PUCRS - PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ENGENHARIA DE SOFTWARE

RELATÓRIO DO TRABALHO 1 - ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

ARTHUR SUDBRACK IBARRA, GUSTAVO GIONGO LOTTERMANN E KEVIN RIBAS

PORTO ALEGRE
2021

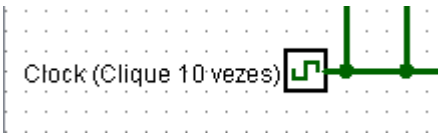
1. *Introdução*

Para este trabalho foi-se solicitado que construíssemos uma ULA sequencial (Unidade Lógica Aritmética), a qual deveria ser responsável por realizar operações de AND, OR, NOT ou somas aritméticas entre números binários de cinco bits. No entanto, sabia-se que deveríamos construir nossos componentes, como os seletores ou o somador, de forma manual, ou seja, sem o auxílio de componentes já prontos e disponibilizados pela ferramenta de construção de circuitos utilizada em aula (com a exceção de flip-flops, responsáveis pelas partes de memória). Ao decorrer deste relatório, será constatada a solução que o grupo encontrou para o desenvolvimento do trabalho, além de uma detalhada explicação a respeito de como o circuito confeccionado funciona, as partes que o compõe e exemplos de sua aplicação.

2. *Desenvolvimento*

2.1. Fragmentos do circuito

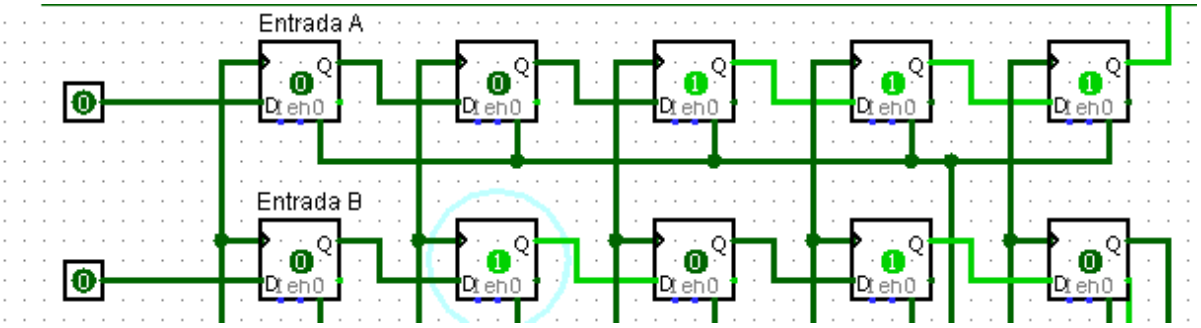
Clock



O clock é um dos componentes mais importantes do circuito, pois ele é responsável pelo deslocamento dos valores contidos em nossos flip-flops. Os flip-flops, por sua vez, são compostos por uma série de portas lógicas, de modo que, a cada sinal que eles recebem, emitem o valor e o inverso (complemento) do valor que estava anteriormente armazenado neles.

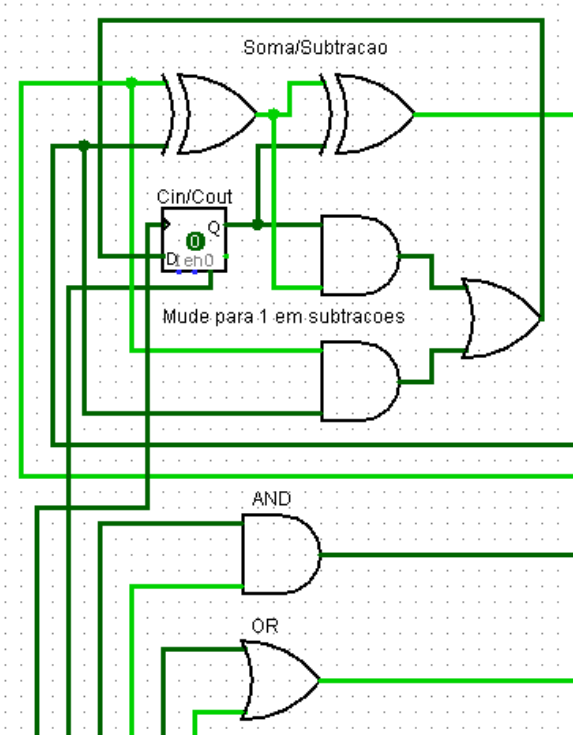
O que foi esclarecido no parágrafo acima é essencial para que possamos compreender o porquê de precisarmos, conforme a imagem sugere, clicar no clock 10 vezes. Sabemos que o circuito em questão é sensível à borda de subida, ou seja, de forma simplificada e em especial para esse circuito, nossos flip-flops somente deslocam seus valores para o próximo quando o clock performa um sinal de subida, e, portanto, como estamos trabalhando com números de cinco bits, são necessárias 10 oscilações do clock, uma vez que cinco dessas oscilações serão desprezadas.

Entradas



Nas entradas A e B são informados os números binários de cinco bits com os quais queremos trabalhar. Conforme oscilamos o nosso clock, esses números irão sendo deslocados para a direita para que possamos realizar as operações de nossa ULA com um bit de A e outro bit de B a cada borda de subida do clock. Para a imagem acima, por instância, teríamos as entradas: A = 00111, B = 01010.

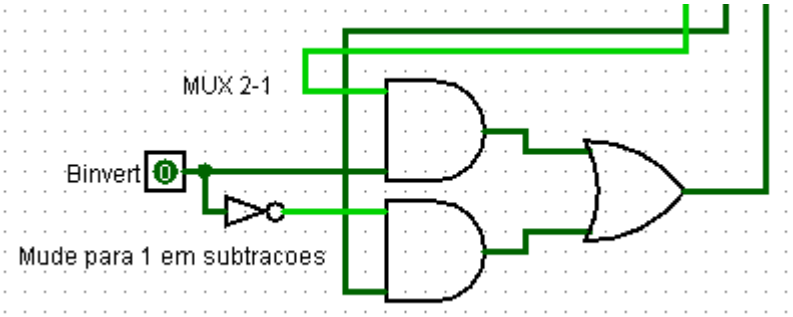
Operações do ULA



Nesta etapa do circuito os bits das entradas A e B, mencionadas anteriormente, passam pelas portas lógicas mostradas na imagem acima, e, desta forma, obtemos simultaneamente um resultado proveniente de todas as operações que constituem a ULA (resultado de OR, de AND e da soma aritmética). É

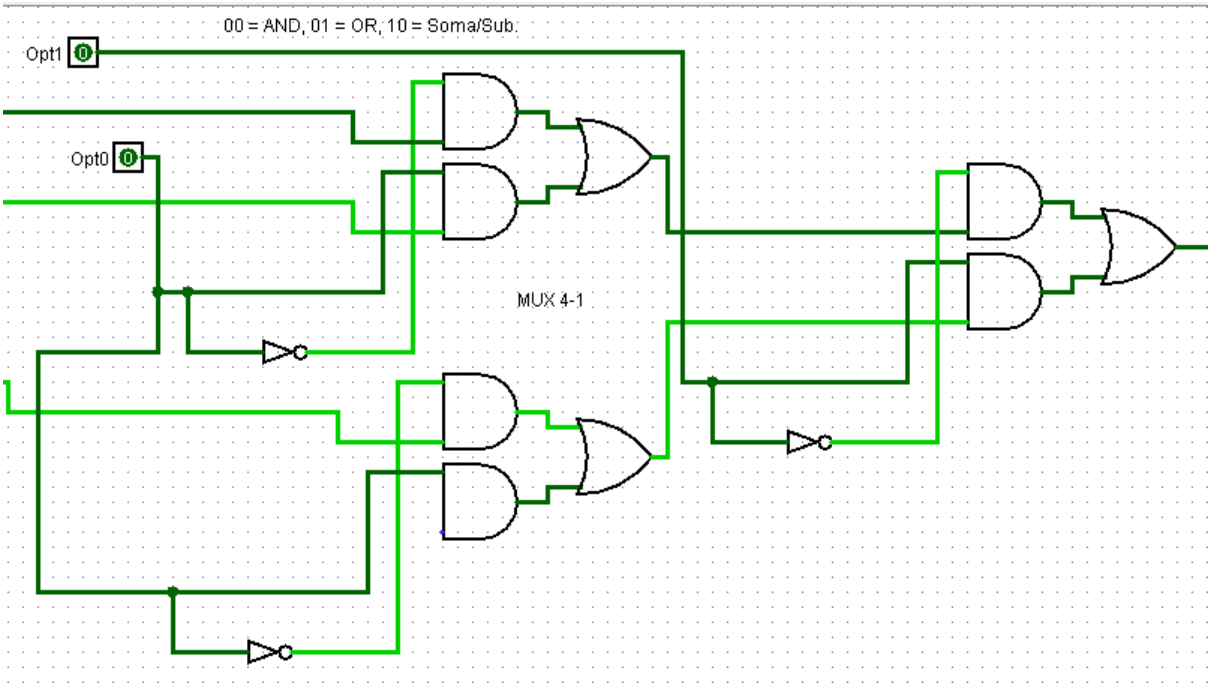
importante ressaltar que, para guardar os excedentes do cálculo da soma, estamos utilizando o mecanismo de flip-flops tipo D - por esses mecanismos serem um recurso de memória, é possível resgatar o excedente de uma soma prévia e somá-lo com os próximos bits das entradas.

MUX 2-1 (Binvert)



Para a implementação do Binvert foi necessário construir um MUX 2-1, este componente é um seletor que possui a funcionalidade de escolher se usaremos os bits de B da forma que eles foram originalmente postos na entrada ou se usaremos o inverso (NOT) deles. O Binvert, como dito, é responsável por inverter todos os bits que compõem a entrada B e é ativado somente para realizarmos subtrações em nosso ULA. Isso se deve porque, para realizarmos operações de subtração, estamos somando a entrada A com o complemento de 2 da entrada B, e, para obtermos esse complemento, o primeiro passo necessário é a inversão completa de B - para isso existe o Binvert. O segundo passo é somar 1 bit ao valor obtido após a inversão de B, no entanto, isto ainda não é feito no componente do circuito mostrado na imagem acima, veremos como cumprir esse segundo passo mais adiante em nosso relatório.

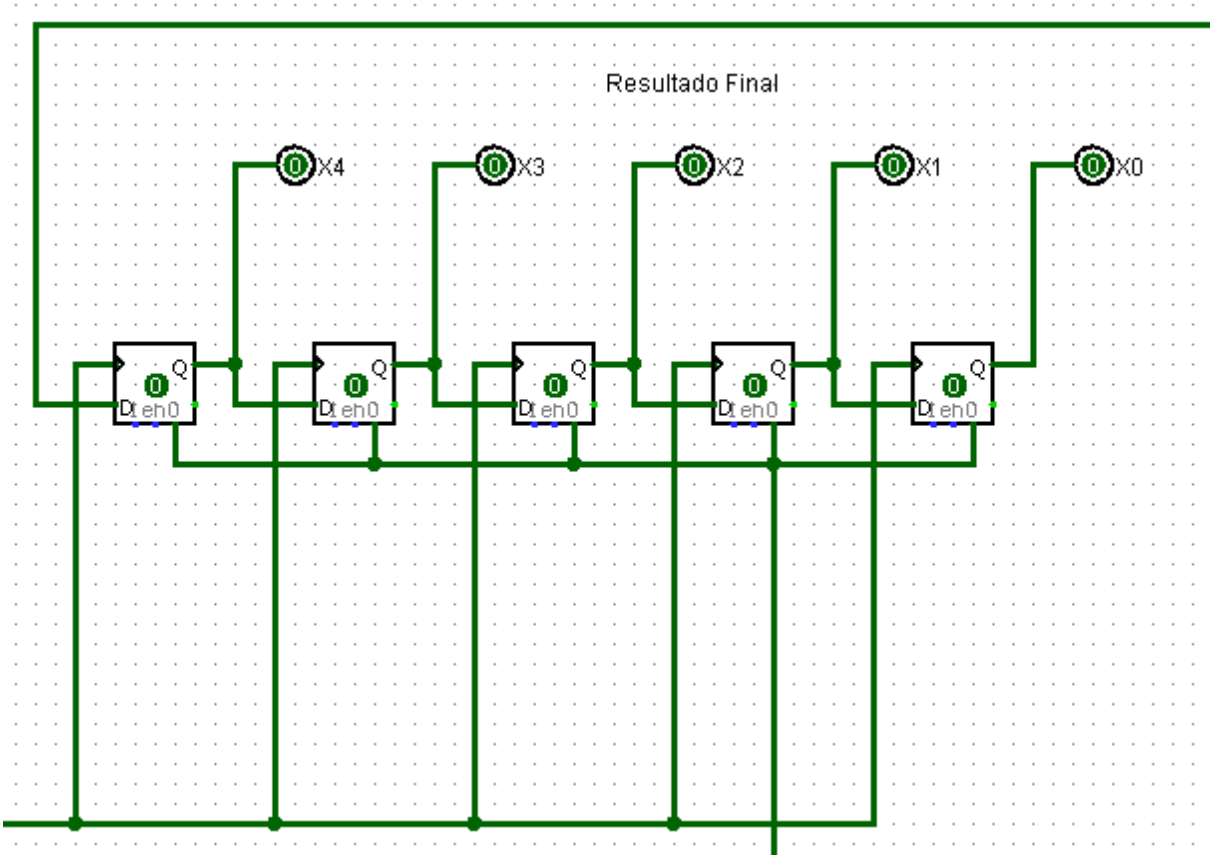
MUX 4-1 (escolha das operações)



O MUX 4-1 da imagem acima é composto por três MUX 2-1 e é utilizado exclusivamente para escolhermos se desejamos realizar a operação de AND, de OR ou de soma em nosso ULA. Os pinos Opt0 e Opt1 juntos são responsáveis por indicar isso em números binários, conforme podemos ver na tabela abaixo:

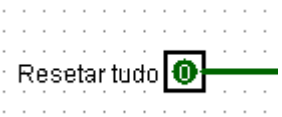
Combinação	Valor em Decimal	Operação Correspondente
Opt1 = 0 e Opt0 = 0	0	AND
Opt1 = 0 e Opt0 = 1	1	OR
Opt1 = 1 e Opt0 = 0	2	Soma
Opt1 = 1 e Opt0 = 1	3	Não está sendo utilizado pela ULA em questão

Resultado



Esta parte do circuito é composta por cinco flip-flops tipo D e é responsável por armazenar os resultados obtidos para as operações do ULA durante as oscilações do clock. A cada borda de subida do clock, os bits presentes nos flips-flops vão sendo deslocados para a direita, de forma a gradativamente ir compondo o resultado final, o qual é obtido após todas as 10 bordas de subida do clock necessárias para o funcionamento deste circuito.

Resetador



O resetador é unicamente responsável por redefinir os valores contidos em todos os flip-flops do circuito para 0. Ele pode ser usado quando desejamos realizar outra operação do ULA com novas entradas sem que as operações que foram realizadas anteriormente afetem este processo.

2.2 Explicação do circuito

Agora que possuímos uma melhor compreensão a respeito dos fragmentos que compõem nosso circuito, será descrito como este funciona em um fluxo completo, englobando todas as partes que já foram aqui mostradas:

Primeiramente definimos as entradas A e B com os valores binários de cinco bits que desejamos em nossos flip-flops de entrada e configuramos os pinos Opt1 e Opt0 em nosso MUX 4-1 de modo a escolher qual operação do ULA iremos realizar. Após isso, podemos começar a oscilar nosso clock, sendo que, a cada borda de subida, os bits das entradas de A e B* irão sendo deslocados e processados pelas portas ou conjunto de portas tradicionais do ULA (AND, OR e somador). Dessa forma, todas elas irão emitir algum resultado proveniente da operação realizada - a porta AND pode emitir 0, enquanto que a porta OR e o somador podem emitir 1. Posto isso, cabe ao nosso seletor 4x1, ou MUX 4-1, decidir se iremos considerar o resultado proveniente da operação de AND, de OR ou de soma, o que depende diretamente do que configuramos para os pinos Opt1 e Opt0, conforme dito anteriormente. Uma vez tendo escolhido o que iremos considerar, os valores retornados pelas operações do ULA irão sendo encaminhados para nossos flip-flops de resultado, os quais serão gradativamente preenchidos conforme oscilamos o clock.

* É importante reconhecer que, caso o Binvert esteja configurado como 1, então serão utilizados os bits de B em formato invertido nas operações do ULA. É nesta parte do ULA que ocorre a operação de NOT, a qual, embora não apareça como uma opção de resultado final, é substancial para montarmos o complemento de 2 da entrada B em subtrações.

2.3 Aplicações

2.3.1 Casos AND e OR

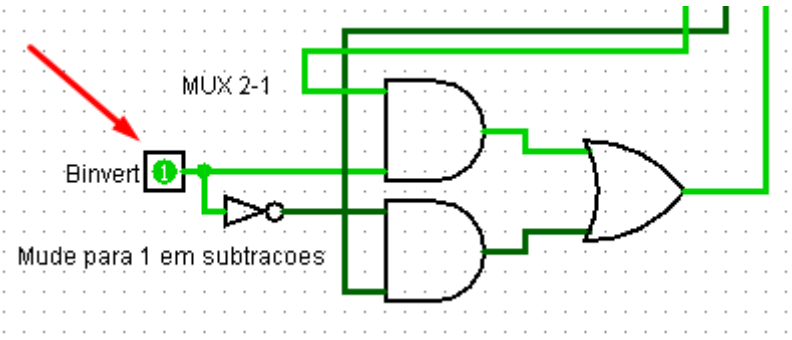
Ent. A	Ent. B	Binvert	Opt1	Opt0	Operação	Resultado
01101	01000	0	0	0	AND	01000
01101	01000	0	0	1	OR	01101
01010	11100	0	0	0	AND	01000
01010	11100	0	0	1	OR	11110
11111	00111	0	0	0	AND	00111
11111	00111	0	0	1	OR	11111
10001	01110	0	0	0	AND	00000
10001	01110	0	0	1	OR	11111
10110	00001	0	0	0	AND	00000
10110	00001	0	0	1	OR	10111

2.3.2 Casos soma

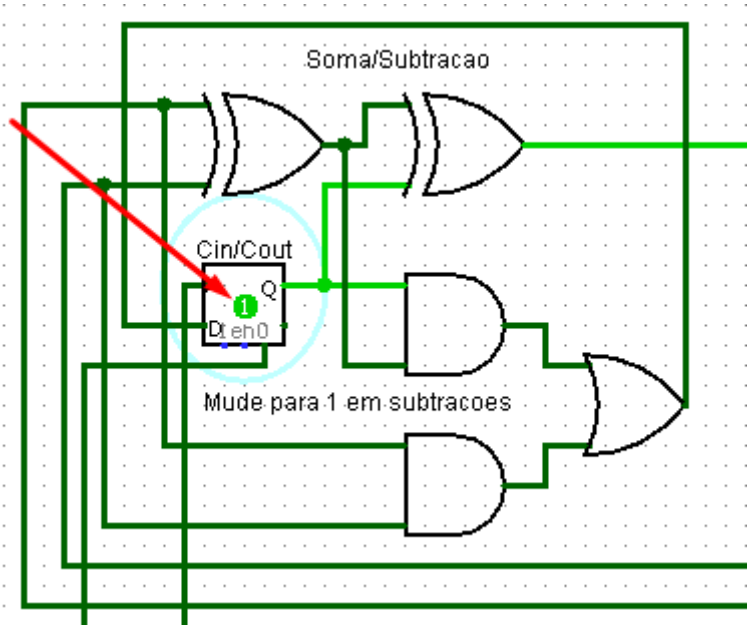
Ent. A	Ent. B	Binvert	Opt1	Opt0	Operação	Resultado	Cout Final
01101	01000	0	1	0	Soma	10101	0
01010	11100	0	1	0	Soma	00110	1
11111	00111	0	1	0	Soma	00110	1
10001	01110	0	1	0	Soma	11111	0
10110	00001	0	1	0	Soma	10111	0

2.3.3 Casos subtração

Para os casos de subtração, é importante ressaltar que algumas configurações adicionais têm de ser feitas, sendo estas a ativação de Binvert para 1 e a definição manual do valor inicial do flip-flop responsável pelos excedentes da soma para 1.



Ativação do Binvert (Operação NOT de B)



Definindo o valor inicial do flip-flop de excedentes para 1

Ent. A	Ent. B	Binvert	Opt1	Opt0	Operação	Resultado	Cout Final
01101	01000	1	1	0	Soma	00101	0
01010	11100	1	1	0	Soma	01110	0
11111	00111	1	1	0	Soma	11000	1
10001	01110	1	1	0	Soma	00011	1
10110	00001	1	1	0	Soma	10101	1