Prática de Lab. de Sistemas Digital – Eng. de Computação – Belo Horizonte

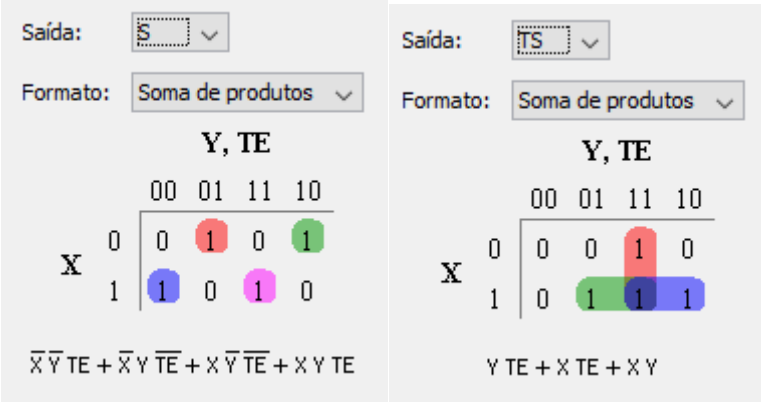
Prof. Mara C. S. Coelho – DECOM

Alunos: Alexandre Roque; Henrique Coelho; Vitor Santana.

Tabela verdade para o somador completo de 1 bit:

X	Y	TE	S	TS
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Obtenção das expressões booleanas a partir do mapa de Karnaugh:



A partir da interpretação da Saída “S”, é possível perceber que se trata de uma junção de alguns “Ou exclusivo”, ou seja, portas XOR. Então obtemos:

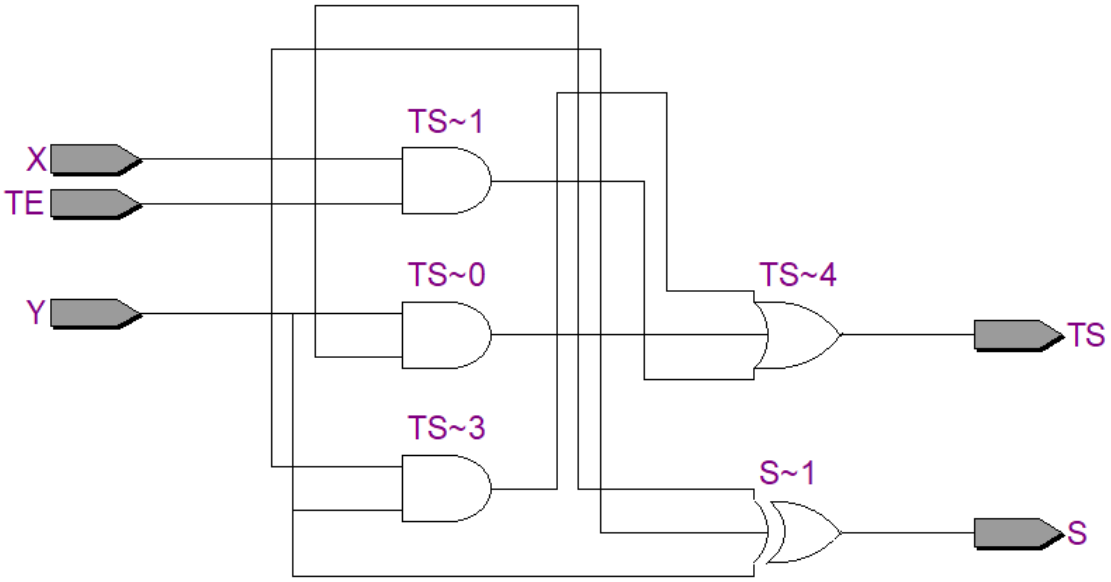
S = X ⊕ Y ⊕ TE.

TS = (Y & TE) | (X & TE) | (X & Y).

Código do somador completo de 1 bit:

```
 8   module somador1bit(X,Y,TE,S,TS);
 9
10      input X,Y,TE;
11      output S,TS;
12
13      assign S = X ^ Y ^ TE;
14      assign TS = (Y & TE) | (X & TE) | (X & Y);
15
16  endmodule
17
```

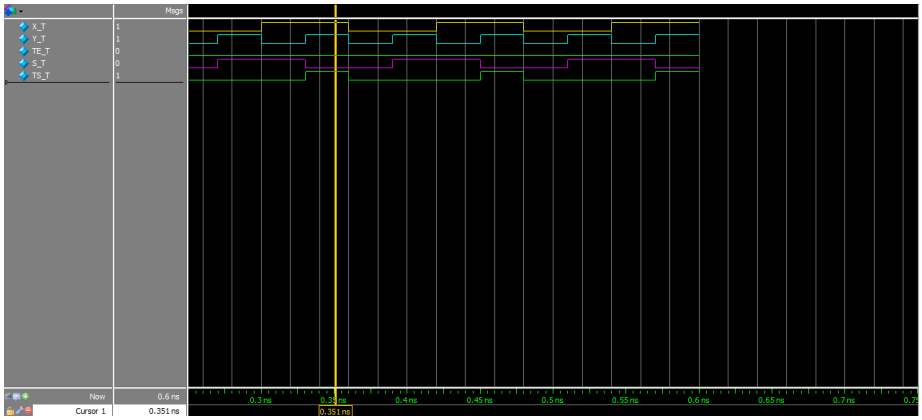
RTL Viewer do somador completo de 1 bit:



Código do Teste Bench usado para a simulação do circuito somador de 1 bit:

```
 8   module testebench_somador;
 9      //variaveis intermediarias
10      reg X_T, Y_T;
11      reg TE_T;
12
13      wire S_T;
14      wire TS_T;
15      //somadorlbit(X,Y,TE,S,TS);
16
17      somadorlbit dut (X_T, Y_T, TE_T, S_T, TS_T);
18
19      initial
20         begin
21            repeat (5)
22               begin
23                  TE_T = 0;
24                  X_T = 0; Y_T = 0;  #30
25                  X_T = 0; Y_T = 1;  #30
26                  X_T = 1; Y_T = 0;  #30
27                  X_T = 1; Y_T = 1;  #30;
28
29               end //repeat
30         end //initial
31   endmodule
```

Simulação no ModelSim do circuito somador completo de 1 bit:



Especificações e análise:

Entrada X_T: Em amarelo. Entrada Y _T: Em ciano. Entrada TE_T: Em verde

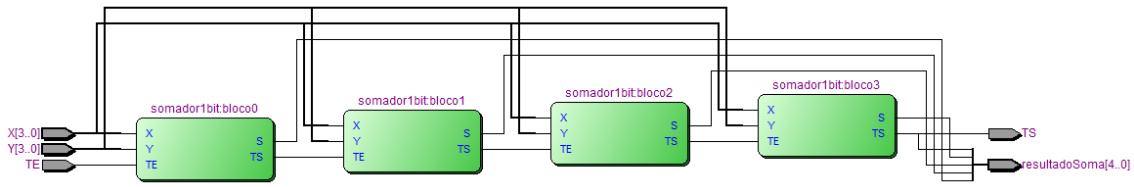
Saída S_T: Em magenta. Saída TS_T: Em verde default.

Como exemplificado no TestBench, ciclos variando de 30 ps, fazendo com que todas as opções da tabela verdade sejam lidas. Observa-se que o circuito está funcionando de acordo com a tabela verdade, No momento em especifico, é possível notar que com as entradas X_T =1 e Y_T = 1, a saída S_T = 0 e TS_T = 1. Isso representa o “vai um”(Carry) na soma de números binários.

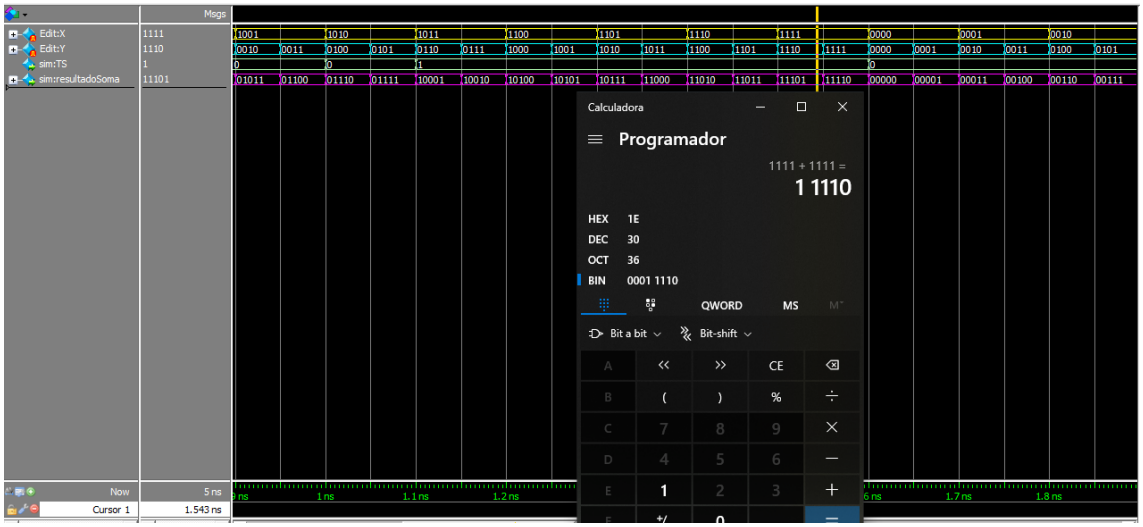
Código para o somador de 4 bits, utilizando estrutura hierárquica:

```
 8  module somador4bit(X, Y, TE, resultadoSoma, TS);
 9
10     input [3:0] X, Y;
11     input TE;
12
13     output [4:0] resultadoSoma;
14     output TS;
15
16     wire [3:0] S;
17     wire [2:0] vaiUm;
18
19     somadorlbit bloco0 (X[0] , Y[0] , TE       , S[0] , vaiUm[0]);
20     somadorlbit bloco1 (X[1] , Y[1] , vaiUm[0] , S[1] , vaiUm[1]);
21     somadorlbit bloco2 (X[2] , Y[2] , vaiUm[1] , S[2] , vaiUm[2]);
22     somadorlbit bloco3 (X[3] , Y[3] , vaiUm[2] , S[3] , TS      );
23
24     assign resultadoSoma [4:0] = {TS , S[3:0]};
25
26  endmodule
27
```

RTL Viewer do somador de 4 bits:



Simulação ModelSim do circuito somador de 4 bits:



Especificações e análise:

Entrada X: Representada em Amarelo. Em modo Counter, com o timer de 50 ps.

Entrada Y: Representada em Ciano. Em modo Counter, com o timer de 100 ps.

Saída TS: Representada em Verde Default.

Saida resultadoSoma: Representada em Magenta.

Start time: 0 ps. End time: 5000 ps.

Como podemos observar na simulação, a soma dos números X e Y é feita e exibida como a saída resultadoSoma, composta pelo TS(Carry) e o vetor S[3:0](A soma propriamente dita). Vale salientar que os números X e Y, são na verdade, um conjunto de números (X0,X1,X2,X3) e (Y0,Y1,Y2,Y3). A partir da utilização da calculadora do sistema operacional Windows, no modo Programador, conseguimos confirmar o resultado da soma 1111 + 1111 em binário, resultando 11110, representado logo após a barra amarela no gráfico.