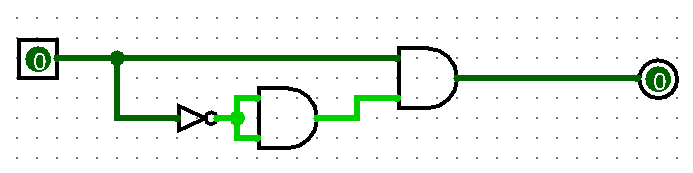
Arquitetura de Computadores 1 – Lista 2

Nome: Lucas Carneiro Nassau Malta

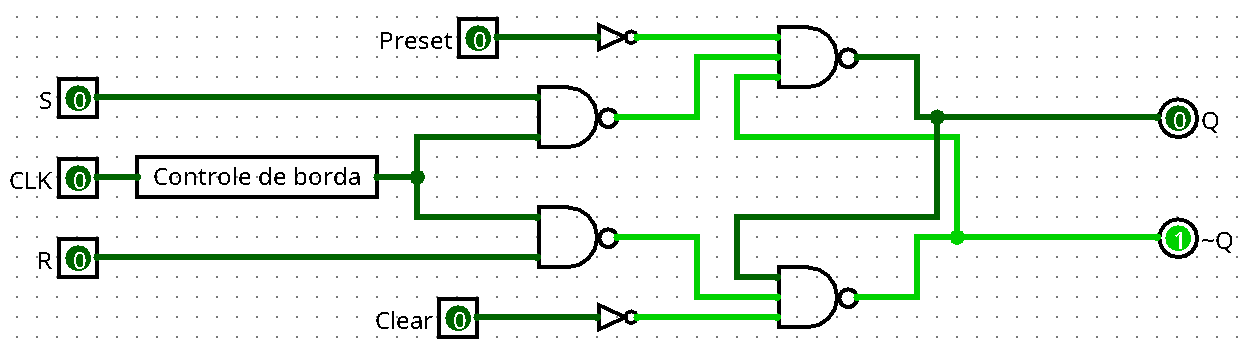
Matrícula: 857340

Antes das questões, seguem os circuitos base para a realização dos exercícios propostos:

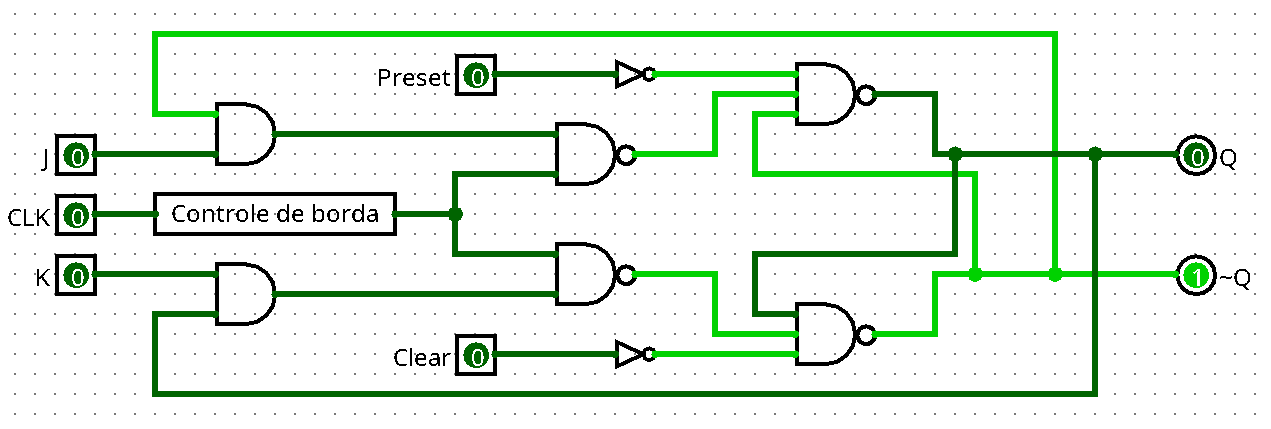
Controle de Borda:



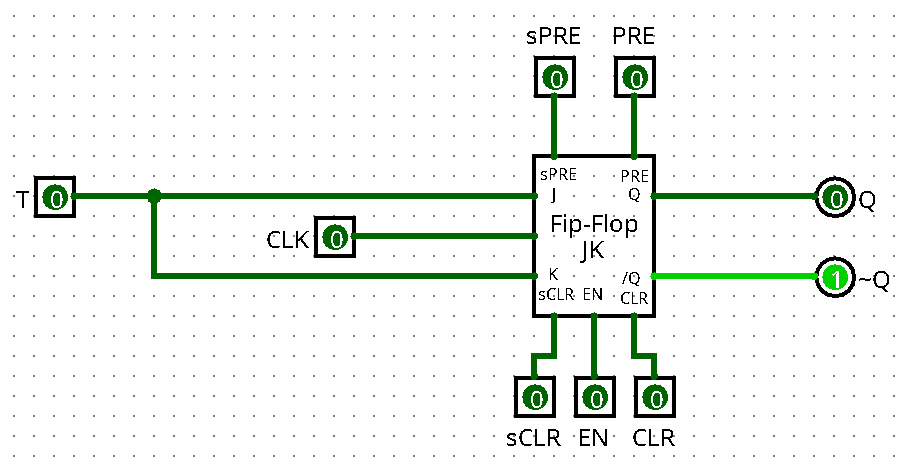
Flip-Flop SR:



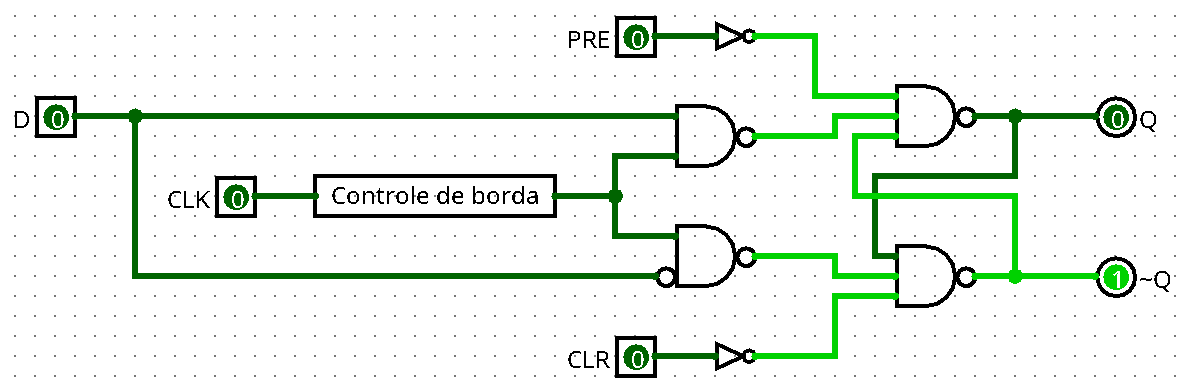
Flip-Flop JK:



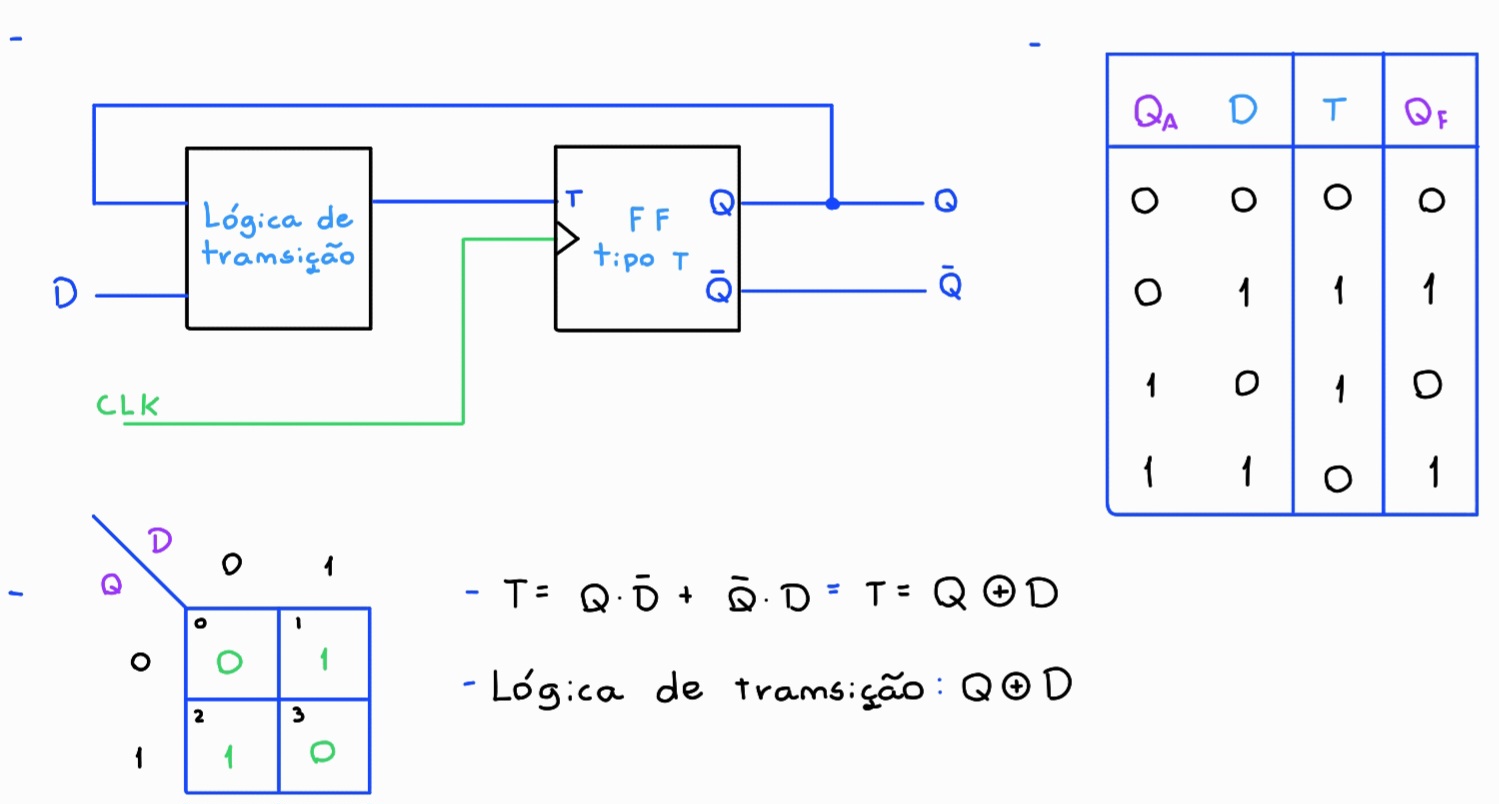
Flip-Flop Tipo T:

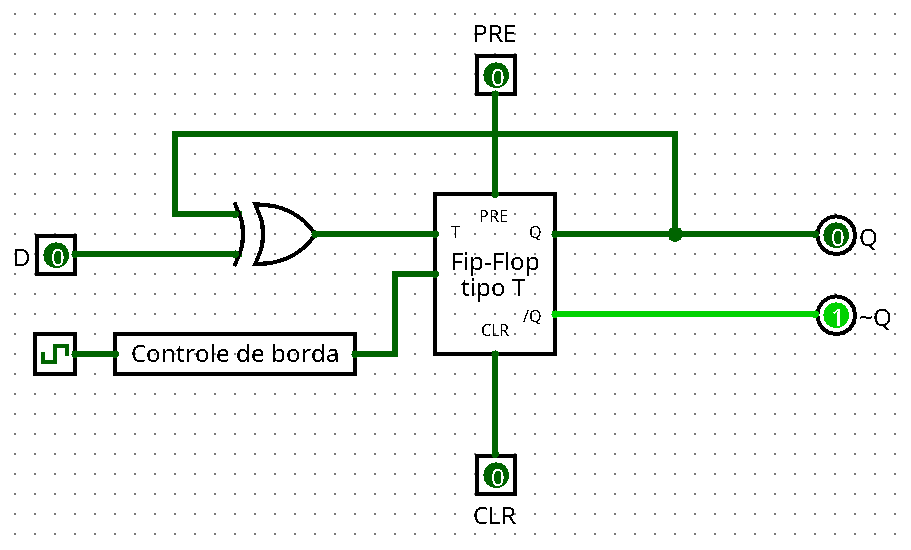


Flip-Flop Tipo D:



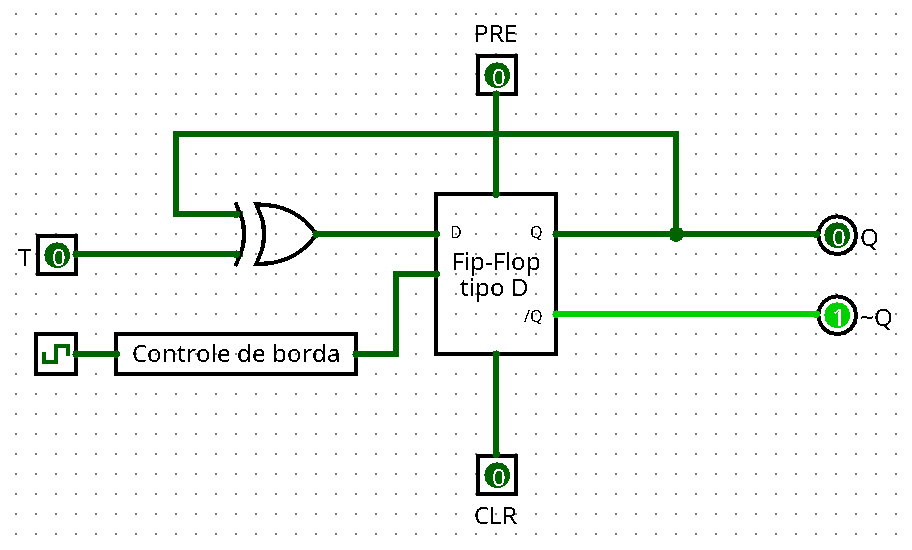
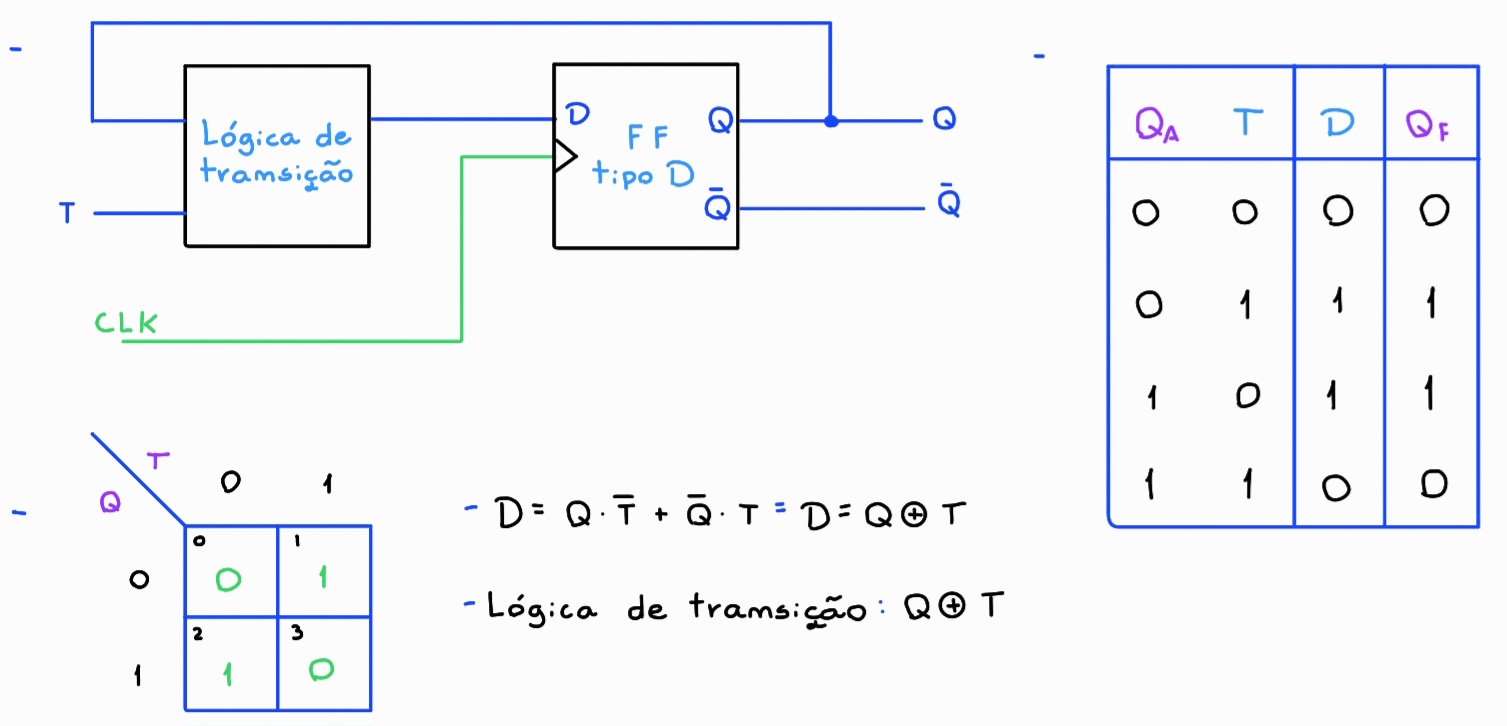
1) Implementar um FF (flip-flop) tipo ‘D’ usando FF ‘T’ e portas lógicas elementares/compostas.





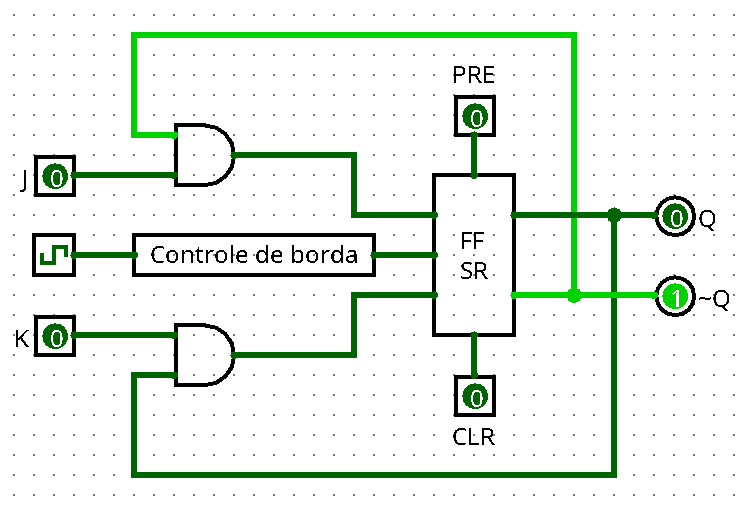
Descrição textual: A entrada CLR inicia o sistema com as saídas Q e ~Q com os valores 0 e 1, respectivamente. Já a entrada PRE inicia essas saídas com os valores 1 e 0. A cada ‘clock’, a saída Q indica o estado da entrada D, enquanto a saída ~Q indica o valor inverso da entrada D.

2) Implementar um FF tipo ‘T’ usando FF ‘D’ e portas lógicas.



Descrição textual: A entrada CLR inicia o sistema com as saídas Q e ~Q com os valores 0 e 1, respectivamente. Já a entrada PRE inicia essas saídas com os valores 1 e 0. A cada ‘clock’, se a entrada T for positiva, as saídas Q e ~Q têm o seu estado invertido.

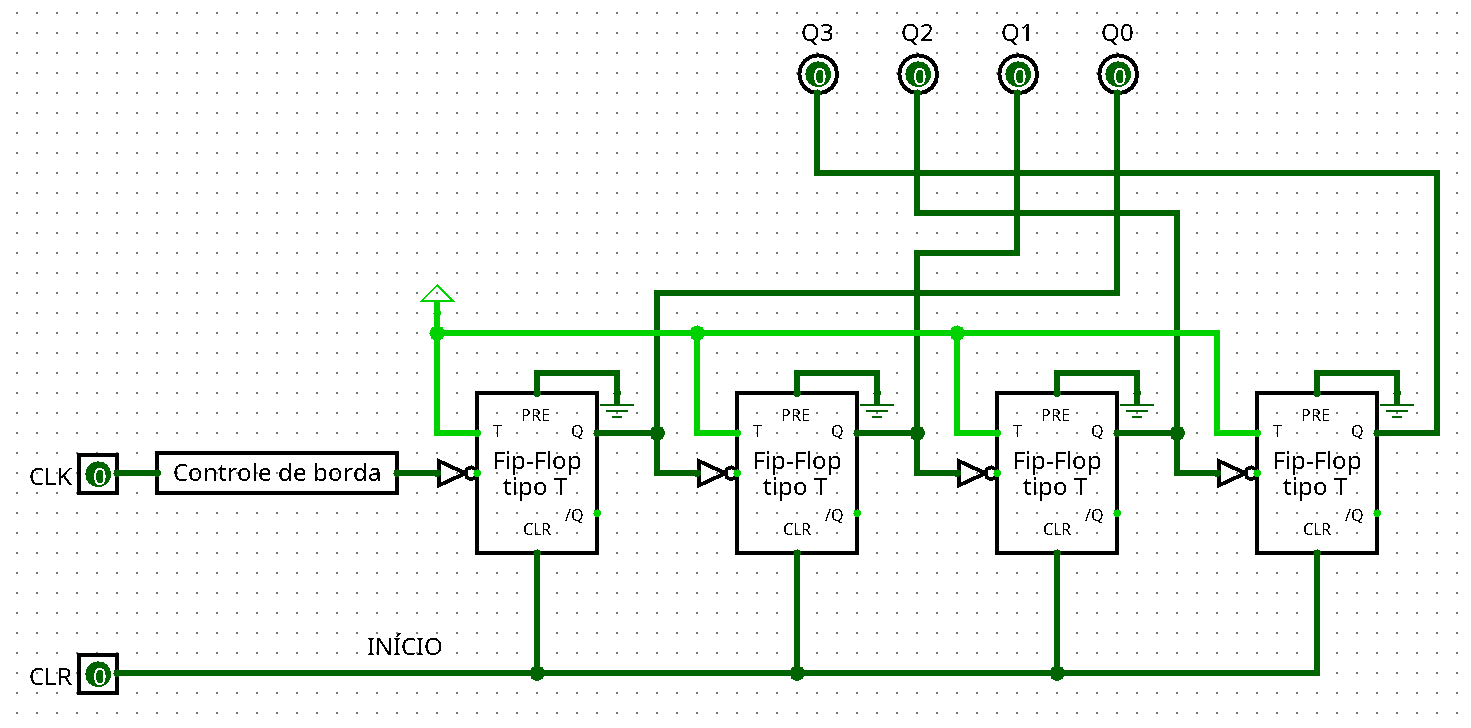
3) Implementar um FF JK usando FF SR e portas lógicas.



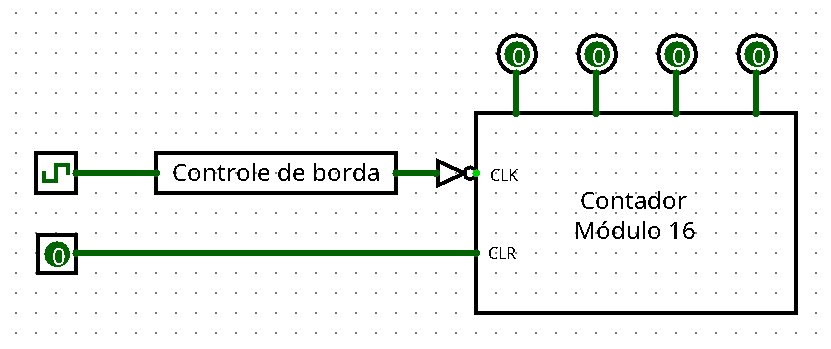
Descrição textual: A entrada CLR inicia o sistema com as saídas Q e ~Q com os valores 0 e 1, respectivamente. Já a entrada PRE inicia essas saídas com os valores 1 e 0. A cada ‘clock’, as entradas J e K podem alteras as saídas Q e ~ Q. Se J e K forem negativos, os valores da saída permanecem inalterados. Se J for positivo e K negativo, as saídas ficam 1 e 0. Se J for negativo e K positivo, as saídas ficam 0 e 1. Por fim, se J e K forem positivos, os valores da saída são invertidos.

4) Implementar um contador BCD assíncrono, sensível à borda de descida, com sinal de Clear.

Contador Módulo 16 assíncrono:



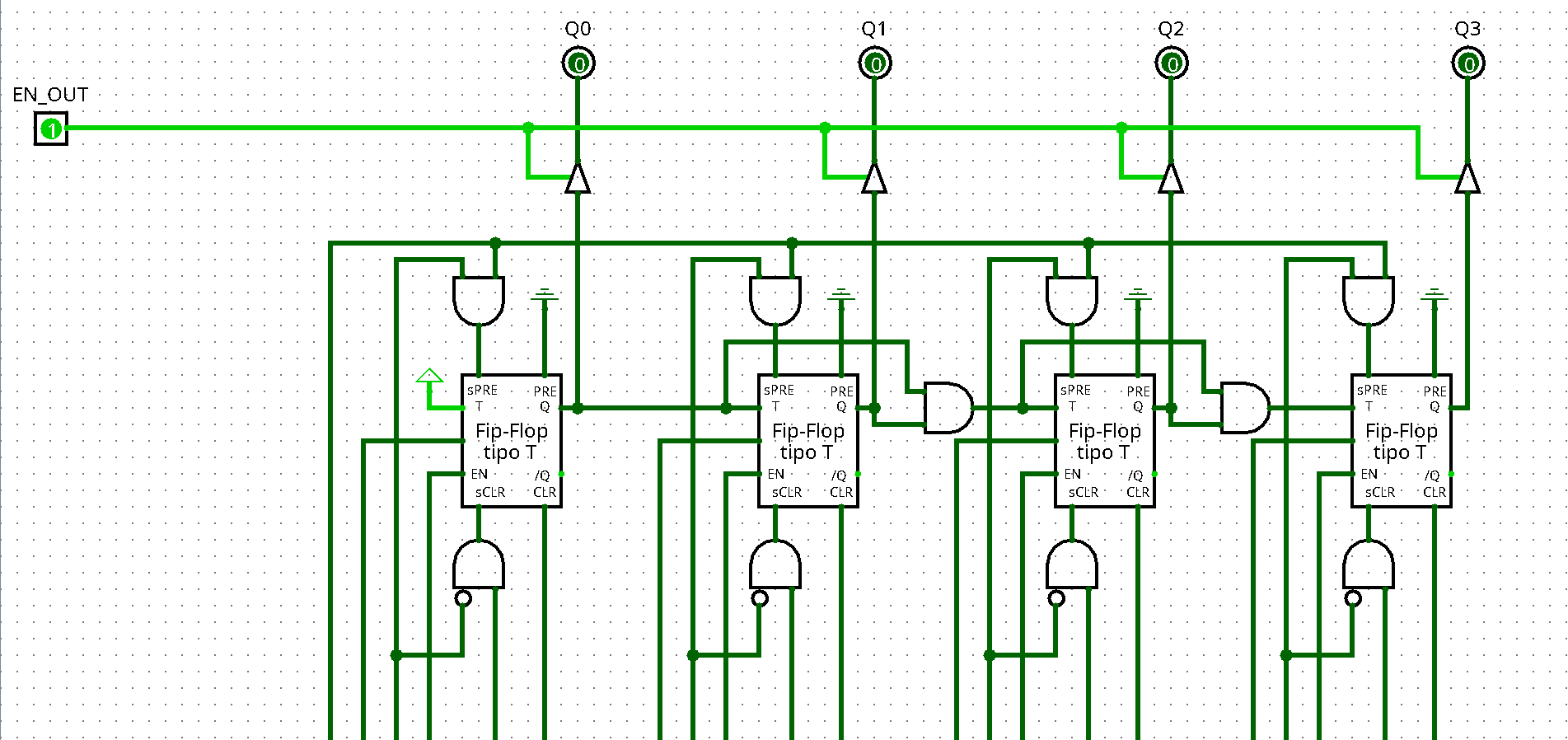
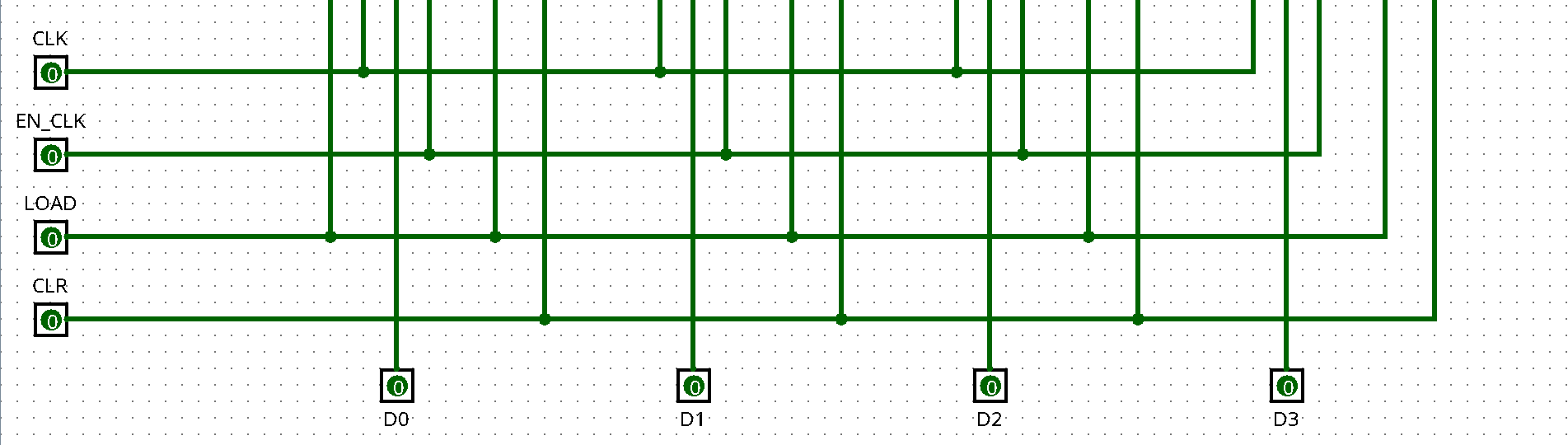
Implementação:



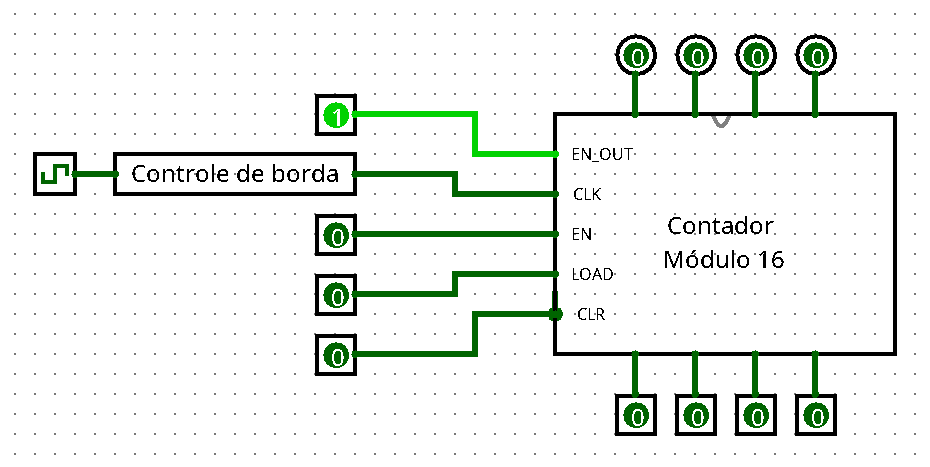
Descrição textual: A entrada CLR inicia o sistema com todas as saídas iguais a 0. A cada ‘clock’, as saídas representam números binários incrementados de um em um que vão, em decimal, de 0 até 15. Esses valores correspondem ao Binary-Coded Decimal.

5) Implementar um contador Módulo 16 síncrono, sensível à borda de subida, com carga paralela síncrona, sinal de Load, Enable Clock

Contador Módulo 16 síncrono:



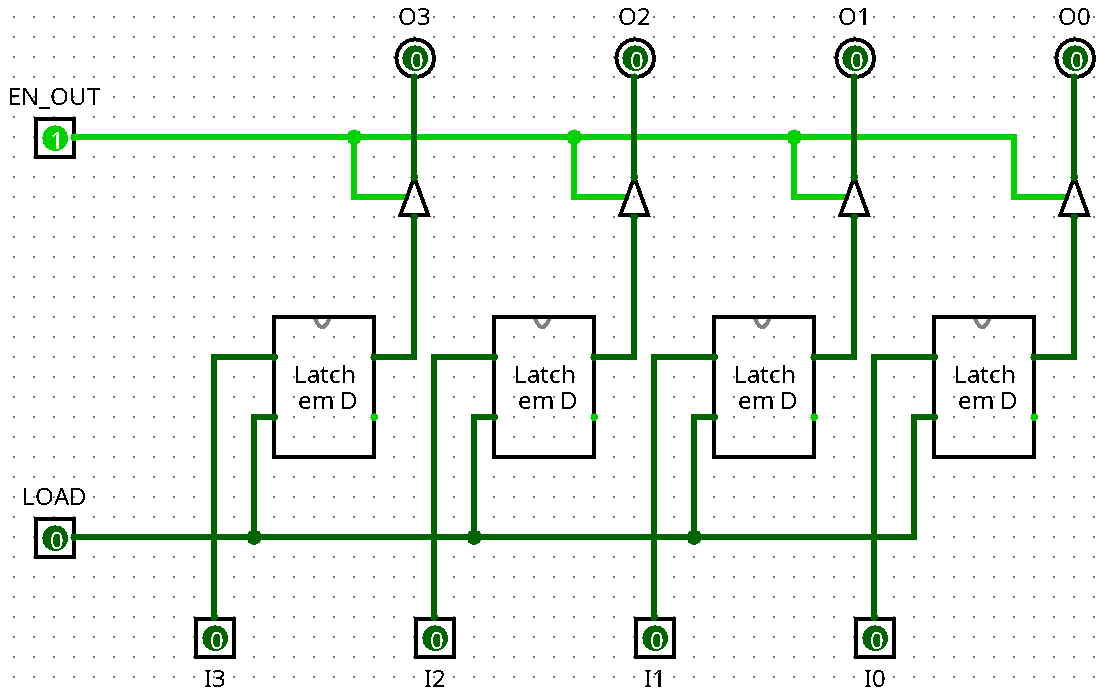
Implementação:



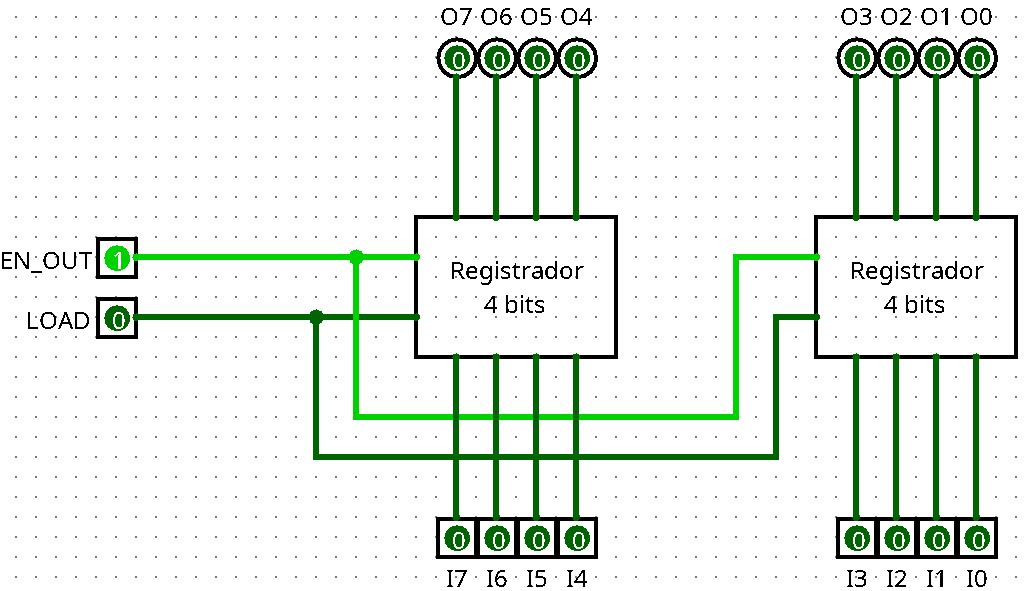
Descrição textual: A entrada EN\_OUT permite o controle da mostragem dos valores das saídas. A entrada EN controla se o pulso de ‘clock’ será válido para o contador ou não. A entrada LOAD faz uma carga paralela dos dados das entradas debaixo diretamente para as saídas. A entrada CLR inicia o contador com todas as saídas iguais a 0. A cada ‘clock’, as saídas representam números binários incrementados de um em um que vão, em decimal, de 0 até 15.

6) Implementar um registrador de dados de 8 bits, com Enable Output e carga paralela síncrona controlada pelo sinal de Load.

Registrador de 4 bits:



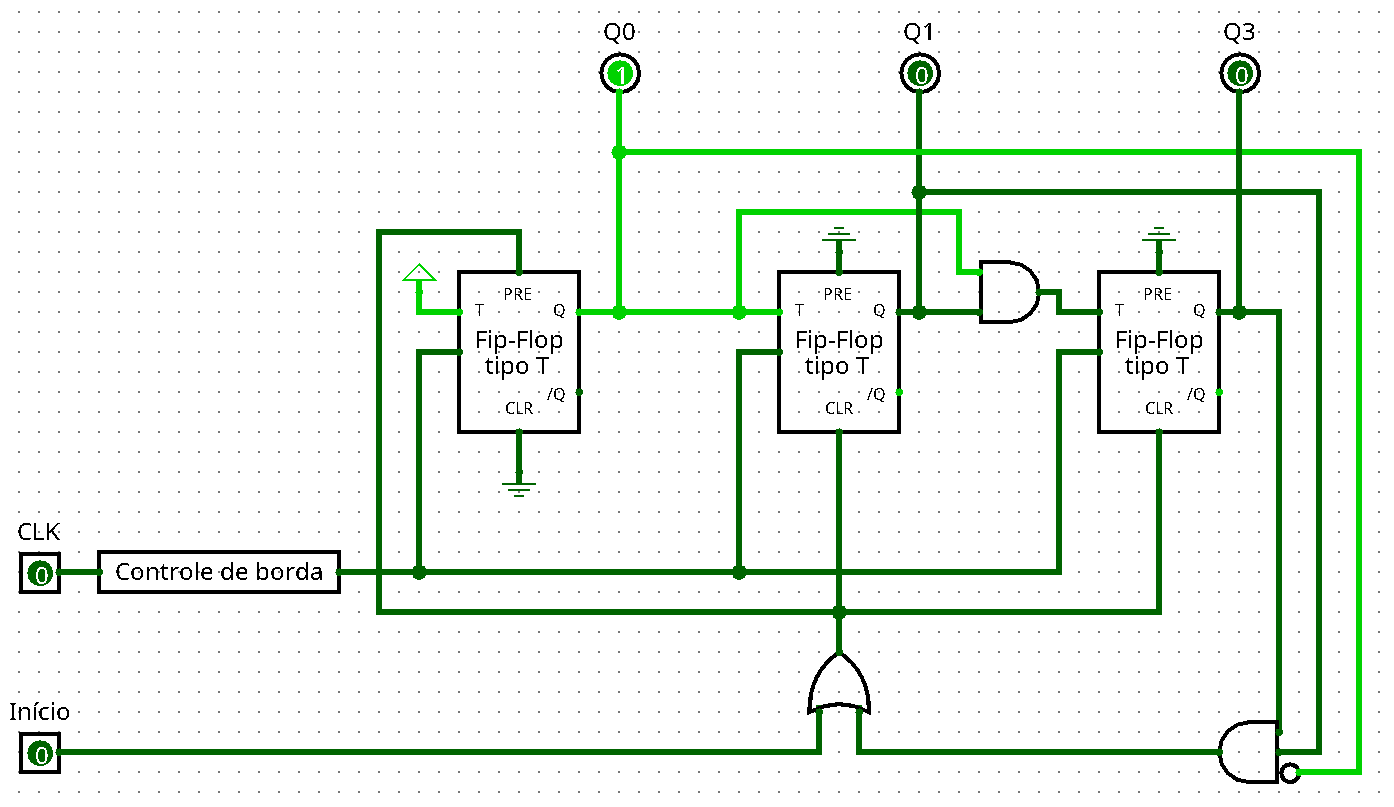
Registrador de 8 bits:



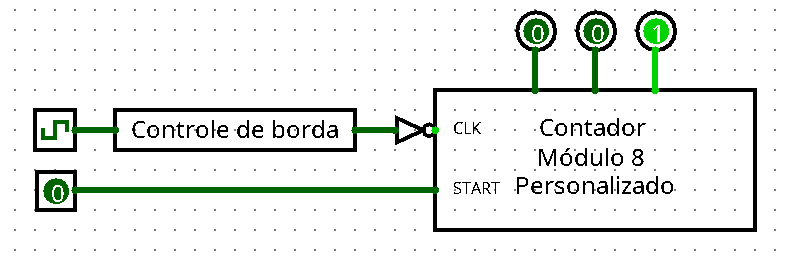
Descrição textual: A entrada EN\_OUT permite o controle da mostragem dos valores das saídas. A entrada LOAD, quando positiva, carrega os valores das entradas I0 até I7 para as saídas O0 até O7.

7) Implementar um contador síncrono que conte indefinidamente a sequência 1, 2, 3, 4, 5, 1, 2, 3, …. A cada pulso negativo de ‘clock’, a contagem avança. Um sínal de início inicia a contagem em 1.

Contador personalizado:



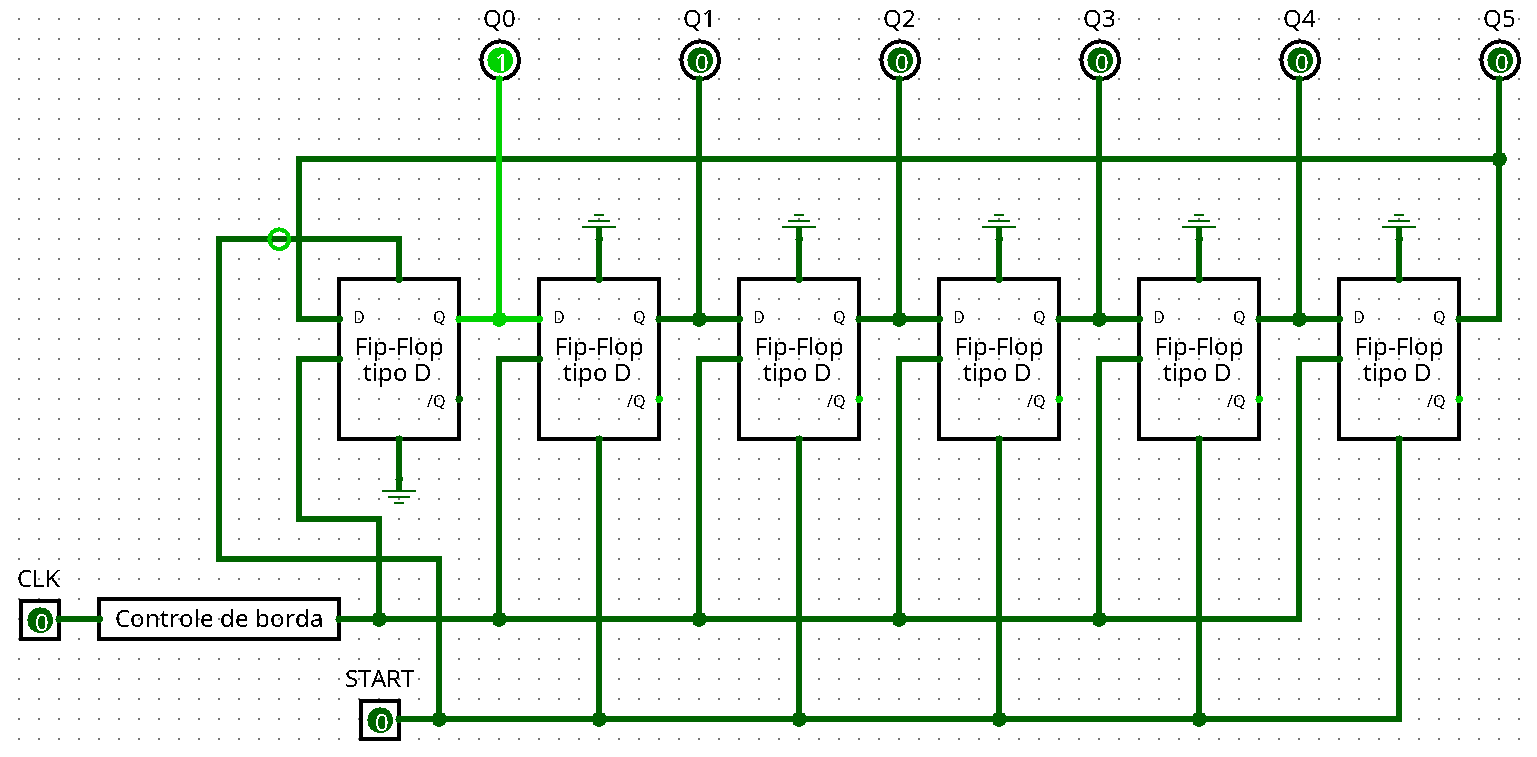
Implementação:



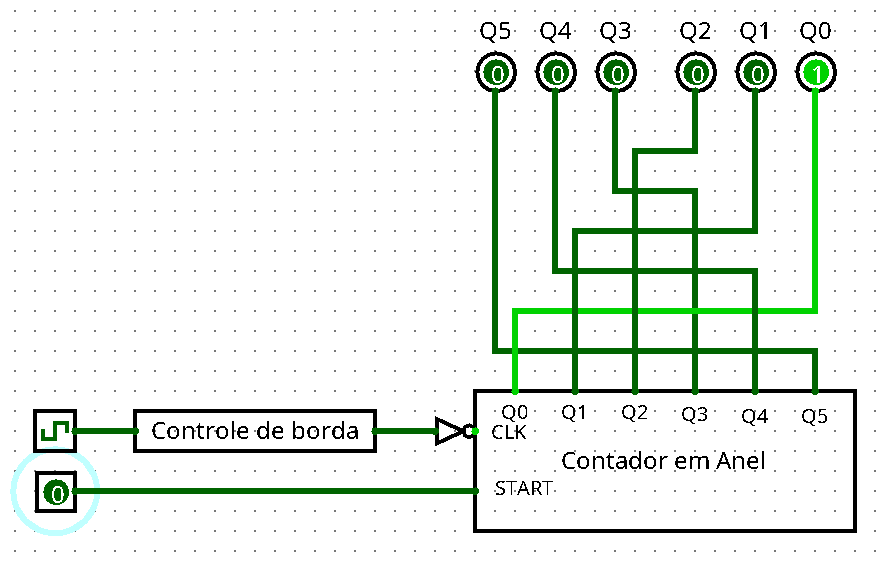
Descrição textual: A entrada START inicia a contagem. A cada ‘clock’, as saídas representam números binários incrementados de um em um que vão, em decimal, de 1 até 5.

8) Implemente um Contador em Anel sensível à borda negativa de ‘clock’ que gera a sequência [Q5, Q4, Q3, Q2, Q1, Q0] = {000001, 000010, 0001000, 001000, 010000, 100000} ciclicamente.

Contador em Anel:

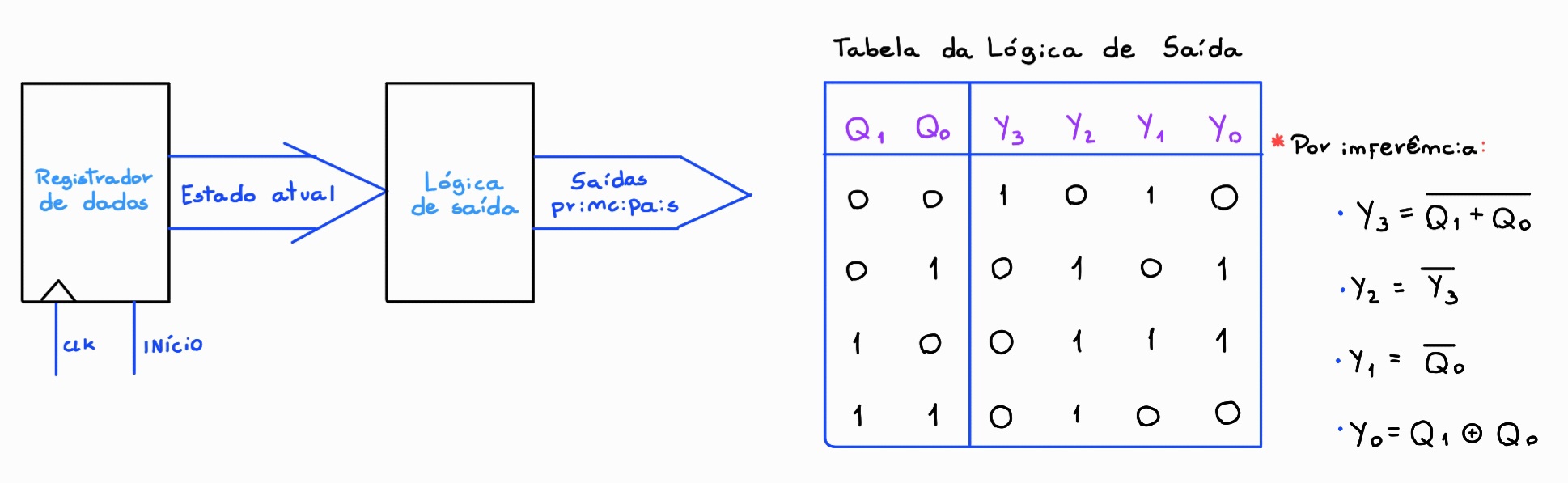


Implementação:

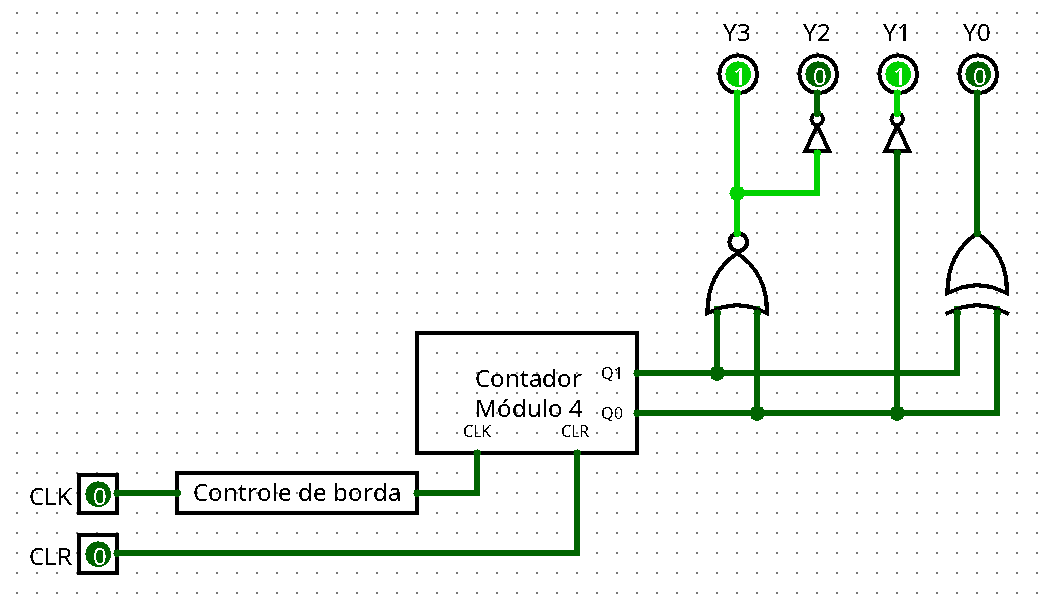


9) Implemente uma Máquina de Estados Finitos que realiza o seguinte diagrama de transição de estados: 10, 5, 7, 4, 10, 5, 7, 4, .…

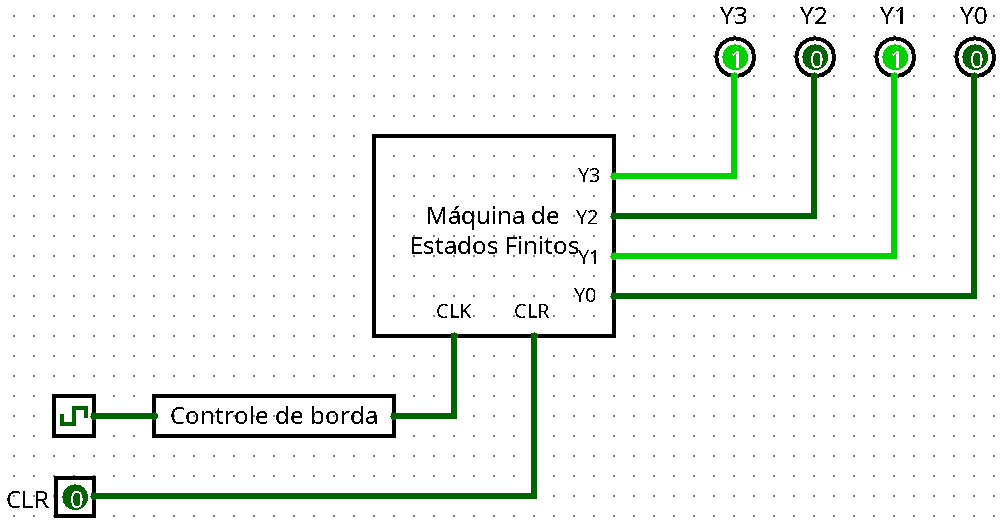
Cálculos:



Máquina de Estados Finitos:



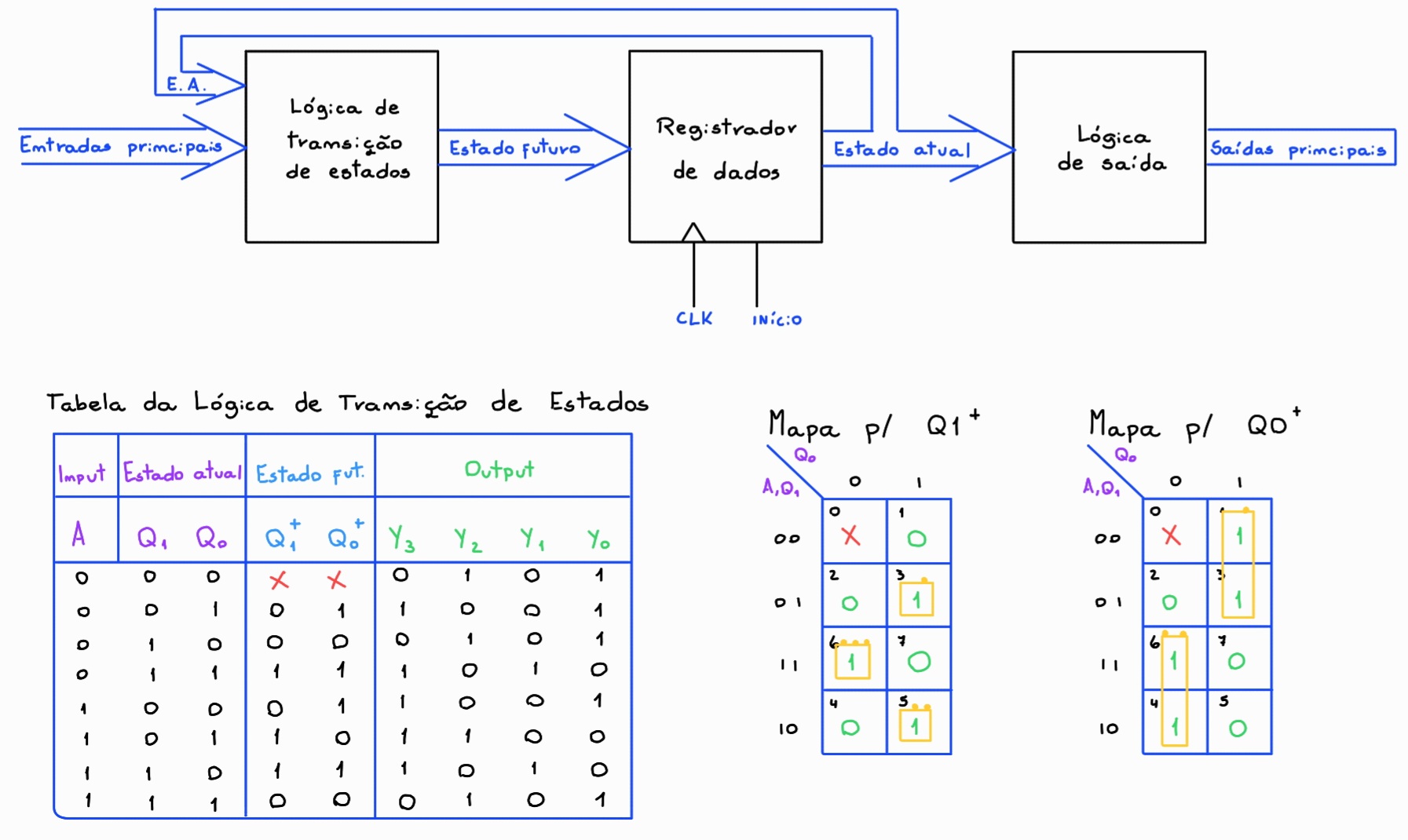
Implementação:

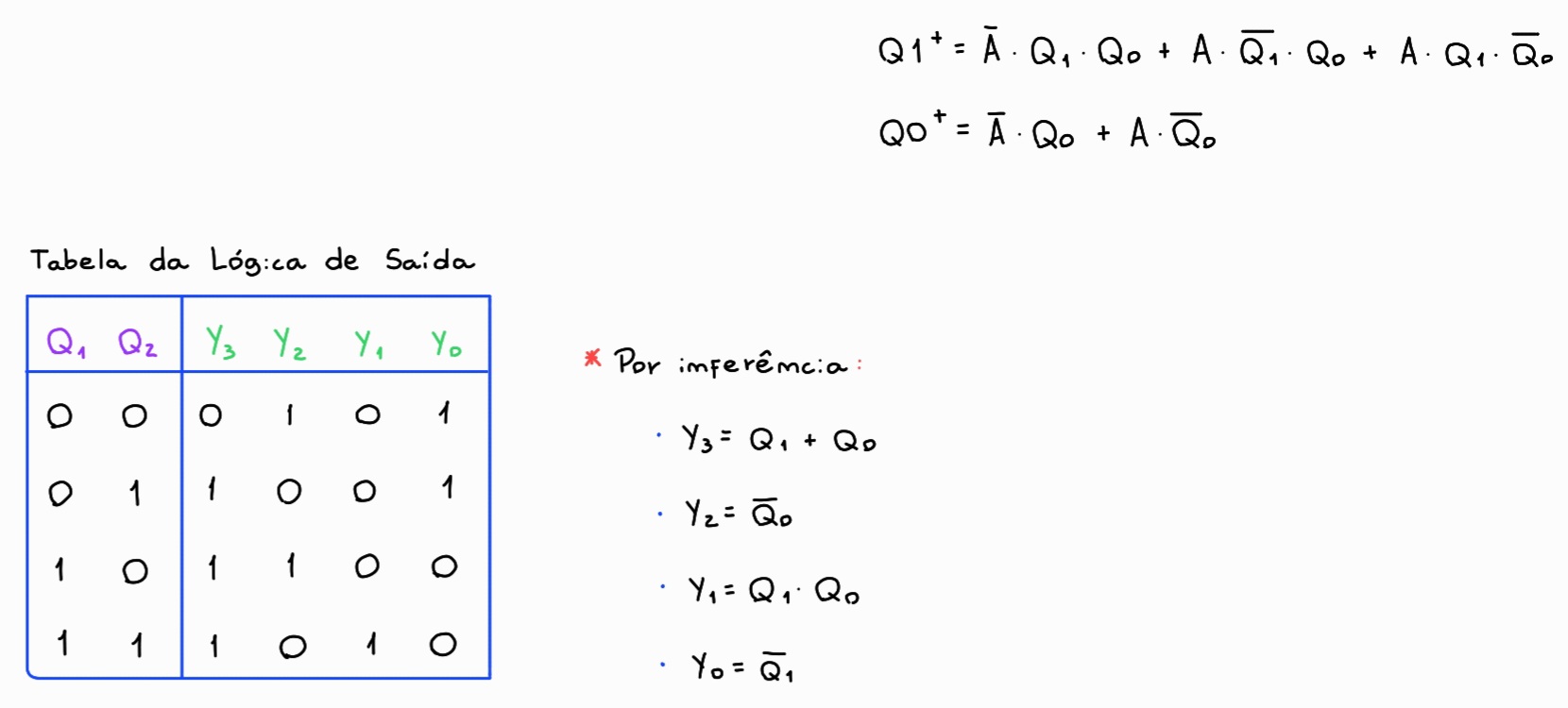


Descrição textual: A entrada CLR inicia a entrada com o número binário correspondente ao decimal 10. A cada ‘clock’, a contagem avança de acordo com a sequência estabelecida na Lógica de Saída.

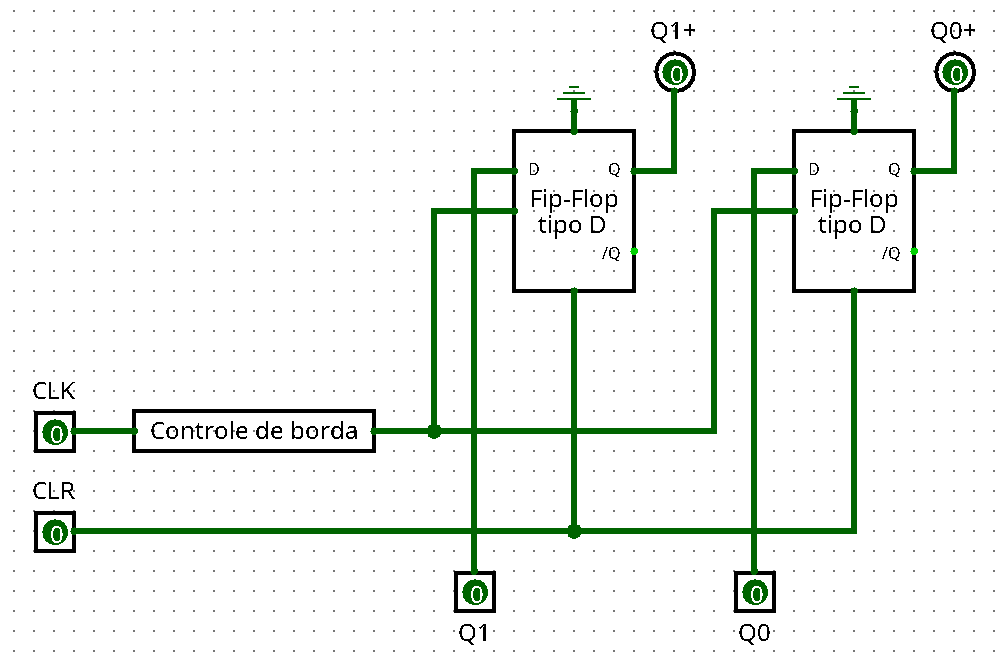
10) Implemente uma Máquina de Estados Finitos que realiza a seguinte transição de estados: 5, 9, 12, 5 (se A = 0), 10 (se A = 0), 5.

Cálculos:

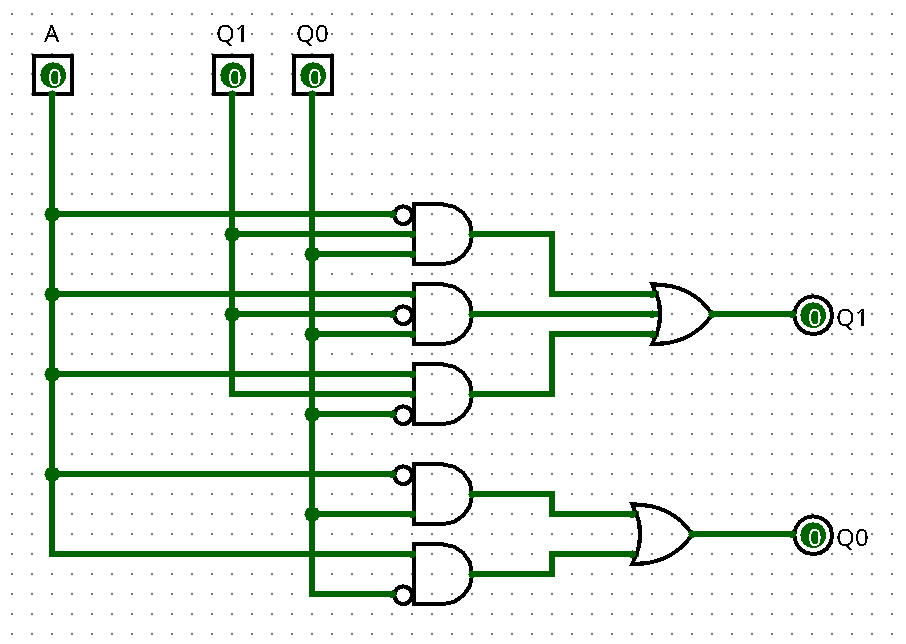




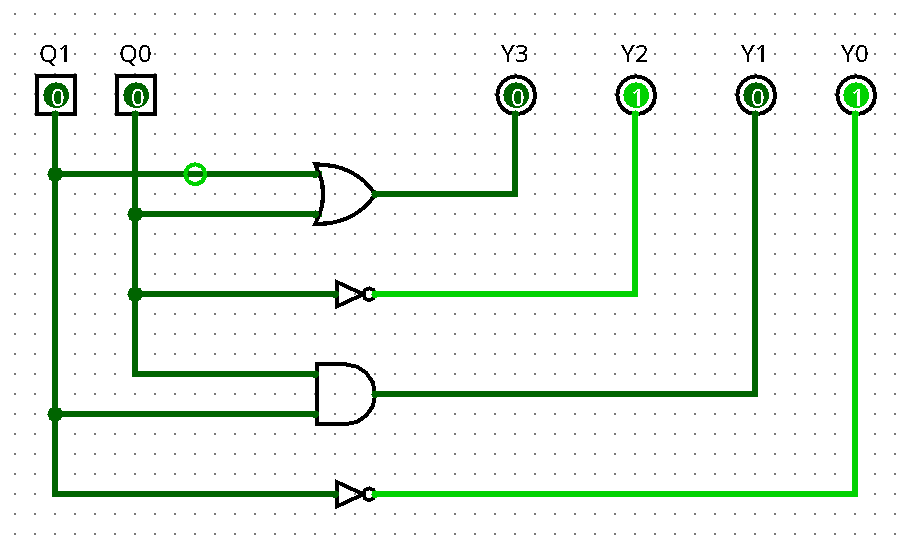
Registrador 2 bits:



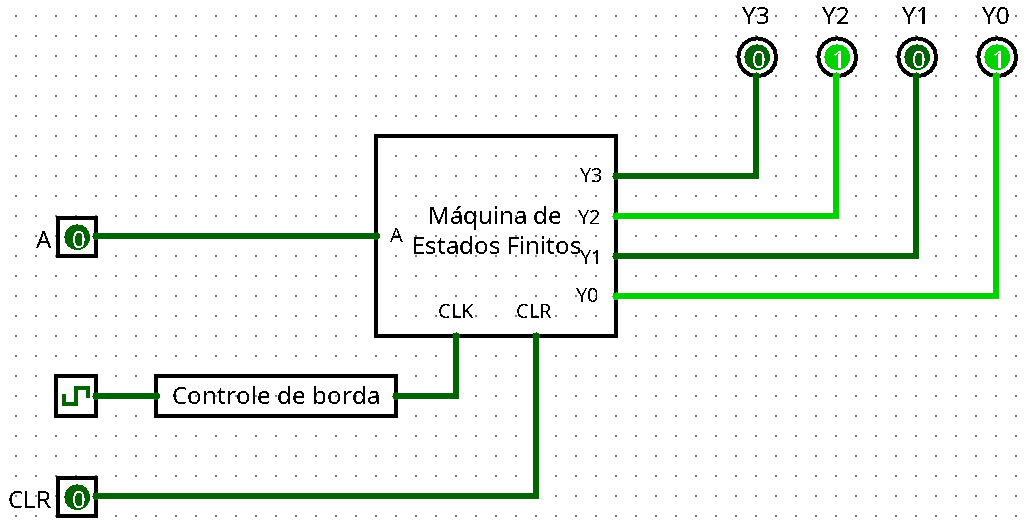
Lógica de Transição de Estados:



Lógica de Saída:



Implementação:



Descrição textual: A entrada CLR indica o início da contagem, carregando o número 5, em binário, para as saídas Y0 até Y3. A cada ‘clock’, a máquina avança um estado, a depender do valor da entrada A, e carrega o resultado para as saídas. Essas lógicas foram definidas nos blocos de Transição de Estados e de Saída.