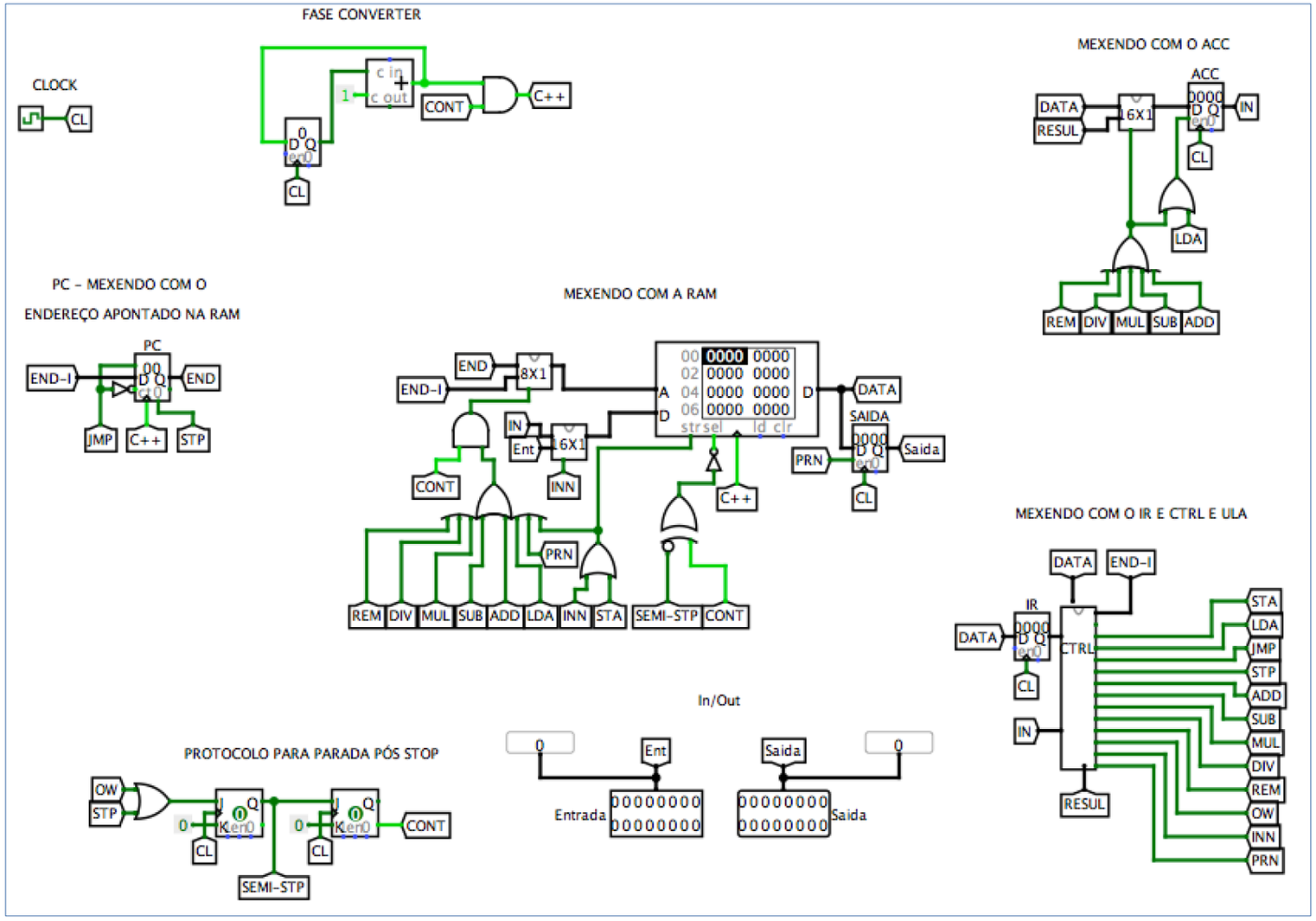
Apresentação EP3 - Implementação do HIPO

Álgebra Booleana - MAC0329Professor Junior Barrera

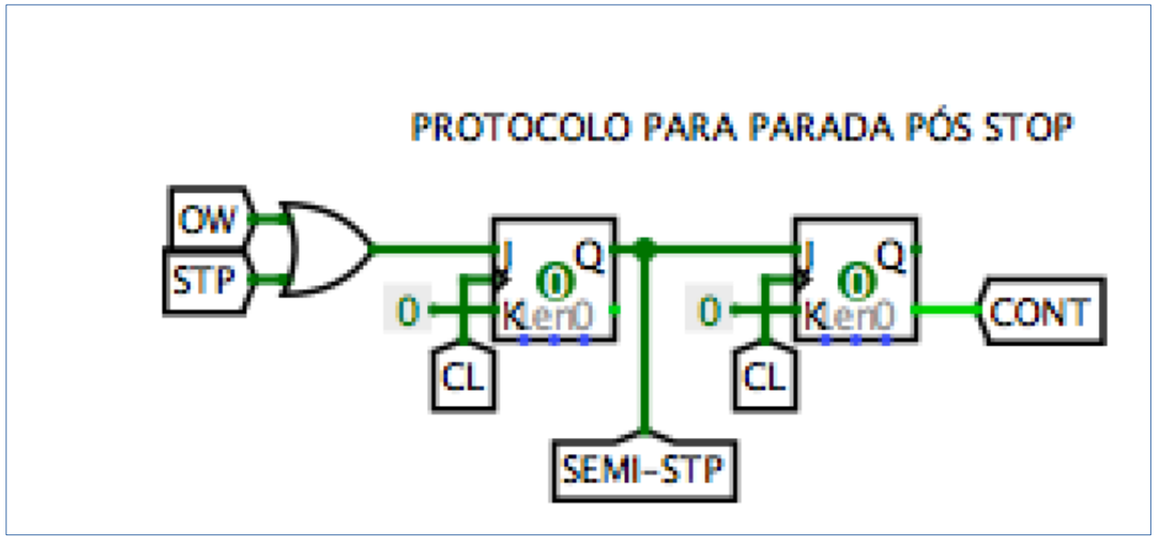
Rápida Apresentação da Main

- Programa Básico: Decidir se um número é primo.
- Modularização visual



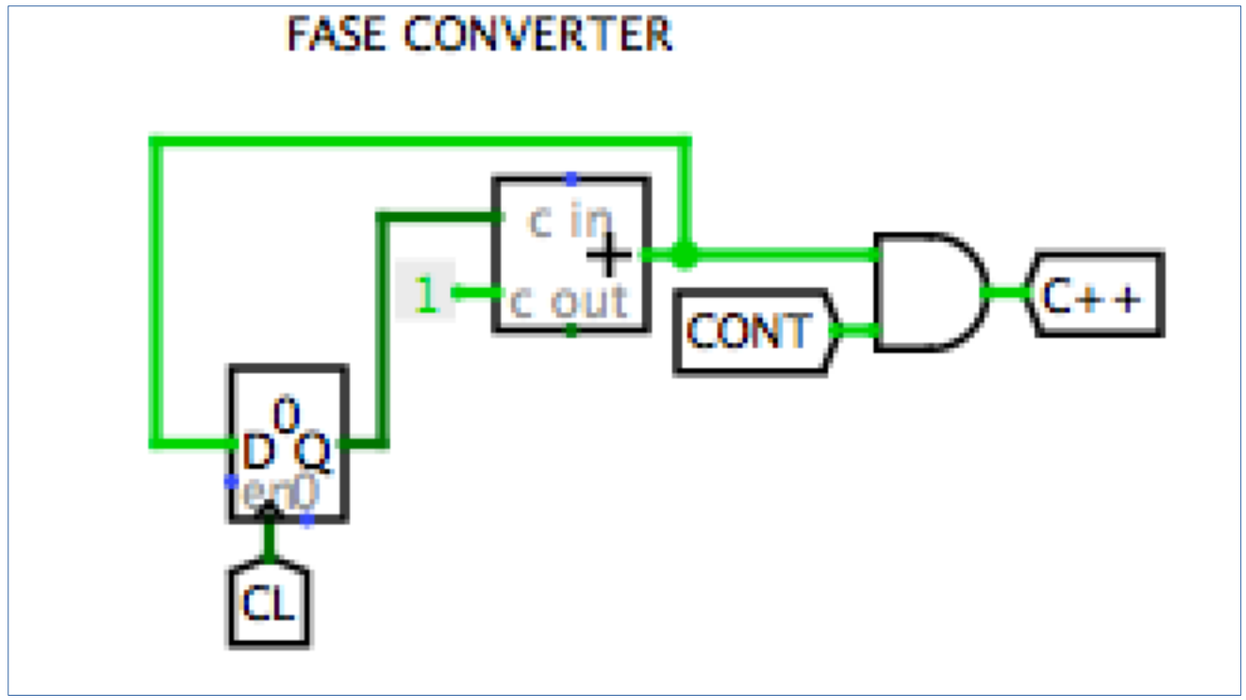
Parada

- Acontece após STP ou Overflow detectado.
- 2 tempos para retornar o PC a 00.
- SEMI-STP para garantir que a função no 00 não será executada durante a parada.



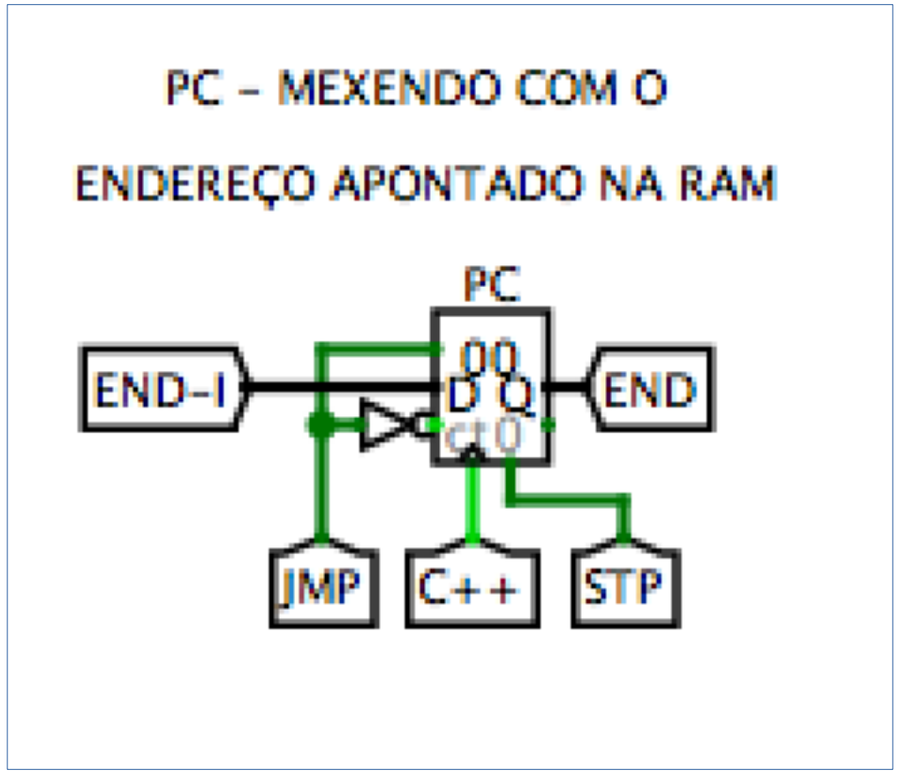
Clocks

- C++ é ligado a cada 2 subidas de clocks.
- Primeiro tempo para FETCH e DECODE
- Segundo tempo para EXECUTE (quando necessário)



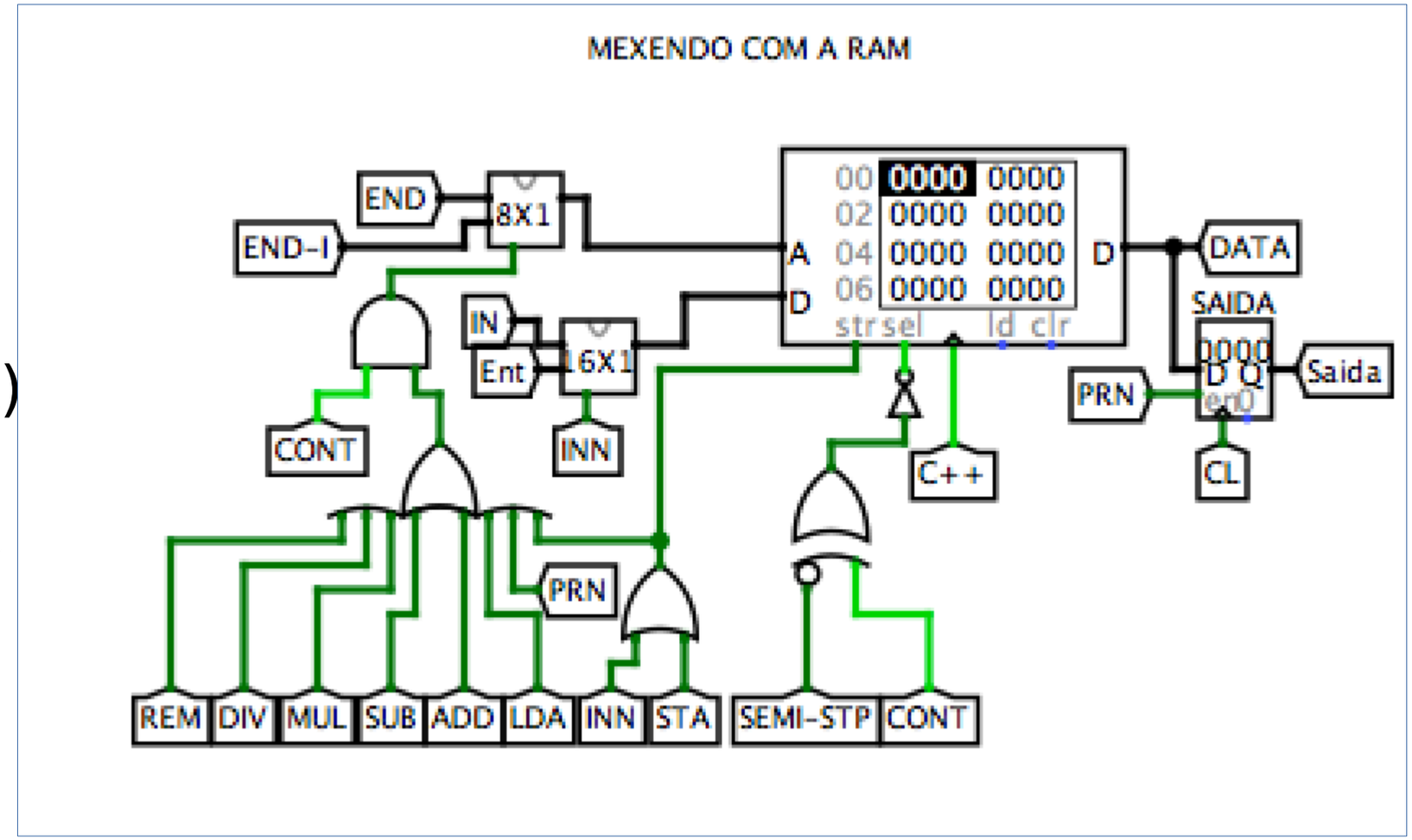
PC

- END é o conteúdo do PC, e indica a posição de leitura da RAM.
- END-I é o endereço de um possível JMP
- JMP indica que é preciso parar o incremento e dar load no END-I
- STP volta para o 0



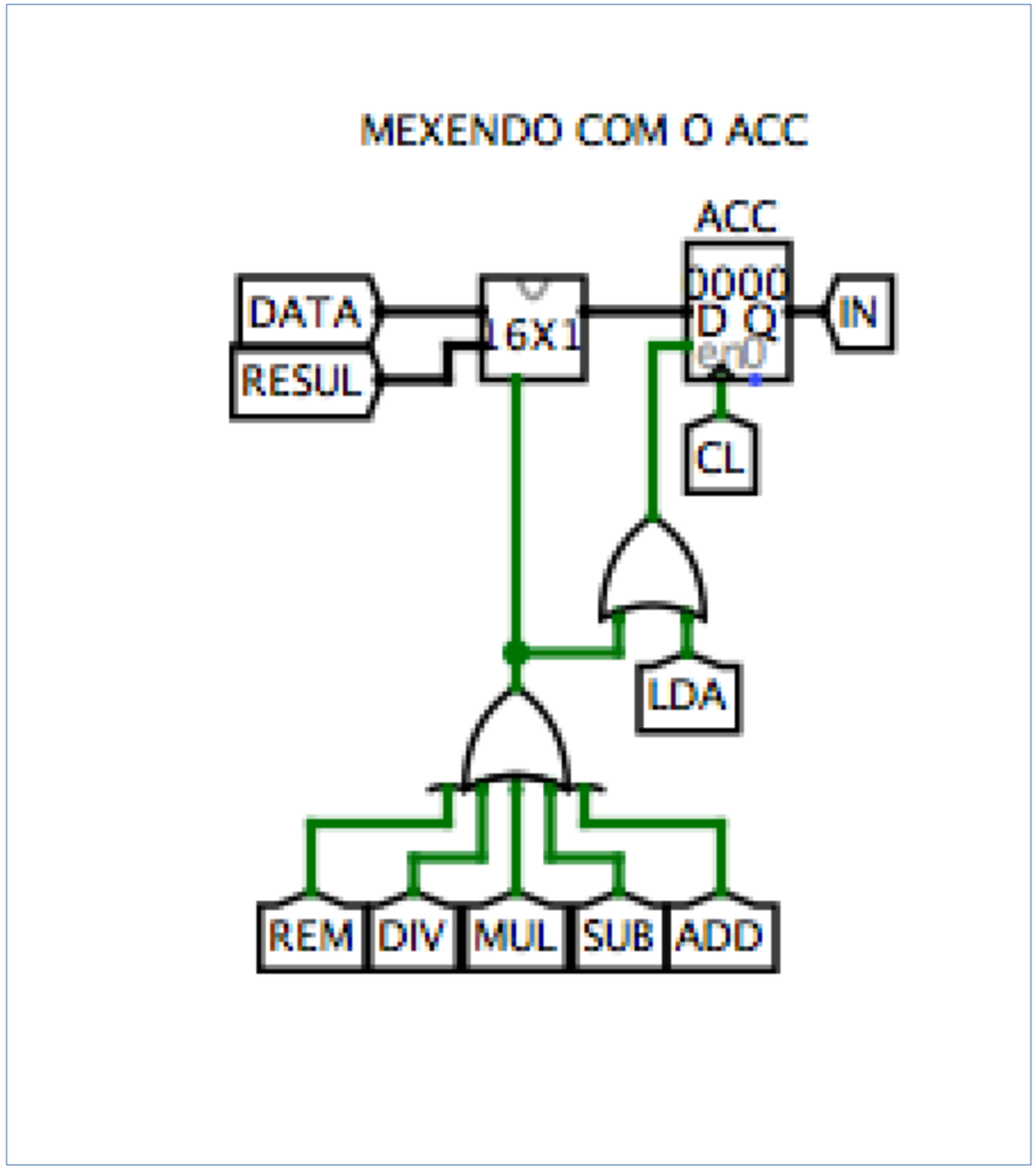
RAM

- END é o endereço apontado pelo PC. Indica em que ponto está a leitura do programa.
- Caso haja alguma operação (como LDA) que necessita olhar para outro endereço, END-I o indica e é “ativado”
- SEMI-STP e CONT garantem parada
- DATA é o conteúdo do que está sendo apontado na RAM. (assíncrono)



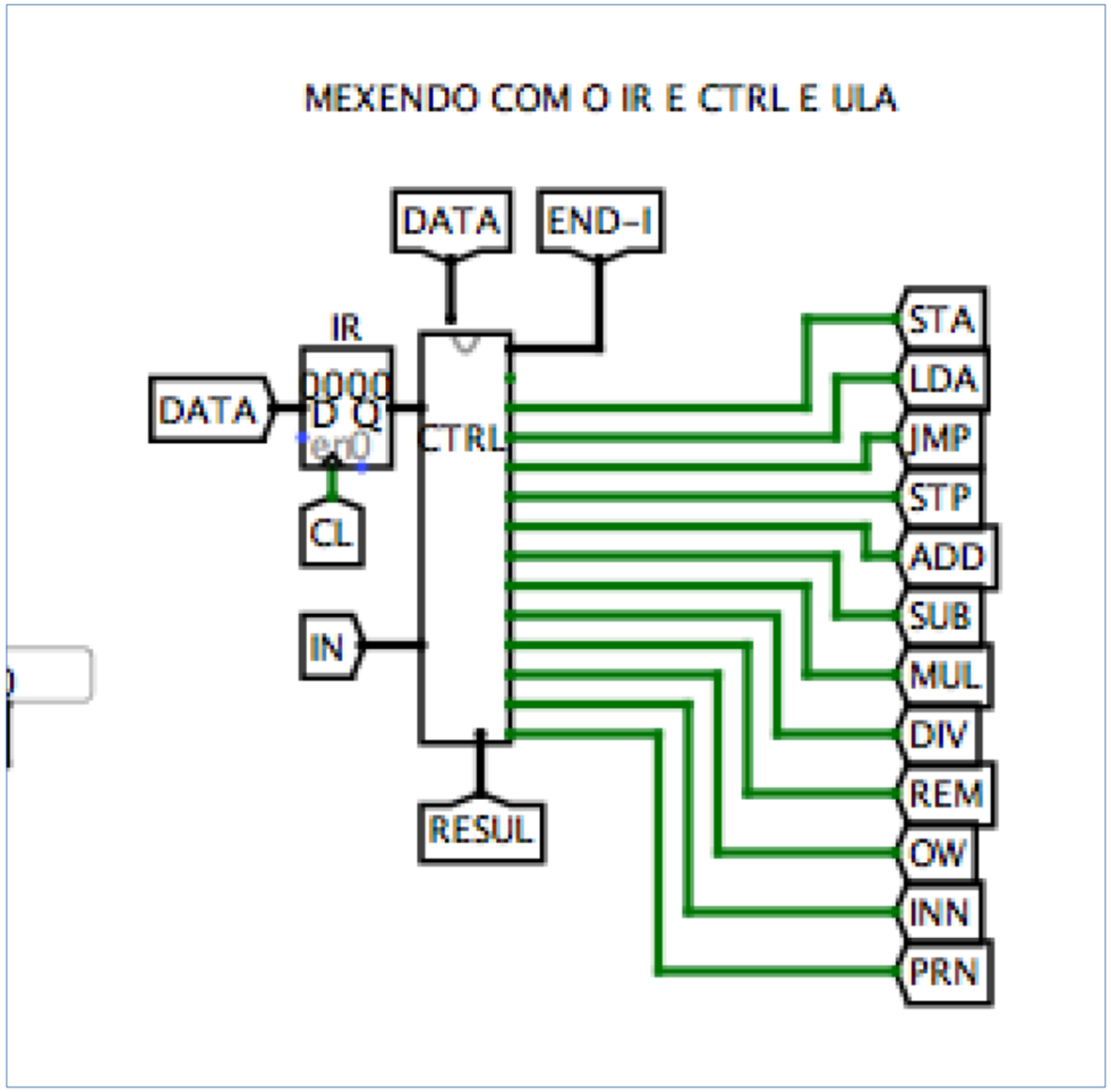
Acumulador

- O conteúdo do ACC só muda caso haja operação aritmética ou LDA.
- RESUL é o que entra no primeiro caso, DATA no segundo.
- IN é o conteúdo do ACC (assíncrono).



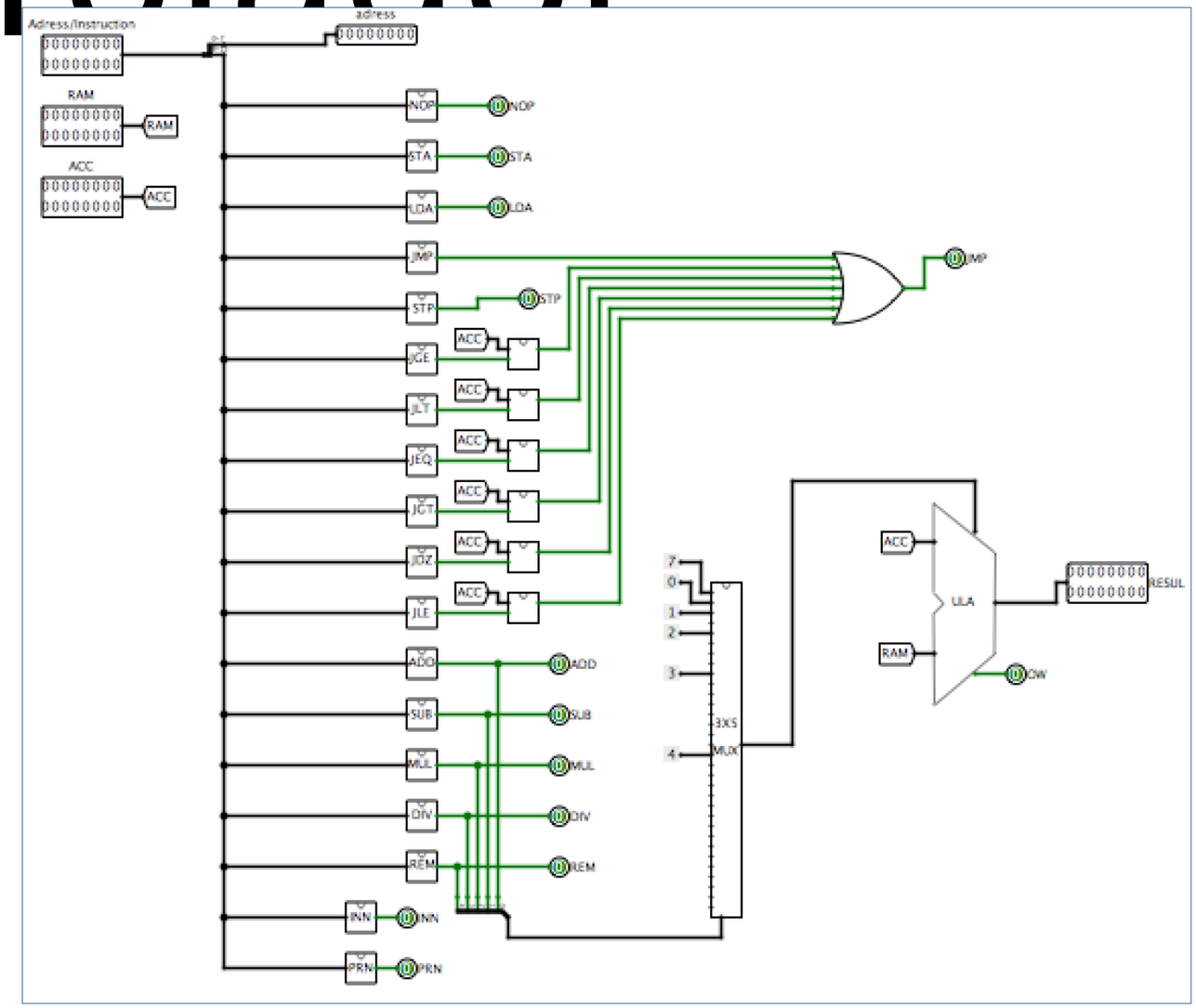
IR e Controlador

- IR é o conteúdo que está na RAM. Atualizado com o CL, e não com C++ (um passo atrás).
- END-I é o endereço passado pela instrução atual.
- RESUL é o resultado da operação aritmética, se houver.
- Indicadores de operações.



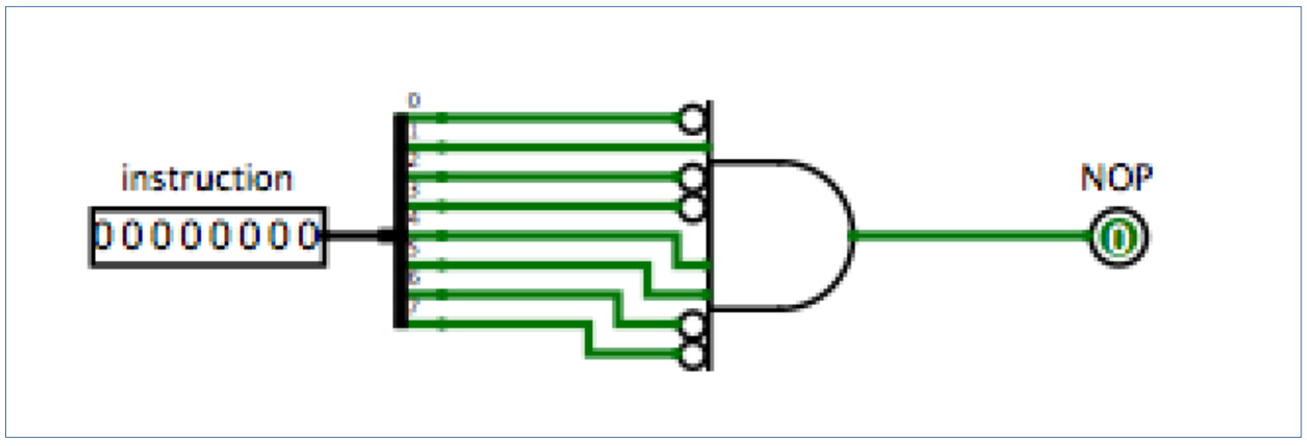
Dentro do Controlador

- O conteúdo recebido do IR é dividido entre endereço (acima) e operação (identificada abaixo).
- A ALU fica dentro do controlador



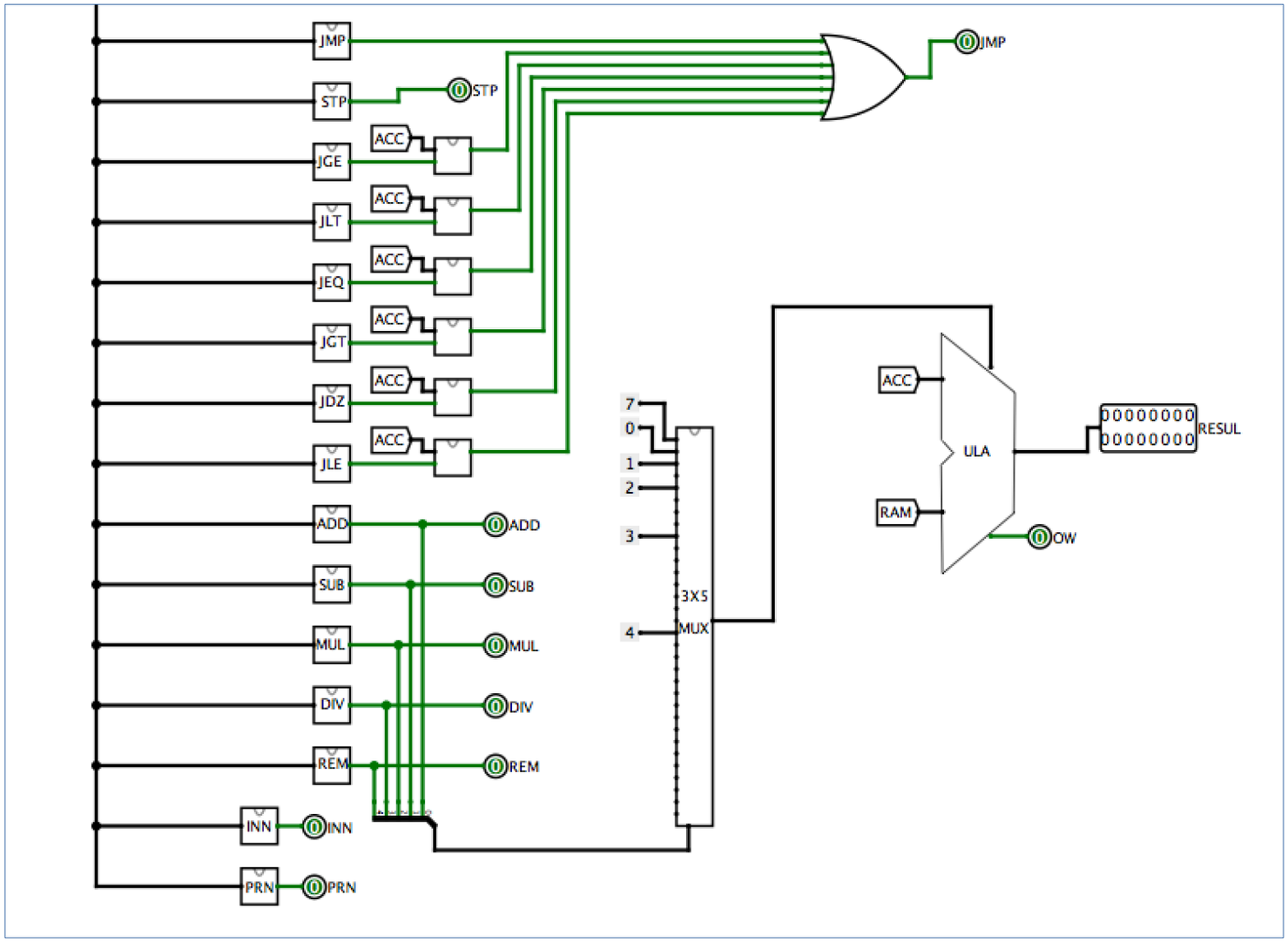
Identificadores de operação

- Simples circuito combinatório.
- Age como um filtro.



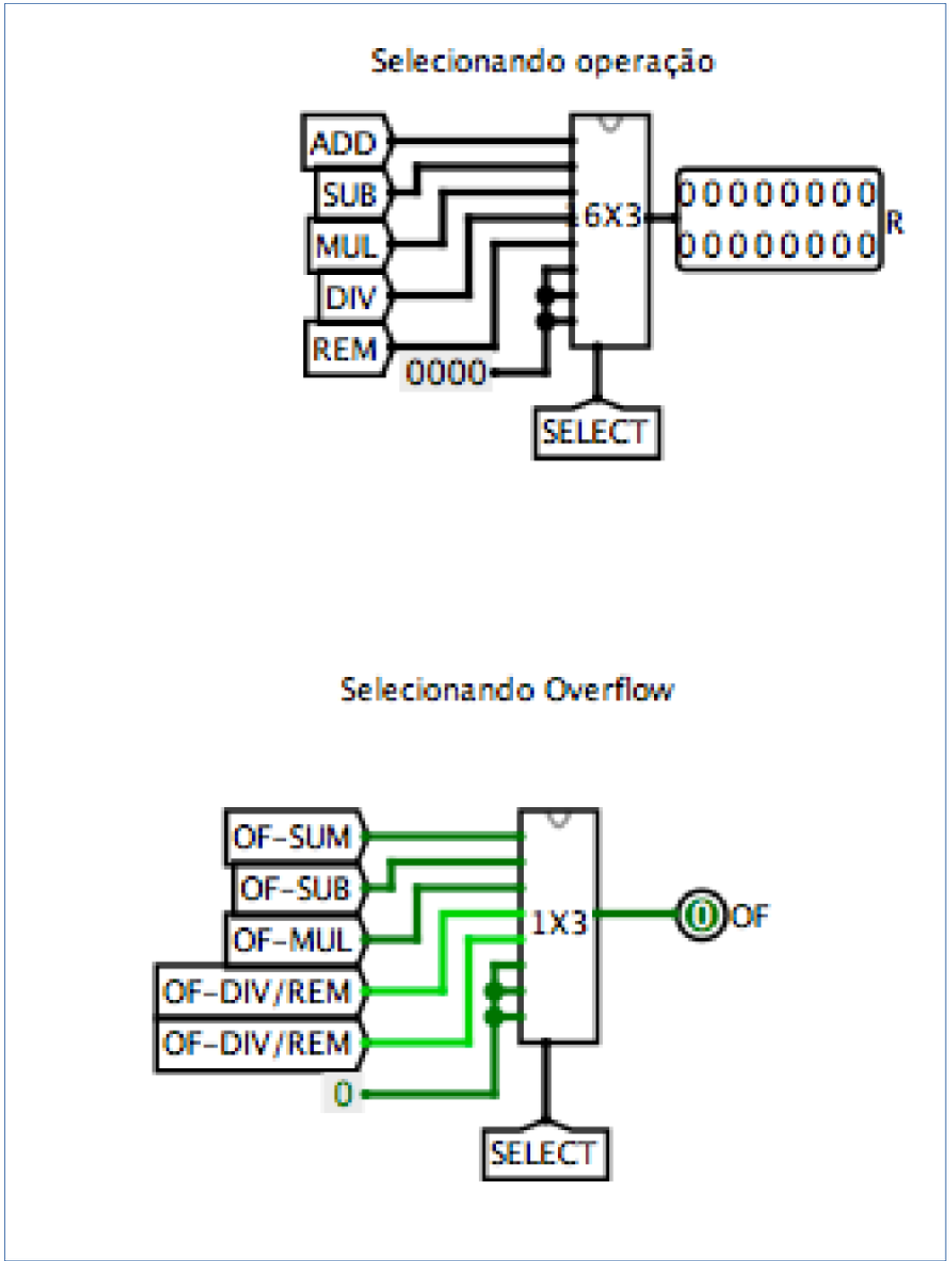
Jumps e ULA

- O jump condicional se resume a um filtro superior para acender a operação de JMP.
- A identificação das operações aritméticas passa por um multiplexador para se adaptar ao controle da ALU (fase 1).



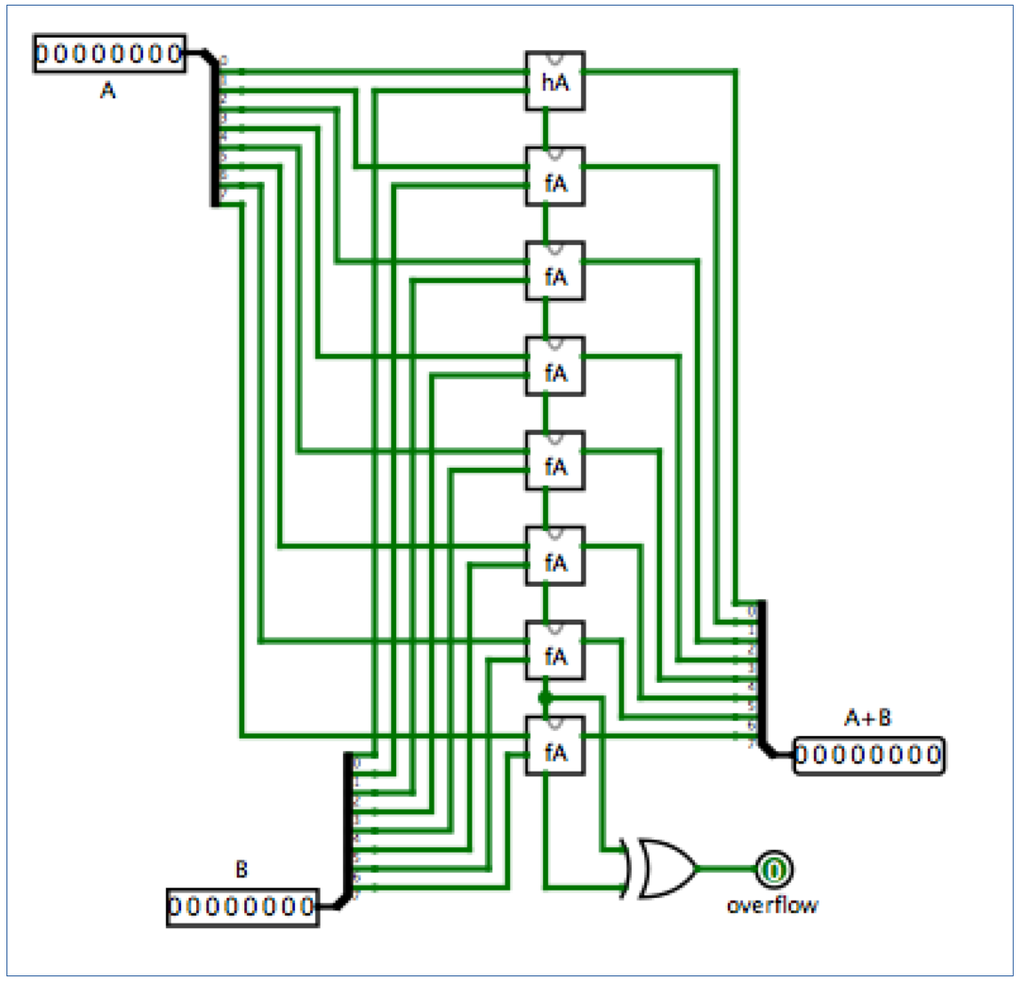
Dentro da ULA

- Multiplexadores 16x3 e 1x3 para selecionar resultado e possível overflow.
- Operações com circuito combinatório (fase 1).



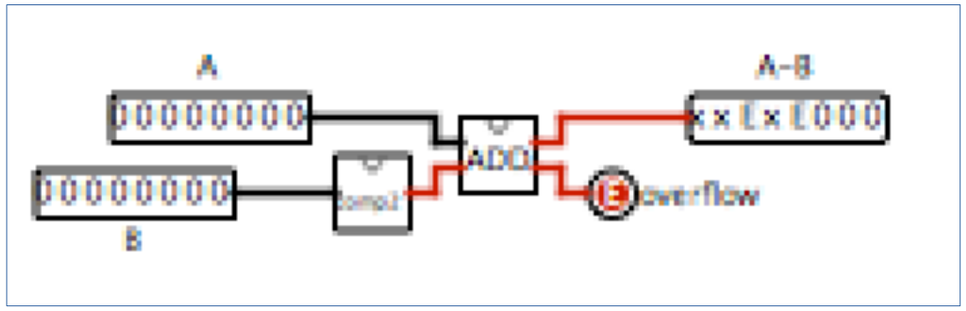
Adição

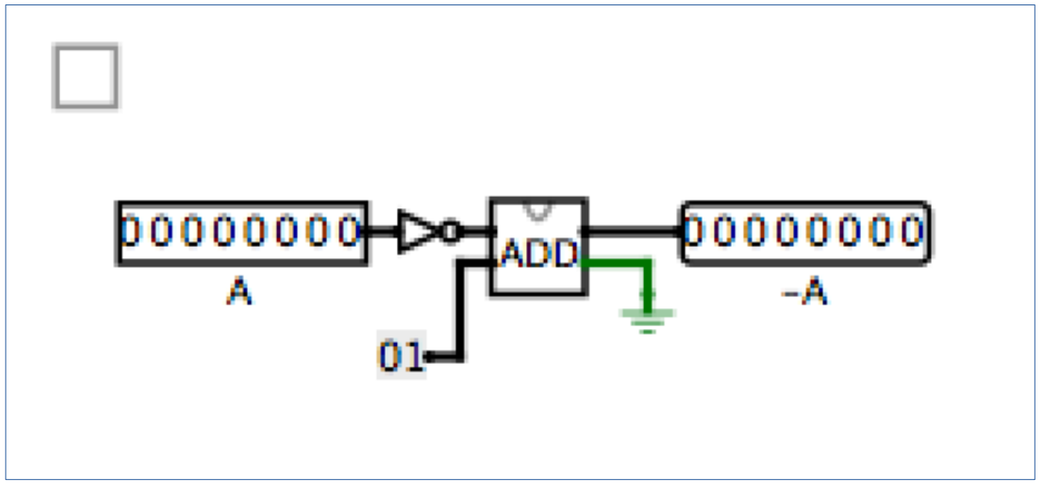
- XOR entre Cout e Cin do último fullAdder decide overflow.



Subtração

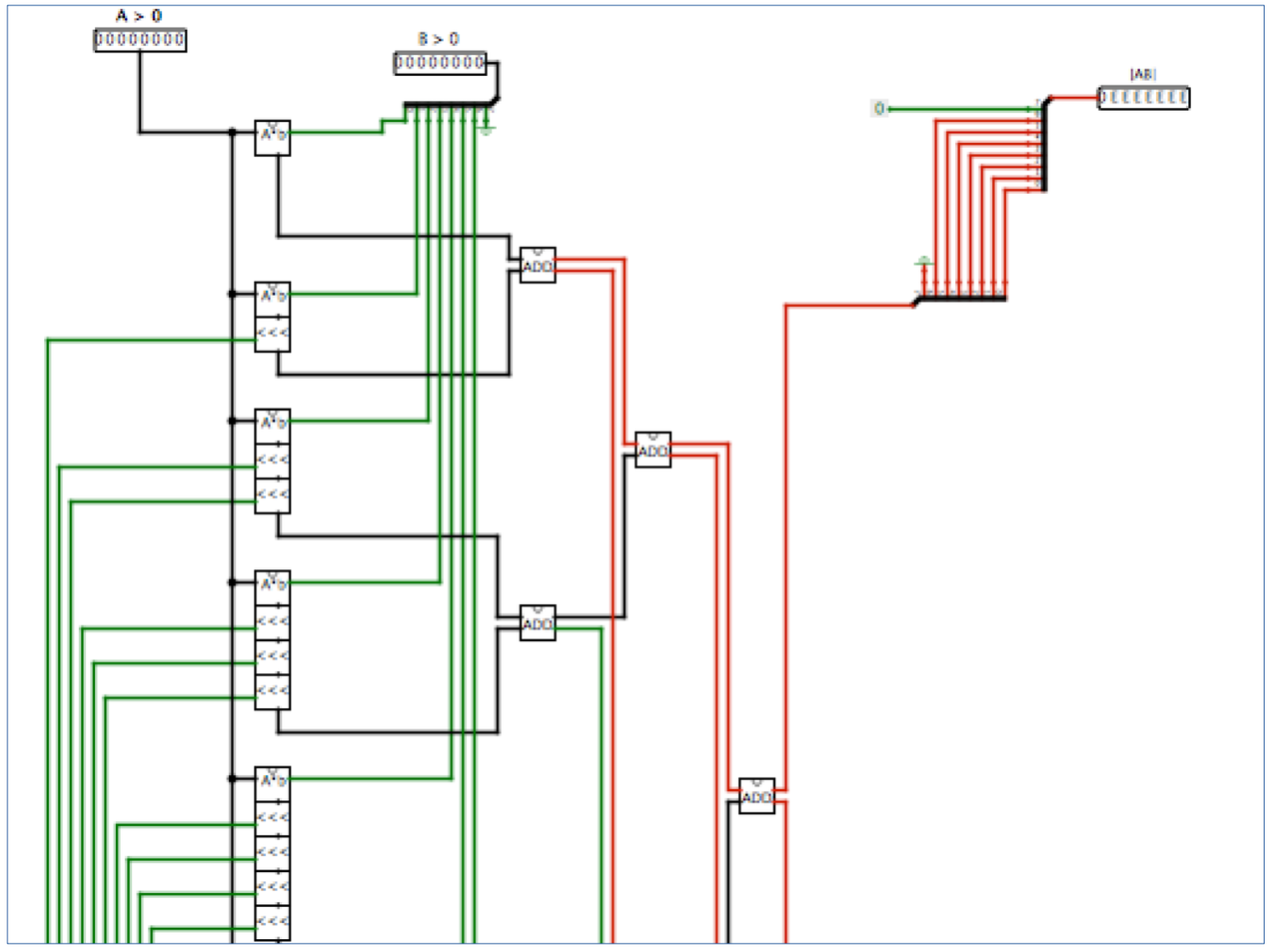
- Fácil adaptação da subtração
- Utiliza complemento de 2 para negar.





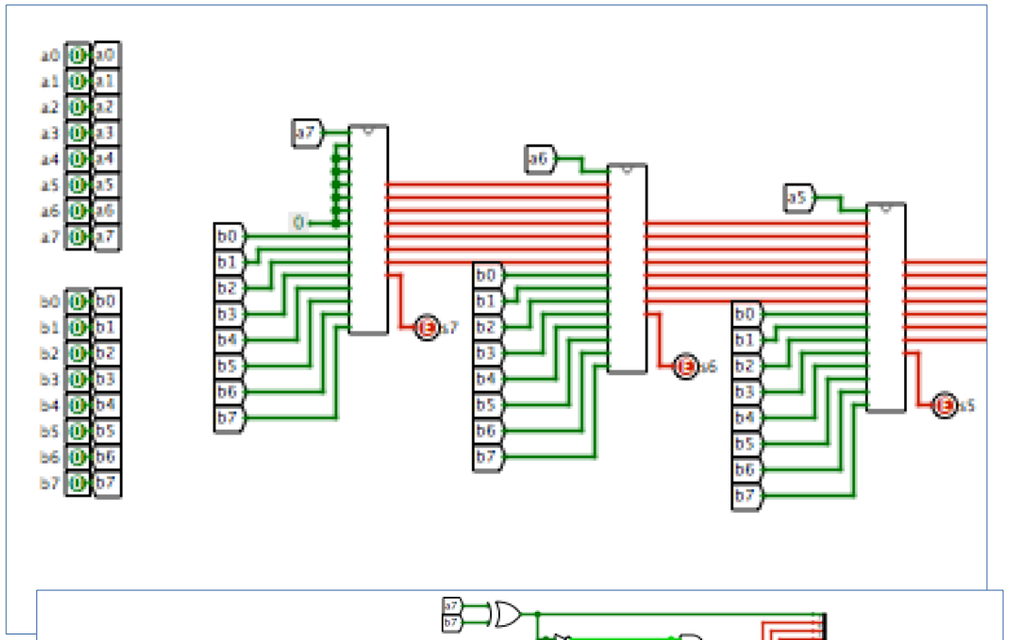
Multiplicação

- Como aprendemos no colégio.
- Multiplicar por um bit e depois left-shift.
- Tratamos o sinal separadamente.



Divisão e Resto

- Como aprendemos no colégio.
- Multiplicar por um bit e depois left-shift.
- Tratamos o sinal separadamente.



Obrigado pela atenção

- João Francisco Daniel - 7578279
- Mateus Anjos - 9298191
- Matheus Oliveira - 8642821
- Victor Sprengel - 9298002