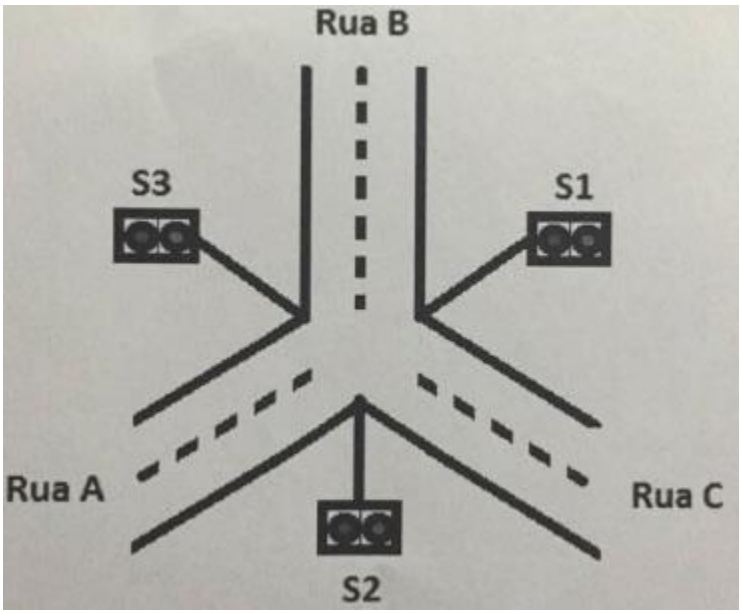
Laboratório de Sistemas Computacionais: Circuitos Digitais

Controle de Semáforos

Em um cruzamento entre três ruas de mão dupla são instalados três semáforos em cada rua para controlar o fluxo de automóveis que irão transitar pelo cruzamento. Os semáforos respeitam as seguintes regras:

- Os semáforos possuem apenas as luzes vermelho e verde;
- Quando um semáforo abre, o motorista pode fazer ambas as conversões. Desta forma, nunca deve haver mais de um semáforo aberto;
- O motorista que estiver na Rua A tem prioridade em relação ao motorista da Rua B;
- O motorista que estiver na Rua B tem prioridade em relação ao motorista da Rua C;
- O motorista que estivar na Rua C tem prioridade em relação ao motorista da Rua A;
- Quando houver carros nas três ruas, o motorista da Rua A tem preferência;
- Quando não houver carros, o sinal deve estar aberto para a Rua A.

Projete um circuito digital para fazer o controle dos semáforos.



O primeiro passo para resolver o problema é a elaboração da **tabela verdade** de acordo com as regras estabelecidas pelo problema.

A tabela verdade descreve o todo o funcionamento lógico de um circuito digital combinacional, ou todo o comportamento de uma equação lógica. Assim, deve contemplar todas as possibilidades de entrada do circuito/equação e cada saída correspondente.

Para a elaboração da tabela verdade de acordo com o problema, considere sendo:

- A: Sensor de presença da rua A (0 – Ausente; 1 – Presente) ;
- B: Sensor de presença da rua B (0 – Ausente; 1 – Presente) ;
- C: Sensor de presença da rua C (0 - Ausente; 1 – Presente) ;
- S1: Semáforo de liberação da rua A (0 – Luz vermelha; 1 – Luz verde) ;
- S2: Semáforo de liberação da rua B (0 – Luz vermelha; 1 – Luz verde) ;
- S3: Semáforo de liberação da rua C (0 – Luz vermelha; 1 – Luz verde) ;

Entradas			**Saídas**		
A	**B**	**C**	**S1**	**S2**	**S3**
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	0	0	1
1	1	0	1	0	0
1	1	1	1	0	0

Para extrair as expressões da tabela verdade, pode-se usar **Mintermos** ou **Maxtermos**.

Mintermos e maxtermos são utilizados para reescrever uma função lógica em uma forma padronizada no sentido de obter-se uma simplificação da mesma. A expressão é escrita na forma de uma Soma de Produtos (Mintermos) ou na forma de um Produto de Somas (Maxtermos).

Consideramos para cada coluna das saídas (S1, S2, S3), as linhas que resultam em nível lógico alto (1) para montarmos as expressões.

Utilizando mintermos, podemos fazer uma soma de produtos para formar as expressões das saídas, onde as entradas com nível lógico baixo (0) devem ser consideradas como negadas na expressão.

Dessa forma, para a coluna S1 temos:

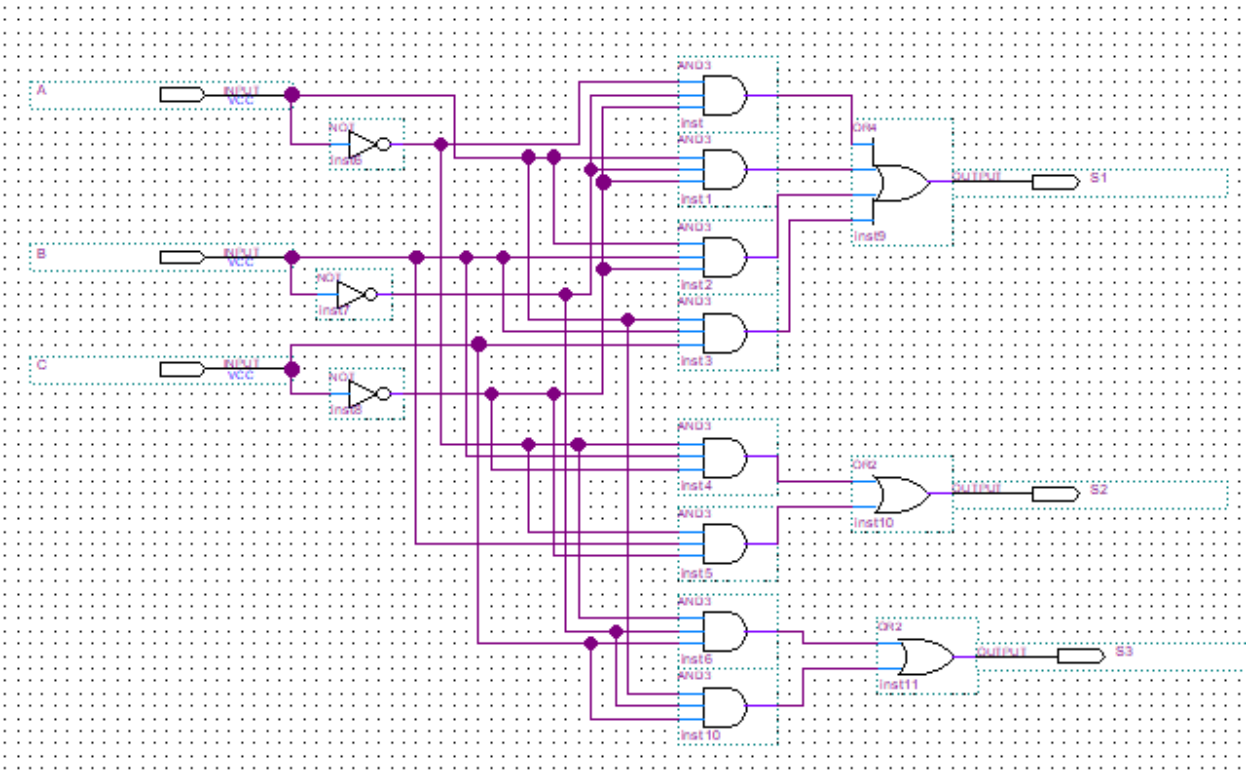
A	B	C	S1	Mintermo
0	0	0	1	A’B’C’
1	0	0	1	AB’C’
1	1	0	1	ABC’
1	1	1	1	ABC

Fazendo a soma dos produtos (mintermos), obtemos a seguinte expressão de S1 = A’B’C’ + AB’C’ + ABC’ + ABC.

Repetindo o processo para as saídas S2 e S3, obtemos as expressões:

- S1 = A’B’C’ + AB’C’ + ABC’ + ABC
- S2 = A’BC’ + A’BC
- S3 = A’B’C + AB’C

De acordo com as expressões, o circuito digital pode ser representado da seguinte forma:



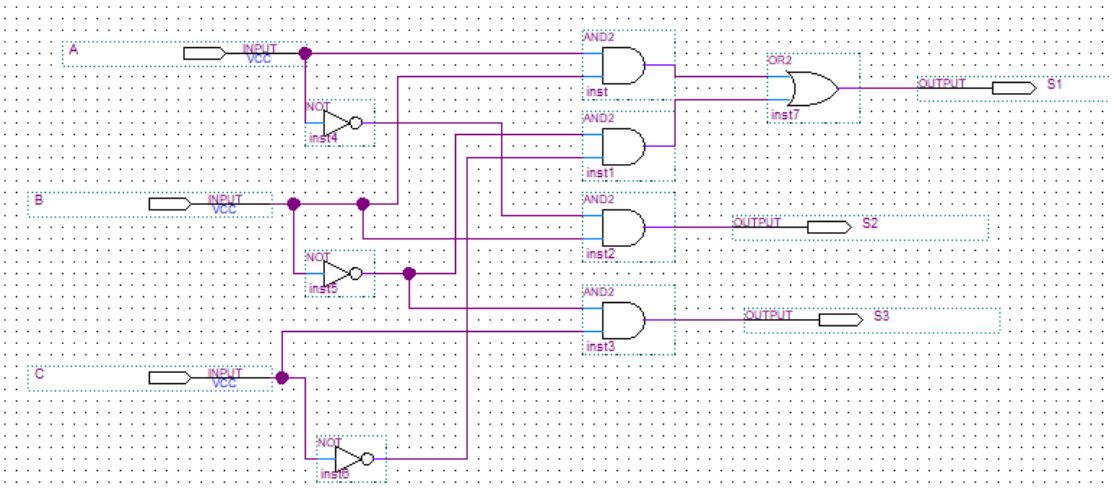
Pode-se perceber que a complexidade do projeto ficou grande devido ao número de portas e fios usados. Porém, podemos simplificar esse circuito digital através da **álgebra booleana**.

A álgebra booleana usa as propriedades dos operadores lógicos para abstrair as operações algébricas em um sistema binário. Podemos utilizar suas propriedades e teoremas, assim como na matemática, para simplificar e encontrar expressões reduzidas, que irão usar menos portas lógicas.

Vamos simplificar as expressões colocando os termos que são comuns em evidência:
S1 = (B'C')(A'+A) + (AB)(C'+C) = AB+B'C'
S2= (A'B)(C'+C) = A'B
S3 = (B'C)(A'+A) = B'C

Assim, temos as seguintes expressões:
- **S1 = AB + B'C'**
- **S2 = A'B**
- **S3 = B'C**

Dessa maneira, o circuito digital para o controle de semáforos é representado da seguinte forma:

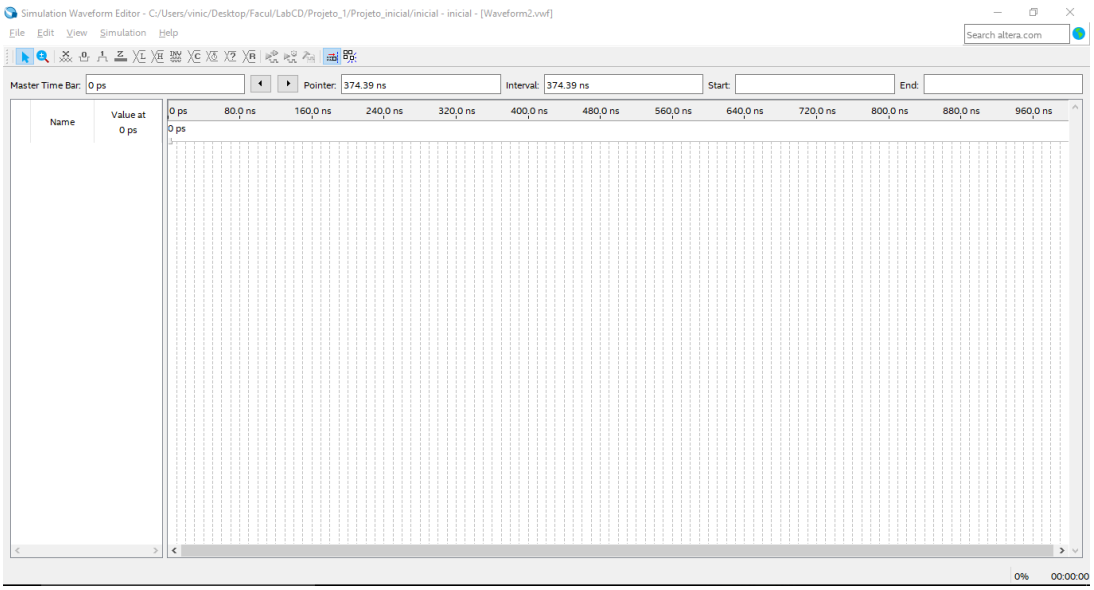


Vale ressaltar que, além da manipulação algébrica, existem outros métodos para a realização de simplificações em equações booleanas, como por exemplo o Mapa de Karnaugh e o método tabular de Quine-McCluskey, os quais não fazem parte do escopo desse tutorial.

Utilizando Simulador de Forma de Ondas (*Simulation Waveform*)

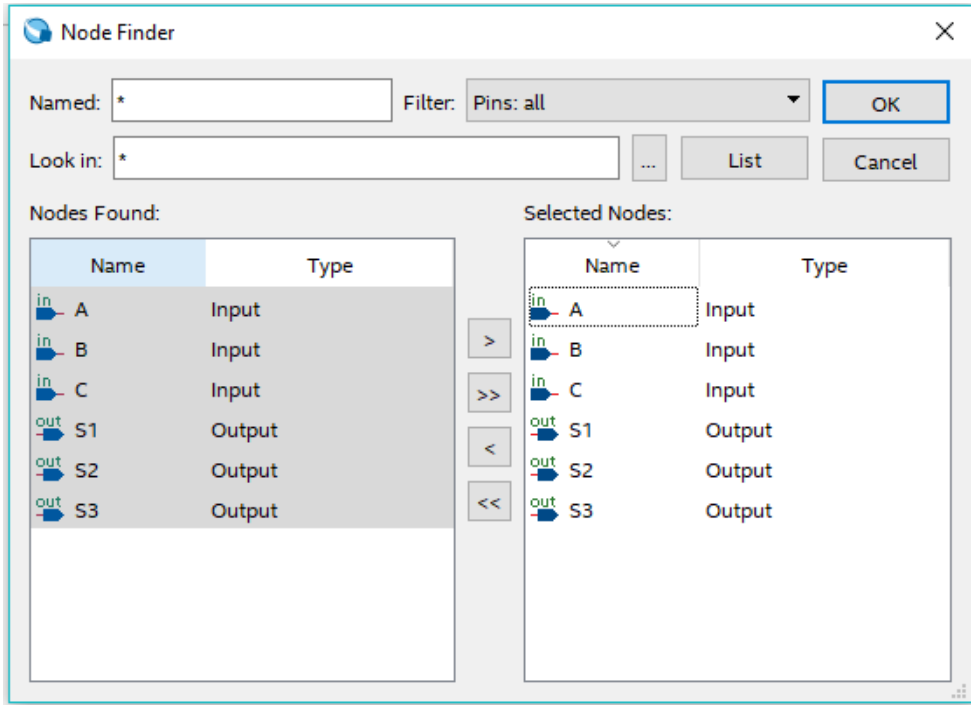
Antes de enviar o projeto para o Kit DE2-15 ou poder realizar testes com no programa, pode-se utilizar *Waveform* para simulação das entradas e saídas lógicas.

Para isso, após compilar o projeto, aperte: *File->New->University Program VWF*



Para inserir os nós de entrada e saída: *Edit -> Insert -> Insert Node or Bus -> Node Finder...* Ao apertar em *List* todas os nós serão mostrados.

Selecione os nós que serão utilizados passando um por um ou passando todos os pontos. Selecione todos os nós.



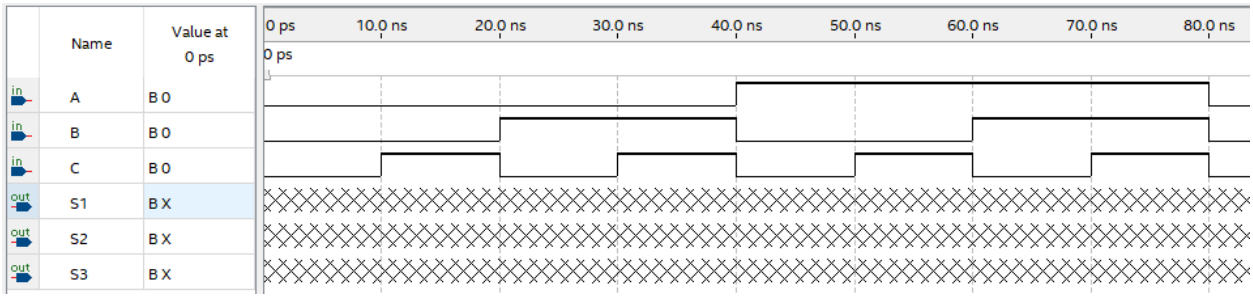
Aperte *OK*.

Agora todos os nós aparecem na tela inicial, informando se são de entrada (*in*) ou saída (*out*), seus nomes e seus valores iniciais.

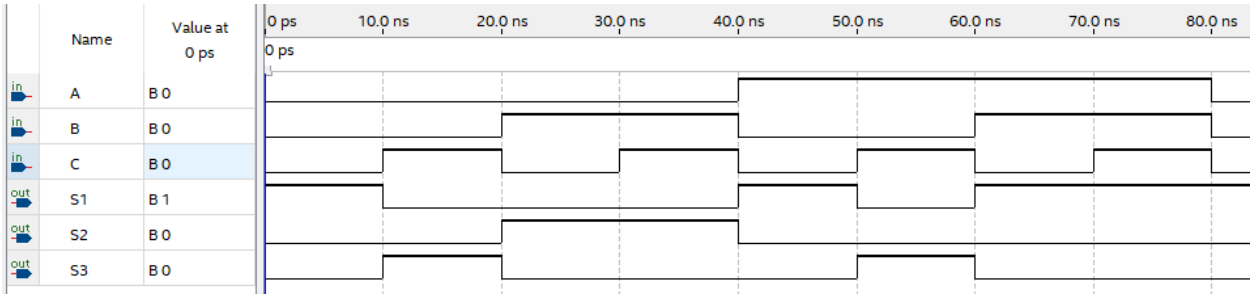
Precisamos controlar os pulsos que serão aplicados aos nós de entrada no decorrer do tempo, para isso, selecione o nó A com o botão direito e no menu superior selecione *Count Value*, aqui é possível alterar a base de conversão do nó, seu valor inicial e o tempo para transição entre 0 e 1 do nó.

- Para o nó A deixe 40ns.
- Para o nó B, 20ns.
- Para o nó C, 10ns.

Sempre prefira deixar os tempos múltiplos para facilitar a leitura do dado em todas possibilidades.



Para compilar: *Simulation -> Run Functional Simulation*



Assim, é possível verificar que as Formas de Onda são as mesmas se comparadas com a Tabela Verdade.

Outra possibilidade é de utilizar na barra de ferramentas superior para definir o valor binário manualmente de uma determinada entrada.

No exemplo, A possui sinal baixo, B alternado a cada 10 ns e C possui sinal alto.