VALORES DO DISPLAY DO RESULTADO SÃO CONVERTIDO PARA VERIFICAR SE ACENDERÃO OU NÃO
```

```
VES = OCTOR(TRIAND(Z,X,N(V)),TRIAND(X,N(C),N(V)),TRIAND(X,V,N(B)),TRIAND(N(X),N(V),N(B)),TETRAND(Z,N(X),N(C),V),TETRAND(N(Z),X,C,V),TETRAND(N(Z),N(X),C,N(V)),TETRAND(N(Z),N(X),C,N(B)));
```

```
VEI = PENTAOR(TRIAND(Z,N(X),N(B)),TRIAND(Z,V,N(B)),TRIAND(N(Z),N(C),N(B)),TRIAND(N(X),V,N(B)),TETRAND(X,C,N(V),N(B)));
```

```
VDS = ENEAOR(TRIAND(Z,N(X),V),TRIAND(Z,N(X),B),TRIAND(Z,V,B),TRIAND(X,C,N(V)),TRIAND(X,C,N(B)),TRIAND(X,N(V),N(B)),TRIAND(N(X),V,B),TRIAND(C,N(V),N(B)),AND(N(Z),N(C)));
```

```
VDI = HEXAOR(AND(Z,X),AND(X,N(C)),AND(Z,N(C)),AND(N(X),N(V)),TRIAND(N(Z),C,V),B);
```

```
HC = HEPTAOR(TRIAND(Z,N(X),V),TRIAND(Z,N(C),N(B)),TRIAND(N(Z),X,C),TRIAND(N(Z),C,N(V)),TRIAND(N(Z),C,N(B)),AND(X,N(V)),TRIAND(N(X),N(C),V));
```

```
HS = OCTOR(TRIAND(Z,N(X),N(C)),TRIAND(Z,C,N(B)),TRIAND(Z,N(C),V),TRIAND(N(Z),C,B),TRIAND(N(Z),N(C),N(B)),TRIAND(X,C,N(V)),TRIAND(X,N(V),B),AND(N(X),V));
```

```
HI = DECAOR(TRIAND(Z,N(X),V),TRIAND(Z,N(X),N(B)),TRIAND(Z,V,N(B)),TRIAND(N(Z),N(C),N(B)),TRIAND(X,C,N(V)),  
TRIAND(X,N(V),B),TRIAND(N(X),N(C),V),TRIAND(N(X),V,N(B)),TETRAND(N(Z),X,C,B),TETRAND(N(Z),C,N(V),B));
```

```
overflow = OR(Z,AND(X,OR(C,V)));
```

```
//AGORA QUE SE TEM OS VALORES DAS DUAS ENTRADAS, A OPERAÇÃO SERÁ REALIZADA
```

```
B = SUM(T,G,RIU);
```

```
RID = carry_out(T,G,RIU);
```

```
V = SUM(R,F,RID);
```

```
RIC = carry_out(R,F,RID);
```

```
C = SUM(E,D,RIC);
```

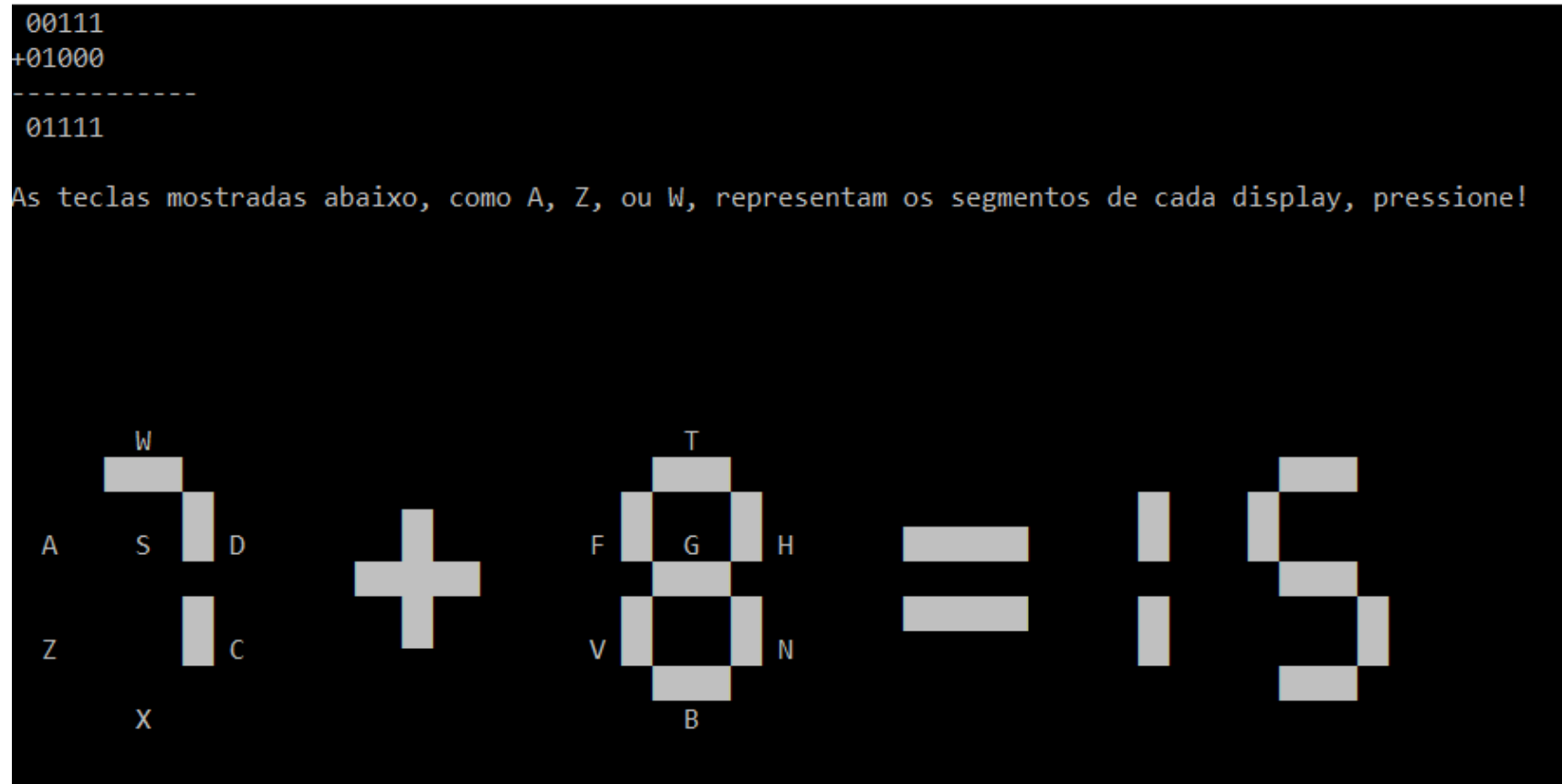
```
RIM = carry_out(E,D,RIC);
```

```
X = SUM(W,S,RIM);
```

```
RIDM = carry_out(W,S,RIM);
```

```
Z = RIDM;
```

O programa funciona de forma que ao pressionar a tecla que representa cada display, a situação atual desse interruptor será alterada, fazendo ele acender ou apagar, usando a função kbhit que verifica se alguma tecla foi pressionada, e depois é só verificar qual foi. E é isso, muito obrigado a todos que acompanharam! :)



Link para testar o programa: <https://www.mediafire.com/file/hscas9r6wc258bs/Projeto1.exe/file>