Aritmética Digital e Construção da Unidade Lógica Aritmética (ULA)

João Pedro Rey

August 2025

1 Aritmética Binária

A Unidade Lógica Aritmética (ULA) é um dos blocos mais importantes da Unidade Central de Processamento (CPU) em sistemas digitais. É responsável por executar operações aritméticas (como soma e subtração) e operações lógicas (como AND, OR, NOT e XOR) sobre dados binários. Todo processador, desde microcontroladores simples até CPUs avançadas, depende de uma ULA eficiente. Compreender seu funcionamento é fundamental para entender como computadores e sistemas embarcados processam informações.

Este documento visa apresentar os conceitos básicos acerca da aritmética binária, a implementação a nível de portas lógicas de uma ULA, a estrutura interna de um circuito somador completo, a simulação de uma ULA de 4 bits.

2 Adição de Números Binários

A operação em sistemas digitais é baseada no sistema binário, onde cada dígito (bit) só pode assumir dois valores: 0 ou 1. Assim como no sistema decimal, é possível realizar soma, subtração, multiplicação e divisão. Neste estudo, consideraremos inicialmente números binários de 4 bits. É importante notar que, para representar números decimais, podemos converter valores para BCD (Binary Coded Decimal) ou hexadecimal.

Existem apenas cinco possibilidades de resultado para a soma de dados binários, sendo estes resultados apresentados a seguir:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10_2 \ (resultado\ 0\ com\ carry\ de\ 1)$$

$$1 + 1 + 1 = 11_2 \ (resultado\ 1\ com\ carry\ de\ 1)$$

Note que para o caso em particular da soma entre 1+1, o resultado é o binário 10, equivalente ao decimal 2, portanto o valor da soma é igual a 0 e a mesma carrega um carry de 1 para a posição do próximo bit a ser somada, o mesmo ocorre para o caso da soma de 1+1+1. A seguir temos exemplo da soma entre os números 11 e 3 em forma binária:

$$1011_2\ (11_{10}) + 0011_2\ (3_{10})$$

Somando os bits da direita para esquerda temos:

$$1 + 1 = 0 + (carry\ de\ 1)$$

$$1 + 1 + (carry\ de\ 1) = 1 + (carry\ de\ 1)$$

$$(carry\ de\ 1) + 0 + 0 = 1$$

$$1 + 0 = 1$$

Resultado da soma: $1110_2\ (14_{10})$

3 Subtração de Números Binários

A subtração entre dois números binários pode ser realizada através de uma operação de adição. Para que isso seja possível é necessário converter o número binário a ser subtraído em seu complemento de 2. Para converter um valor binário em seu complemento de 2, primeiro converte-se o mesmo para o seu complemento de 1, ou seja, os 0s se tornam 1s e os 1s se tornam 0s e em sequência soma-se 1 ao bit da posição LSB (Less Significant Bit). Como exemplo, vamos converter o número binário 1100 em seu complemento de 2. O procedimento se dá como segue:

O complemento de 1 de 1100 é 0011.Somando 1 ao LSB deste número binário obtemos:

$$0011_2 + 1$$

Somando da direita para a esquerda:

$$1 + 1 = 0 + (carry\ de\ 1)$$

$$1 + 0 + (carry\ de\ 1) = 0 + (carry\ de\ 1)$$

$$0 + 0 + (carry\ de\ 1) = 1$$

$$0 + 0 = 0$$

Resultado da soma: 0010_2

Obtido o complemento de 2 do número binário a ser subtraído, basta realizar a operação de adição apresentada anteriormente. Como exemplo, vamos realizar a operação de subtração decimal $9_{10} - 6_{10}$ de forma binária:

Primeiro, obtém-se o complemento de 2 do equivalente binário de 6_{10}

$$0110_2\ (6_{10}) => Complemento\ de\ 1 = 1001 => Somado\ a\ 1 => 1010 => Complemento\ de\ 2$$

Em sequência, soma-se o complemento de 2 ao equivalente binário de 9_{10}

$$1001_2\ (9_{10}) + 1010_2\ (6_{10})$$

Somando os bits da direita para a esquerda, temos:

$$1 + 0 = 1$$

$$1 + 0 = 1$$

$$0 + 0 = 0$$

$$1 + 1 = 0 + (carry\ de\ 1)$$

Resultado da soma: $0011_2\ (3_{10})$

É importante salientar que sempre que houver um carry na soma do bit mais significativo, o mesmo deve ser descartado. Neste caso, os procedimentos de soma e subtração consideraram apenas números decimais sem sinal.

Para realizar operações aritméticas com números decimais positivos e negativos é necessário acrescentar um bit extra denominado bit de sinal na posição à esquerda do MSB (Most Significant Bit). Caso o bit de sinal tenha valor 0, o número apresenta sinal positivo, caso o bit de sinal apresente o valor 1, o número apresenta sinal negativo. Em certos casos, pode ocorrer o chamado overflow aritmético, situação cujo o resultado de uma operação excede o intervalo representável. Em operações com complemento de 2, ele pode ser detectado comparando o carry de entrada e de saída do MSB através de uma porta XOR. Caso os bits sejam distintos, e a saída da XOR for verdadeira, ocorreu um overflow na operação.

Agora que vimos como realizar a adição ou subtração entre dois números binários, vamos conhecer o circuito lógico que implementa estas operações.

4 Somador Binário Paralelo

A figura a seguir apresenta a configuração básica de um somador binário paralelo de quatro bits. Nesta configuração notamos a presença de dois registradores X e Y cuja função é armazenar os dados binários que serão utilizados para realizar as operações aritméticas.

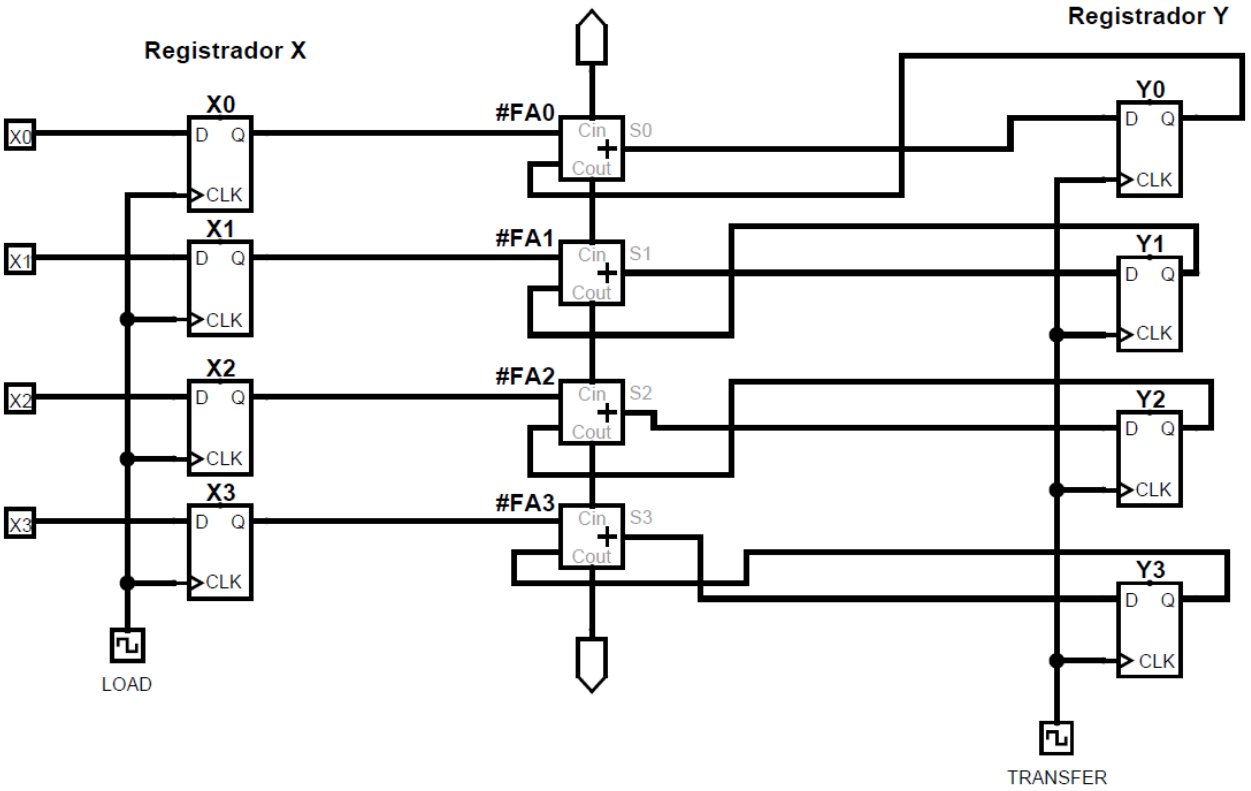

Figure 1: Somador Binário Paralelo com Registradores.

Este circuito somador recebe os dados binários armazenados na unidade de memória através de quatro Flip-Flops D durante a borda de subida do pulso de clock denominado LOAD. O registrador Y localizado na saída do circuito é denominado de acumulador, pois sua função é armazenar o resultado da operação binária, podendo ou não transferir este resultado para a unidade de memória ao fim de cada operação. O bloco intermediário do circuito é composto por quatro circuitos somadores completos (Full-Adder) responsáveis por realizar a operação binária. Note que o somadores possuem três entradas (Bit X, Bit Y e Carry in) e duas saídas (Bit Soma e Carry out). Inicialmente, o registrador Y encontra-se no estado 0000. Após a operação, o dados contidos nas saídas dos somadores completos são transferidos para o registrador Y durante a borda de subida do pulso de clock denominado TRANSFER.

Note que o bit C_{in} do somador FA0 à primeira vista de maneira intuitiva deve ser 0, no entanto, para o caso da subtração de números binários ele deve assumir o valor 1 para seja feito o complemento de 2 do número binário a ser subtraído. O bit C_{out} do somador FA3 é útil para indicar caso haja um overflow aritmético durante a operação com os dados binários.

5 Circuito Somador Completo

Para compreender o principal circuito digital responsável pela realização das operações aritméticas entre números binários, podemos construir a tabela verdade para o somador completo como segue:

X	Y	Cin	S	Cout
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

Figure 2: Tabela-Verdade para o circuito Somador Completo.

Através da tabela-verdade extraída do circuito somador completo, podemos avaliar individualmente cada saída a fim de estruturar um circuito lógico combinacional que realize a operação que desejamos. Portanto, começando com o bit de saída S que representa o resultado da soma entre dois bits, podemos expressá-lo por:

$$S = \overline{X} \cdot \overline{Y} \cdot C_{in} + \overline{X} \cdot Y \cdot \overline{C_{in}} + X \cdot \overline{Y} \cdot \overline{C_{in}} + X \cdot Y \cdot C_{in}$$

Analisando a expressão, podemos fatorar o termo C_{in} e $\overline{C_{in}}$ como segue:

$$S = C_{in} \cdot (\overline{X} \cdot \overline{Y} + X \cdot Y) + \overline{C_{in}} \cdot (\overline{X} \cdot Y + X \cdot \overline{Y})$$

Os termos em evidência representam as expressões das portas XNOR e XOR, respectivamente. Portanto a expressão para a saída S é dada por:

$$S = C_{in} \cdot \overline{(X \oplus Y)} + \overline{C_{in}} \cdot (X \oplus Y)$$

$$S = C_{in} \oplus X \oplus Y$$

Podemos realizar a mesma análise para o bit de saída C_{out} como segue:

$$C_{out} = \overline{X} \cdot Y \cdot C_{in} + X \cdot \overline{Y} \cdot C_{in} + X \cdot Y \cdot \overline{C_{in}} + X \cdot Y \cdot C_{in}$$

Utilizando o mapa de Karnaugh para simplificar a expressão obtemos:

	C_{in}	$\overline{C_{in}}$
$\overline{X}\overline{Y}$	0	0
$\overline{X}Y$	1	0
XY	1	1
$X\overline{Y}$	1	0

Figure 3: Mapa de Karnaugh para simplificação da expressão de C_{out}.

Portanto, a expressão para o bit de saída C_{out} será:

$$C_{out} = X \cdot Y + Y \cdot C_{in} + X \cdot C_{in}$$

Portanto, com as expressões dos bits de saída S e C_{out} podemos construir um circuito lógico combinacional que realiza a operação de soma entre dois números binários como segue:

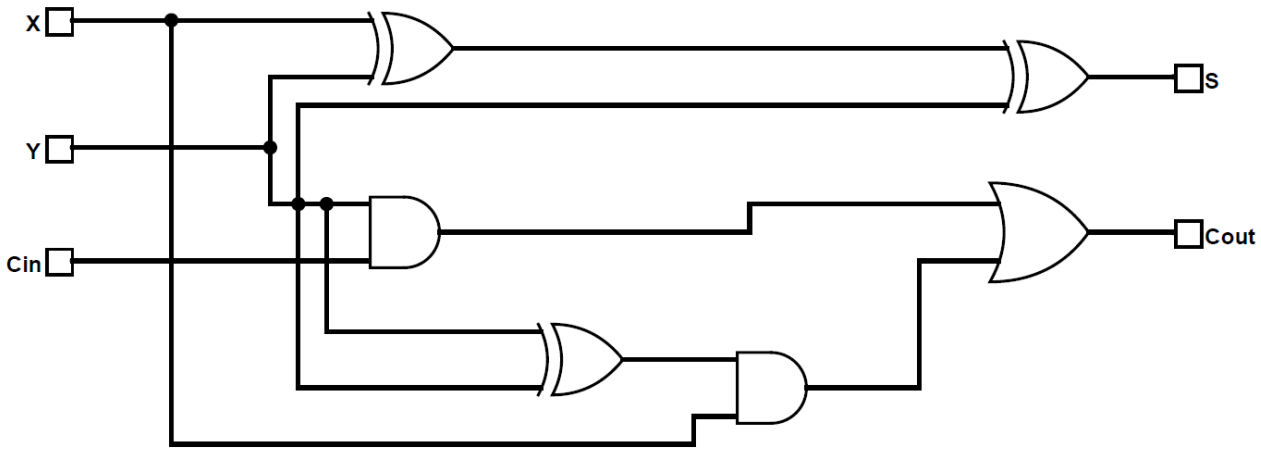

Figure 4: Circuito somador completo construído a partir de portas lógicas.

6 Unidade Lógica Aritmética

A Unidade Lógica Aritmética (ULA) de um sistema digital é responsável por realizar todas as operações necessárias entre dois dados binários de forma a retornar o valor da operação em sua saída. A figura 5 apresentada na sequência representa uma ULA completa, capaz de realizar operações lógicas e aritméticas entre dois números binários de quatro bits de entrada.

A estrutura da ULA pode ser analisada em blocos separados. O primeiro bloco é composto por quatro somadores completos responsáveis por realizar a soma entre os números binários A e B. A saída de cada somador é conectada a um bit de seleção dos multiplexadores situados na saída da ULA responsáveis por selecionar a operação desejada. Note que os bits que compõem o número binário A são ligados diretamente às entradas dos somadores completos, enquanto os bits que compõem o número binário B são ligados a uma das entradas de quatro portas XOR. O bit ADD/SUB está conectado a todas as entradas restantes das portas XOR. Desta forma, quando o bit ADD/SUB assumir o valor de 1 é possível realizar, através da lógica das portas XOR, o complemento de 1 do número binário B, cujo valor é somado a 1 através do bit de carry in do somador 0. Desta forma é possível gerar o complemento de 2 do número binário B e portanto realizar a operação de subtração.

Os blocos subsequentes são responsáveis por realizar as operações lógicas AND, OR e XOR entre os números binários de entrada. É interessante notar que através do uso de portas OR ligadas ao bit XOR/NOT é possível realizar a operação NOT do número A. A lógica aplicada consiste em manter o bit XOR/NOT em nível lógico 1, desta forma todas as portas XOR recebem em uma de suas entradas um bit de valor 1. Portanto, mesmo que o valor do bit do número binário B seja 0, a porta XOR recebe o valor 1 em sua entrada. Dessa forma, caso o valor do bit do número A seja igual a 1, a saída da porta XOR é 0. Portanto, é realizada a negação bit a bit do número A e este dado é retornado na saída.

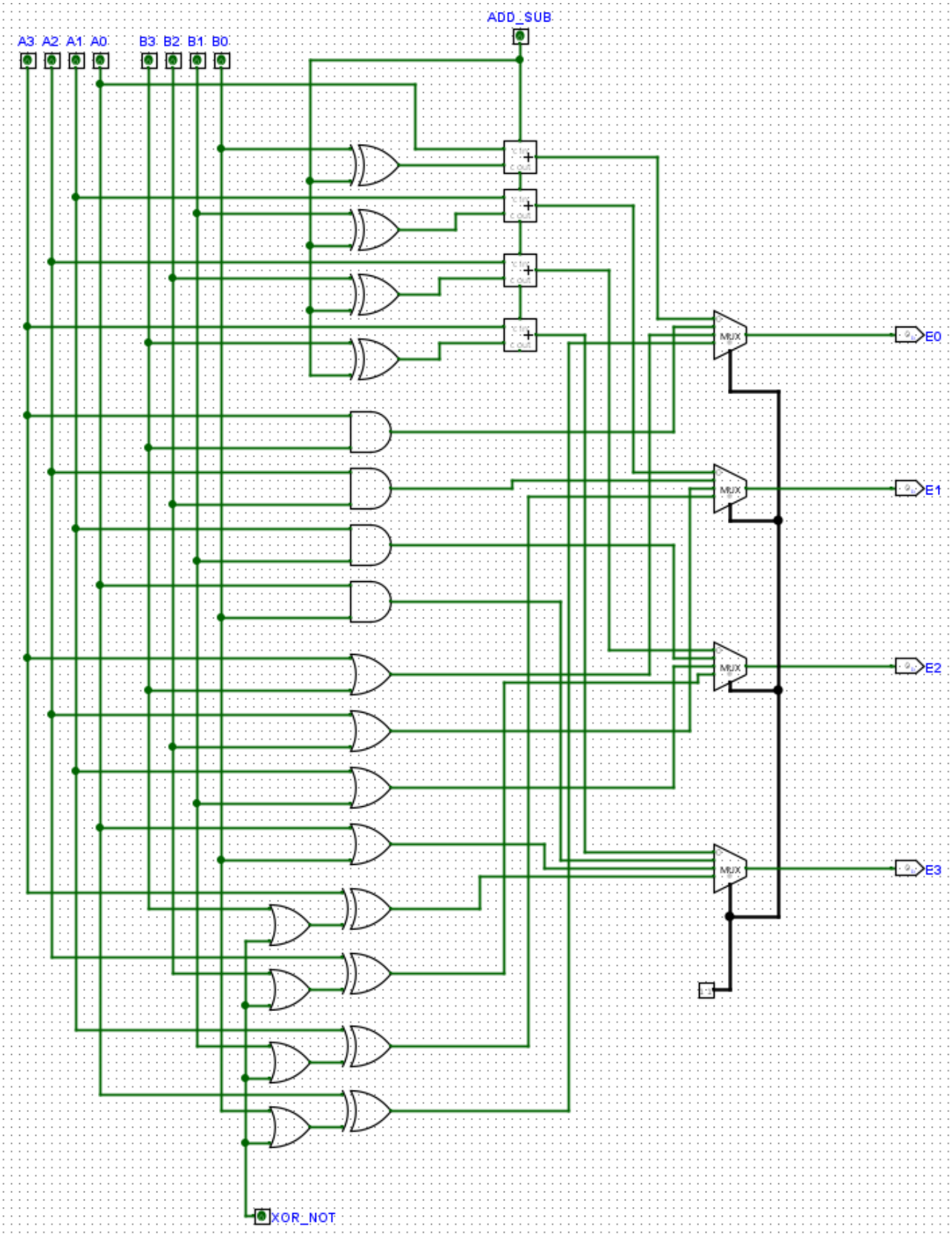

Figure 5: Arquitetura de uma Unidade Lógica Aritmética (ULA) de 4 bits.

Na sequência está representado através da simulação do circuito utilizando o software Logisim, a operação binária de subtração entre os números 9 e 6. Perceba que na entrada do número binário A temos o conteúdo 1001, enquanto na entrada do número Binário B temos o conteúdo 0110. O bit ADD/SUB deve estar em 1 para realizar o complemento de 2 do número 0110. Logo, temos o conteúdo 0011 na saída dos multiplexadores, representando o número decimal 3.

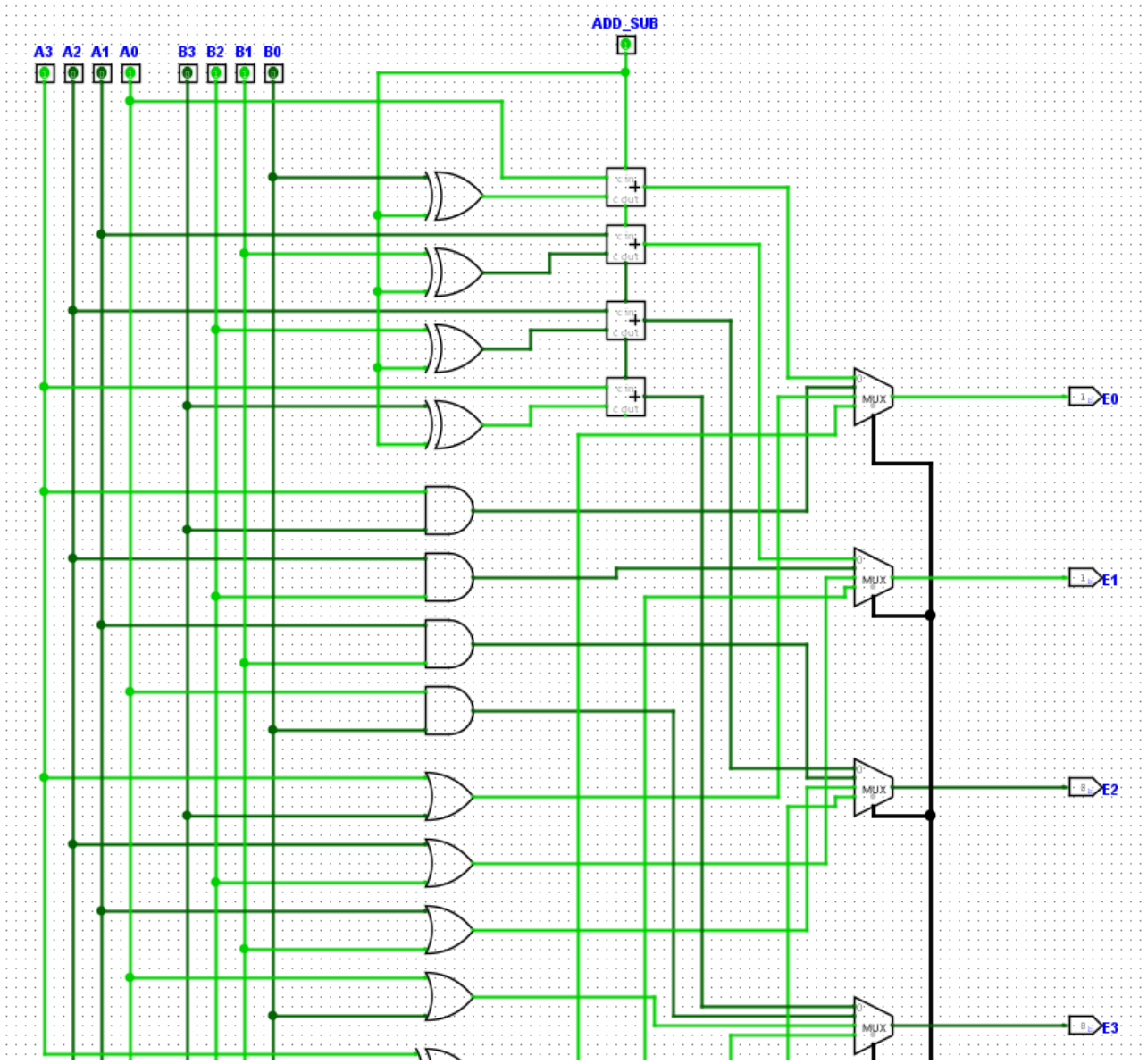

Figure 6: Operação de subtração entre os números binários 1001 e 0110.

Para somar dois números decimais como por exemplo 8 e 4, devemos manter o bit ADD/SUB em 0. Desta forma, o número binário A recebe o conteúdo 1000 e o número binário B recebe o conteúdo 0100. Observe que na saída temos o conteúdo binário 1100, cuja valor decimal equivalente é 12 como está apresentado pela figura 7.

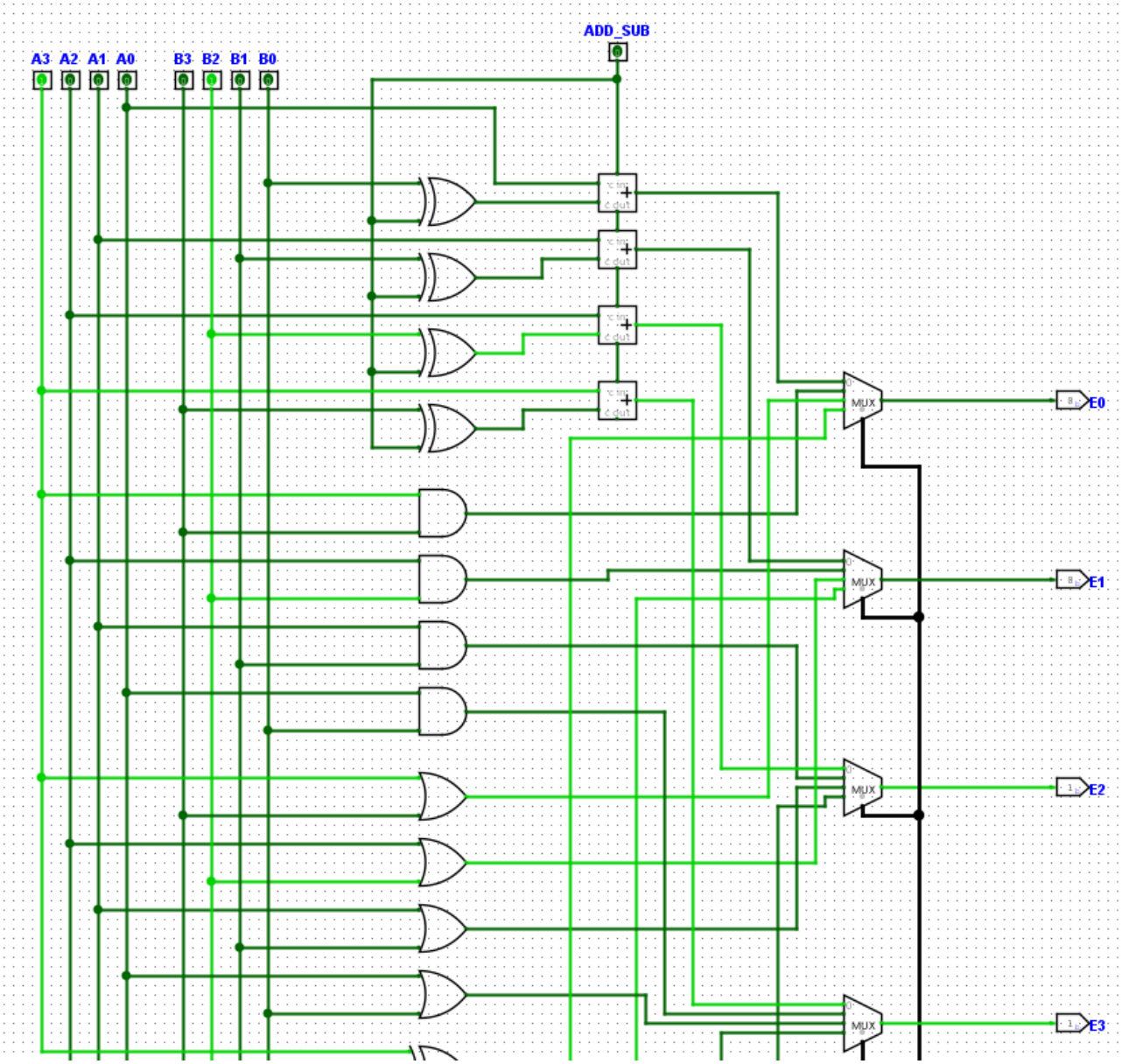

Figure 7: Operação de adição entre os números binários 1000 e 0100.

Os bits de comando dos multiplexadores alteram a operação lógica ou aritmética a ser realizada a partir dos números binários de entrada. Por exemplo, caso os bits de comando sejam 00 a operação realizada é a adição. Caso os bits de comando sejam 01, a operação realizada é a AND. A operação AND entre dois números binários está apresentada a seguir:

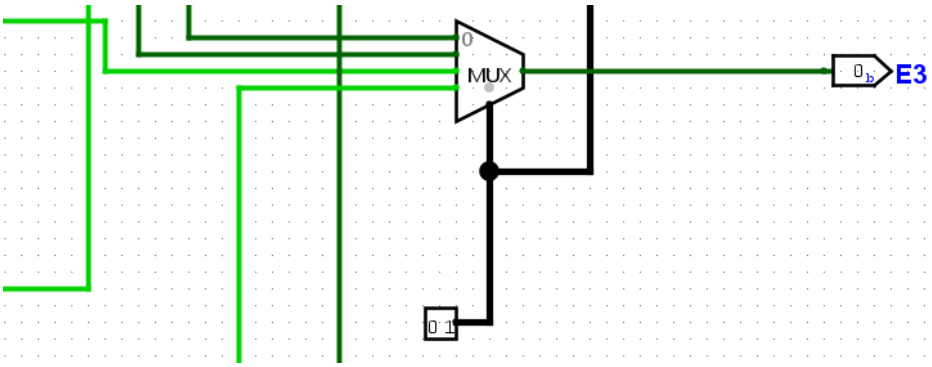

Figure 8: Bits de comando dos multiplexadores.

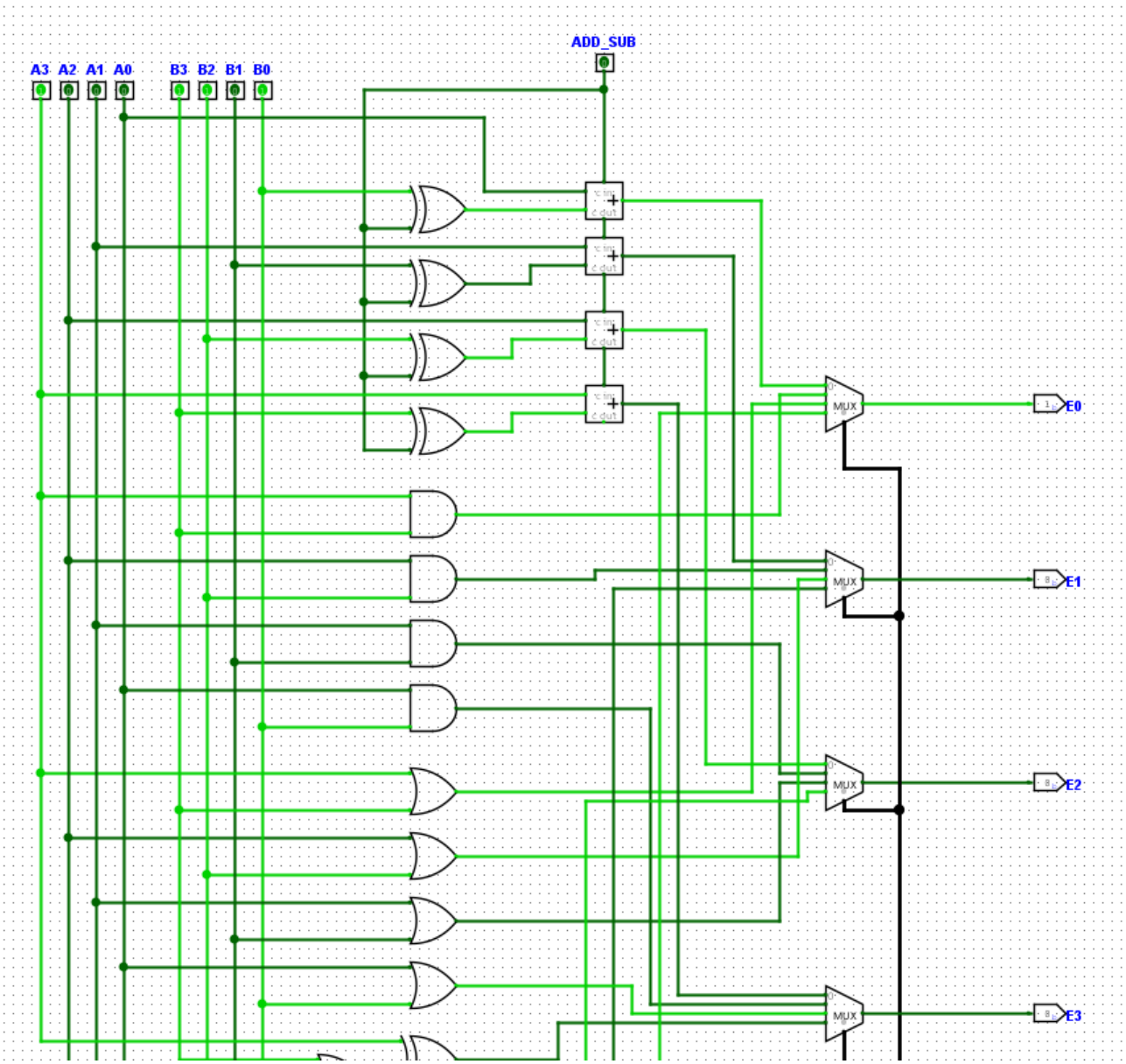

Figure 9: Operação lógica AND entre os números binários 1000 e 1101.

Selecionando os bits de comando como 10 temos a operação lógica OR. Caso sejam selecionados os bits de comando para 11 teremos a operação lógica NOT do número binário A, portanto, a saída será o complemento de 1 de A. Em sequência está representada a operação NOT para o número binário de entrada 1010 como ilustra a figura 10. Note que independentemente dos valores contidos no número binário B, o bit XOR/NOT mantém uma das entradas das respectivas portas XOR em nível lógico 1, possibilitando que o complemento de 1 do número binário A seja retornado na saída da ULA.

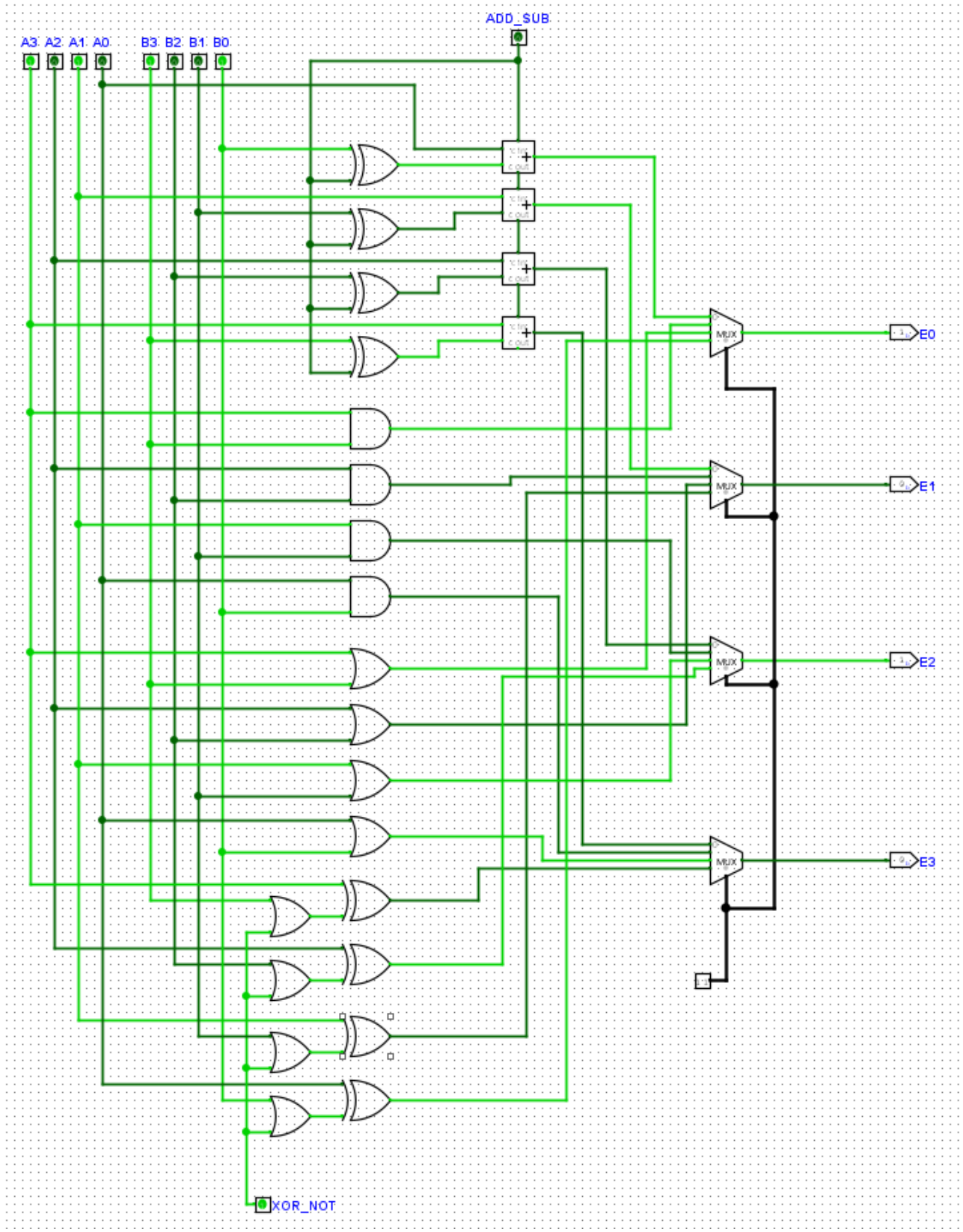

Figure 10: Operação NOT do número binário 1010.

Portanto, através da lógica combinacional utilizando portas lógicas é possível estruturar uma Unidade Lógica Aritmética capaz de realizar operações complexas em alguns poucos nanosegundos. A ULA é parte fundamental da arquitetura de um microprocessador e integra a Unidade Central de Processamento (CPU) do sistema digital. Compreender como são realizadas estas operações entre números binários é a base para entender como os sistemas digitais operam e qual a lógica envolvida em sua estrutura.