**Circuitos Lógicos**

**Aula nº 06 - Somador de 4 bits**

**Nome(s):** Afonso Salvador de Magalhães e Thamya Vieira Hashimoto Donadia

**Data:** 09/06/2022

1. **Objetivos da aula**

A proposta da aula consiste na construção de um somador/ subtrator de 4 bits, por meio do uso de 4 somadores completos, de um multiplexador e da teoria do complemento de 2. Ademais, o projeto permite a indicação do estouro (overflow).

1. **Metodologia**
   1. **Abordagem do problema**

A priori, foi projetada a arquitetura do programa, o qual foi modularizado em 4 arquivos distintos: um somador completo de 1 bit (somador.vhd); um somador completo de 4 bits (soma\_4bits.vhd), construído a partir de 4 somadores de 1 bit; um multiplexador 2x1 de 4 bits (mux2\_1\_4bit); e, por fim, o top level (top.vhd) que utiliza as demais entidades criadas.

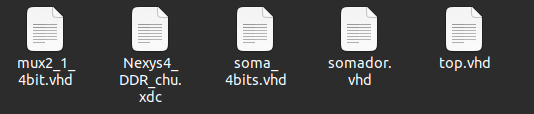
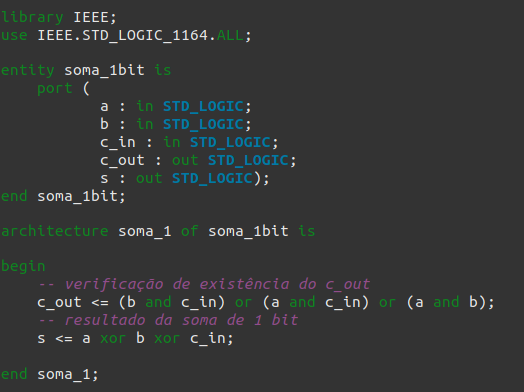


Figura 1- Arquivos utilizados para montagem do programa.

* 1. **Montagem da base do programa**

O projeto foi, então, iniciado pela modificação do arquivo de constraints, em que foram utilizados apenas o switches, para o input de dados, e os LEDs discretos, a fim de indicar o valor final. Em seguida, a dupla construiu o somador completo de 1 bit, de modo que foi calculado o “vai 1” (c\_out) e o resultado da soma (s), como apresentado na figura 2.

Figura 2 - Código do somador completo de 1 bit (somador.vhd)

Ademais, foi utilizada a mesma função, só que desta vez em série, em outro arquivo (soma\_4bits.vhd), para confeccionar um somador completo de 4 bits. Isso foi feito por meio da conexão de cada bit dos valores somados/subtraídos em um somador de 1 bit, em seguida, da ligação do c\_out de um somador com a entrada de c\_in do somado acima, como apresentado na figura 3.

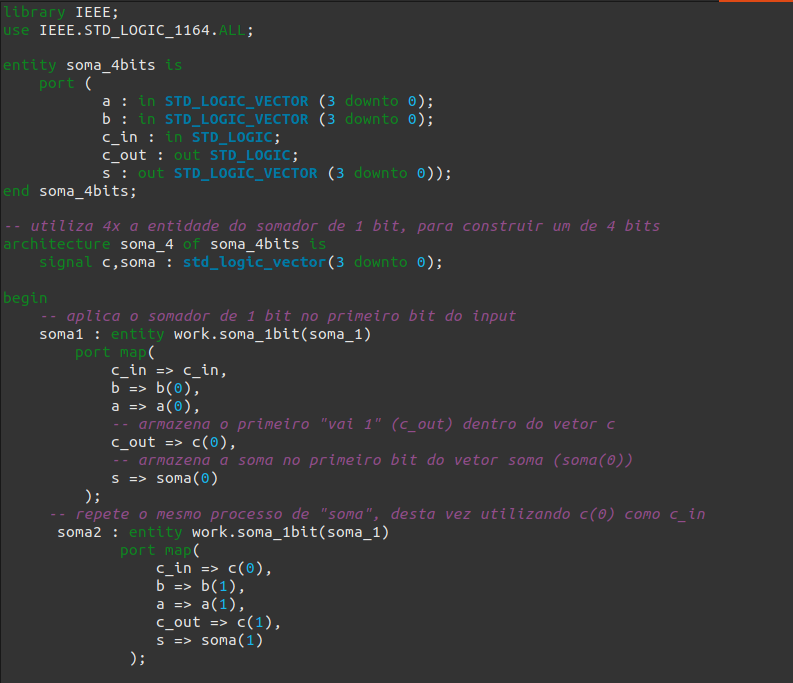


Figura 3 - Código do somador completo de 4 bits (soma\_4bits.vhd).

Além disso, no final das somas de cada bit, o último bit de cada número (soma, valor a, valor b) foi utilizado para a detecção de overflow, visto que uma soma de valores negativos não pode resultar em positivo e uma soma de positivos não pode resultar em negativos. Desse modo, o circuito lógico da figura 4 foi implementado em VHDL como apresentado na figura 5.

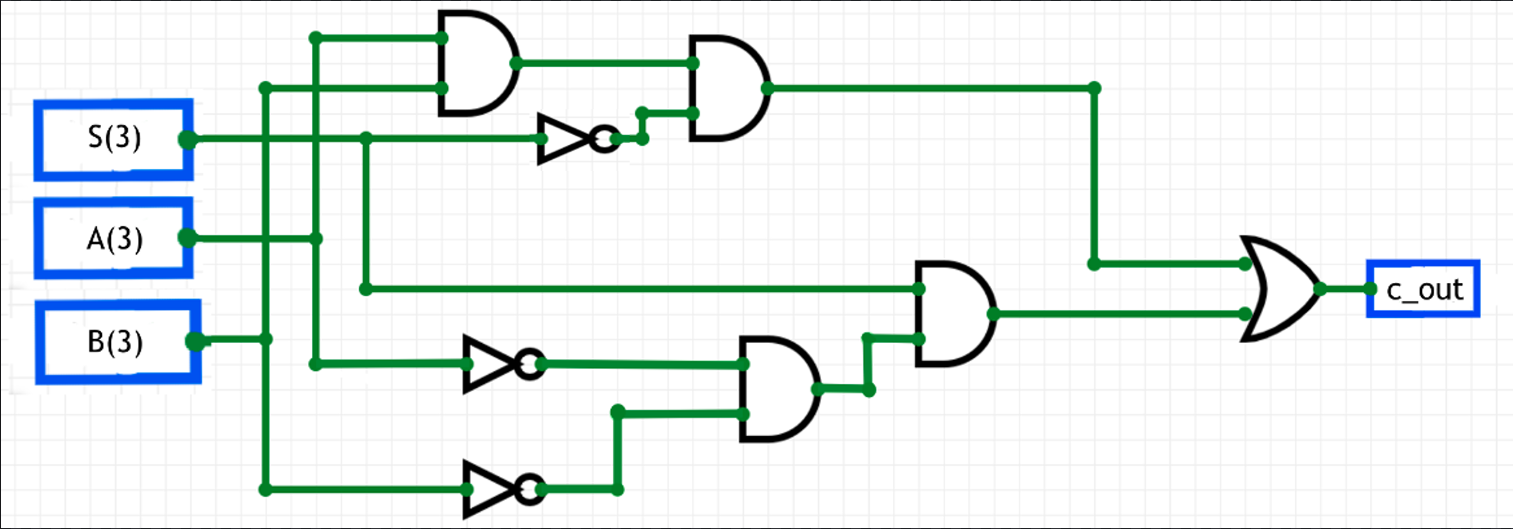


Figura 4 - Circuito Lógico de detecção de overflow.

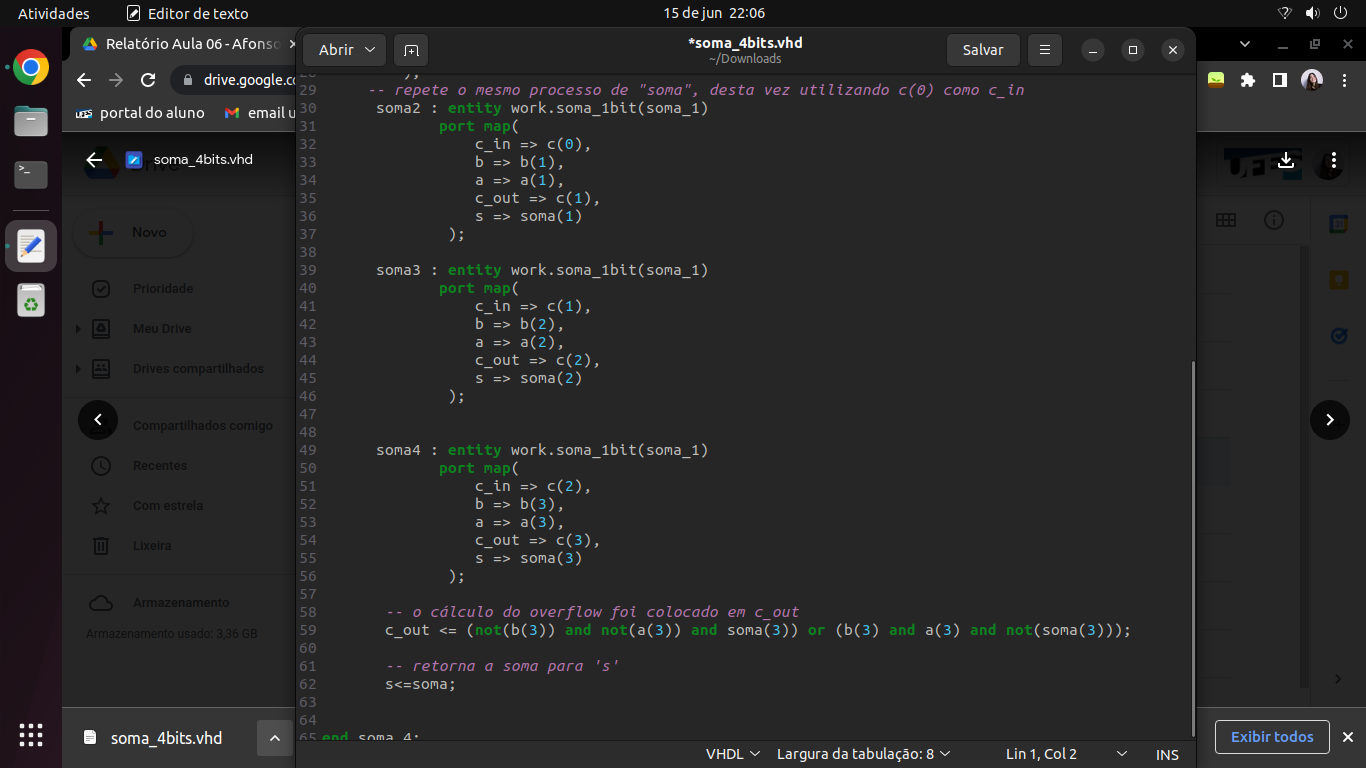


Figura 5 - Cálculo do overflow

* 1. **Programação do top**

Após a projeção de todas as funções a serem utilizadas no exercício, foi iniciada a montagem do arquivo top, no qual todas as funções foram diretamente ou indiretamente utilizadas. Antes de iniciar a montagem, foi planejada a disposição de informações necessárias na placa, com todas os switches de variáveis também possuindo a função de acender seu respectivo led, o que facilitou a visualização dos resultados dos testes.

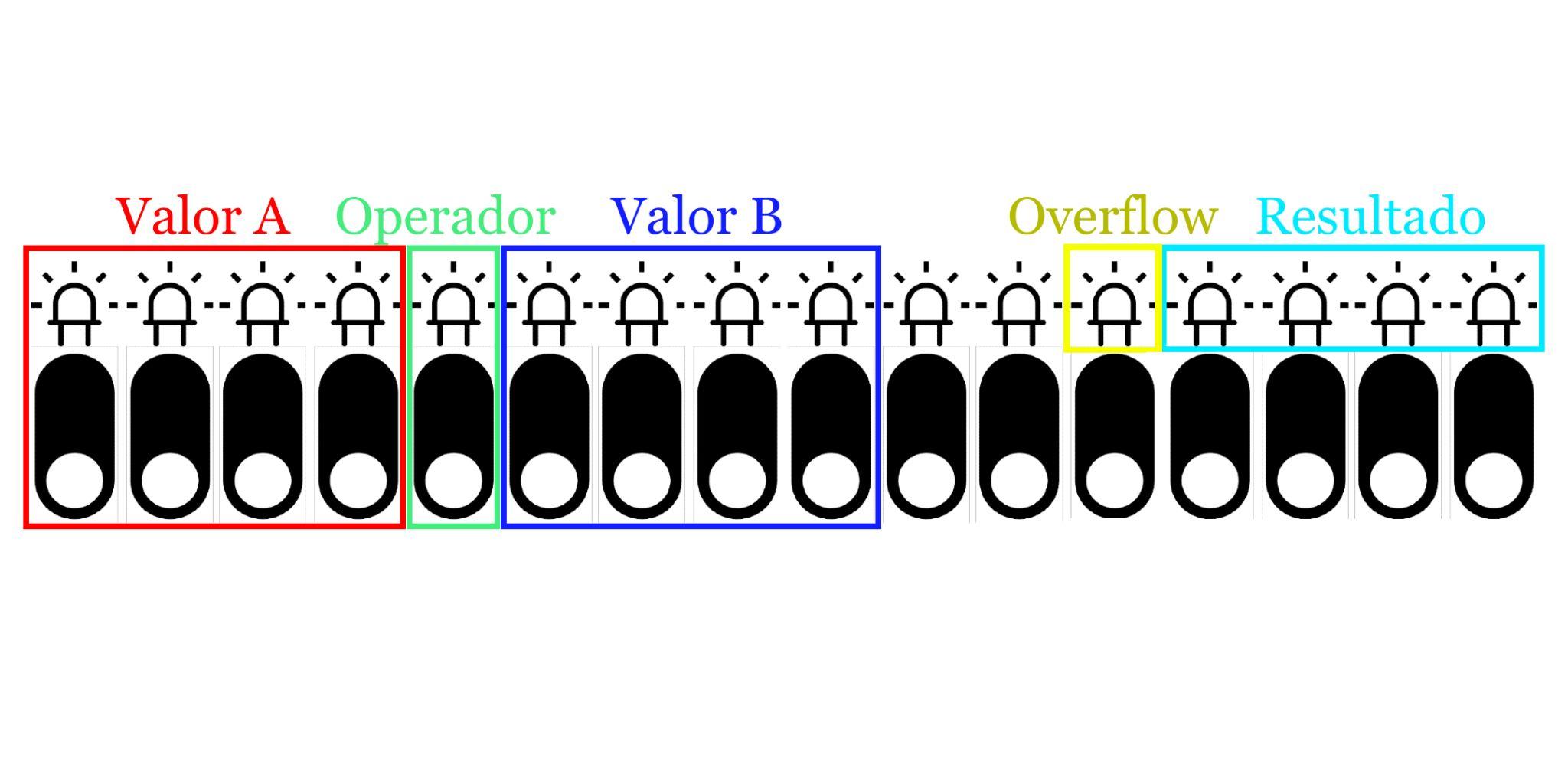


Figura 6 - Disposição de informação nos elementos da placa

Ademais, ainda com o intuito de facilitar a visualização dentro do código, foram criados diversos sinais (únicos e vetores) para ligar à valores de input, de modo a obter um sinal com nomes intuitivos.

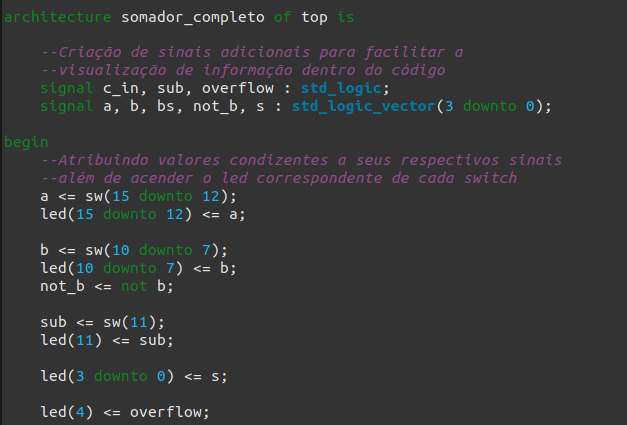


Figura 7 - Alocação e atribuição de sinais lógicos, além de ligação com LEDs.

Assim, foi iniciada a montagem da parte lógica do código. A primeira sessão consiste no MUX de seleção entre a operação de soma e a operação de subtração. Caso a subtração seja selecionada, o sinal aciona o c\_in no primeiro bit do somador completo, além de inverter o sinal do valor B, negativando-o.

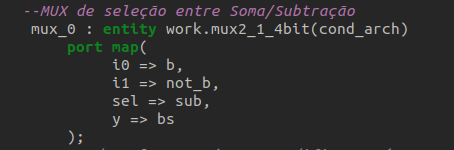


Figura 8 - MUX de seleção soma/subtração

Após a seleção da operação e ser feita, a entidade de soma é chamada, que recebe como input os sinais a, b (após a sua possível modificação) e o c\_in. Como output tem-se a soma completa e o indicador de overflow que, caso acionado, indica que o resultado da soma não está correto.

1. **Resultados**

Para exemplificação do código, apresenta-se abaixo um exemplo de algumas das operações feitas na placa com o código já finalizado.

* 1. **Teste 1 (soma com overflow de valor positivo)**

Neste primeiro teste, há uma soma simples que possui overflow. A soma é feita entre os números 7 e 1, que deveria resultar em 8. No entanto, não há espaço em 4 bits para representar o número 8, já que a representação se limita a valores de 7 a -8. Desse modo, há overflow.

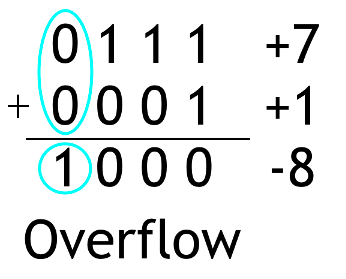


Figura 9 - Resultado esperado do teste 1



Figura 10 - Resultado prático do teste 1

* 1. **Teste 2 (soma sem overflow)**

Neste teste, há uma soma entre os valores entre os valores de -8 e 7, a qual não resulta em overflow, uma vez que a soma entre valores positivos e negativos nunca sai do escopo de representação. Assim, a operação resulta no valor de -1.

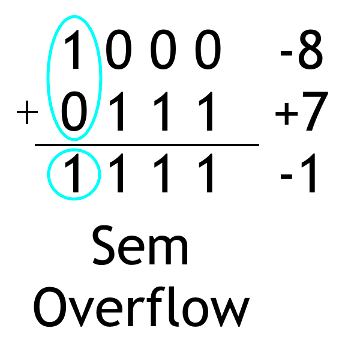


Figura 11 - Resultado esperado do teste 2

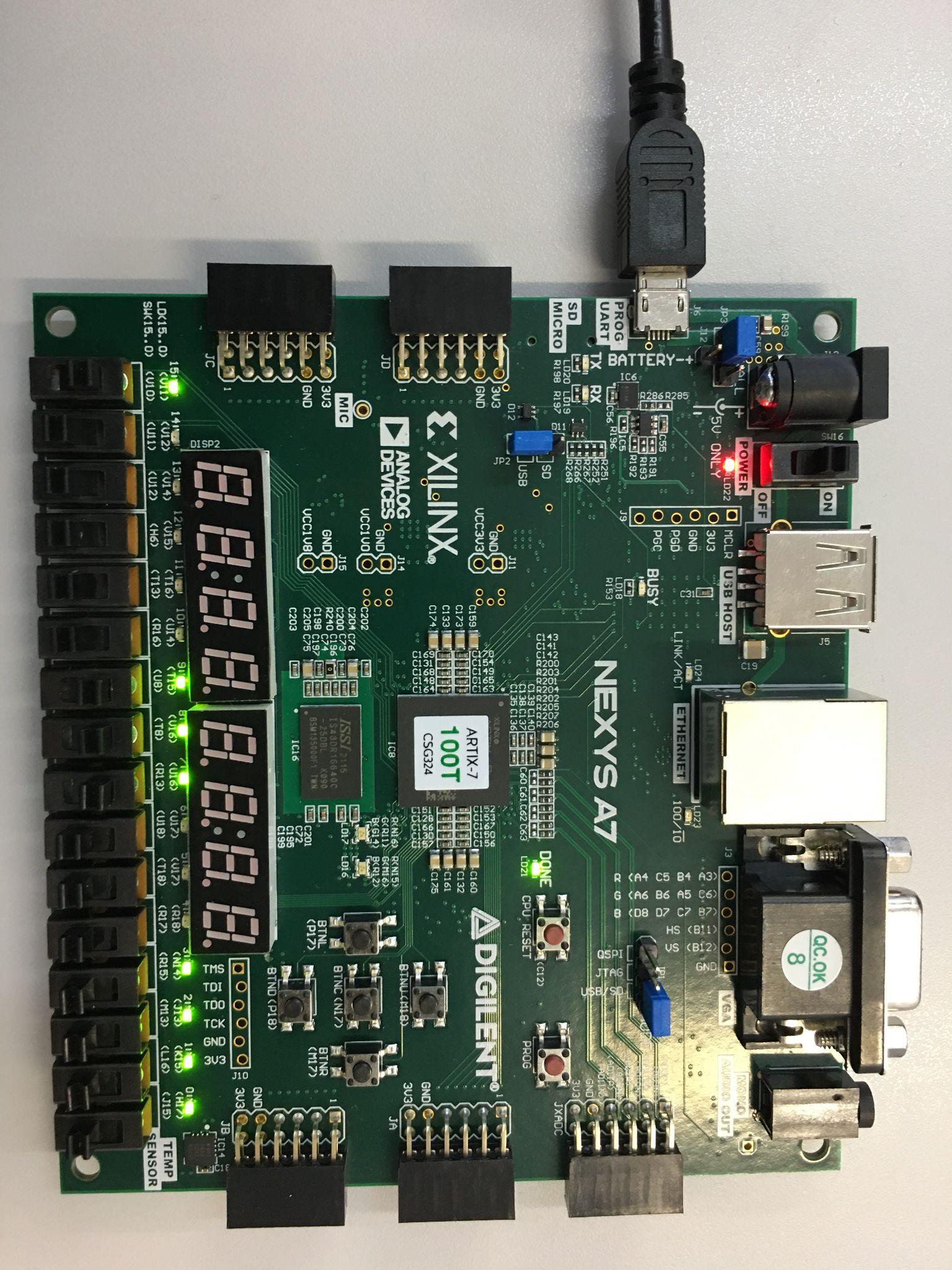


Figura 12 - Resultado prático do teste 2

* 1. **Teste 3 (soma com overflow de valor negativo)**

Por último, foi realizada uma soma com dois valores negativos, -1 e -8, em que, visto que não há representação de valores menores que -8, resulta em um overflow. Desse modo, a operação apresenta um resultado diferente do correto (-9).

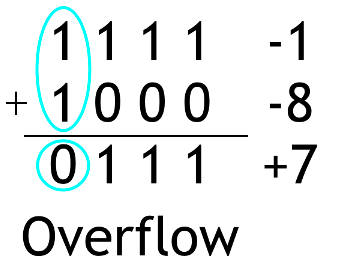


Figura 13 - Resultado esperado do teste 3

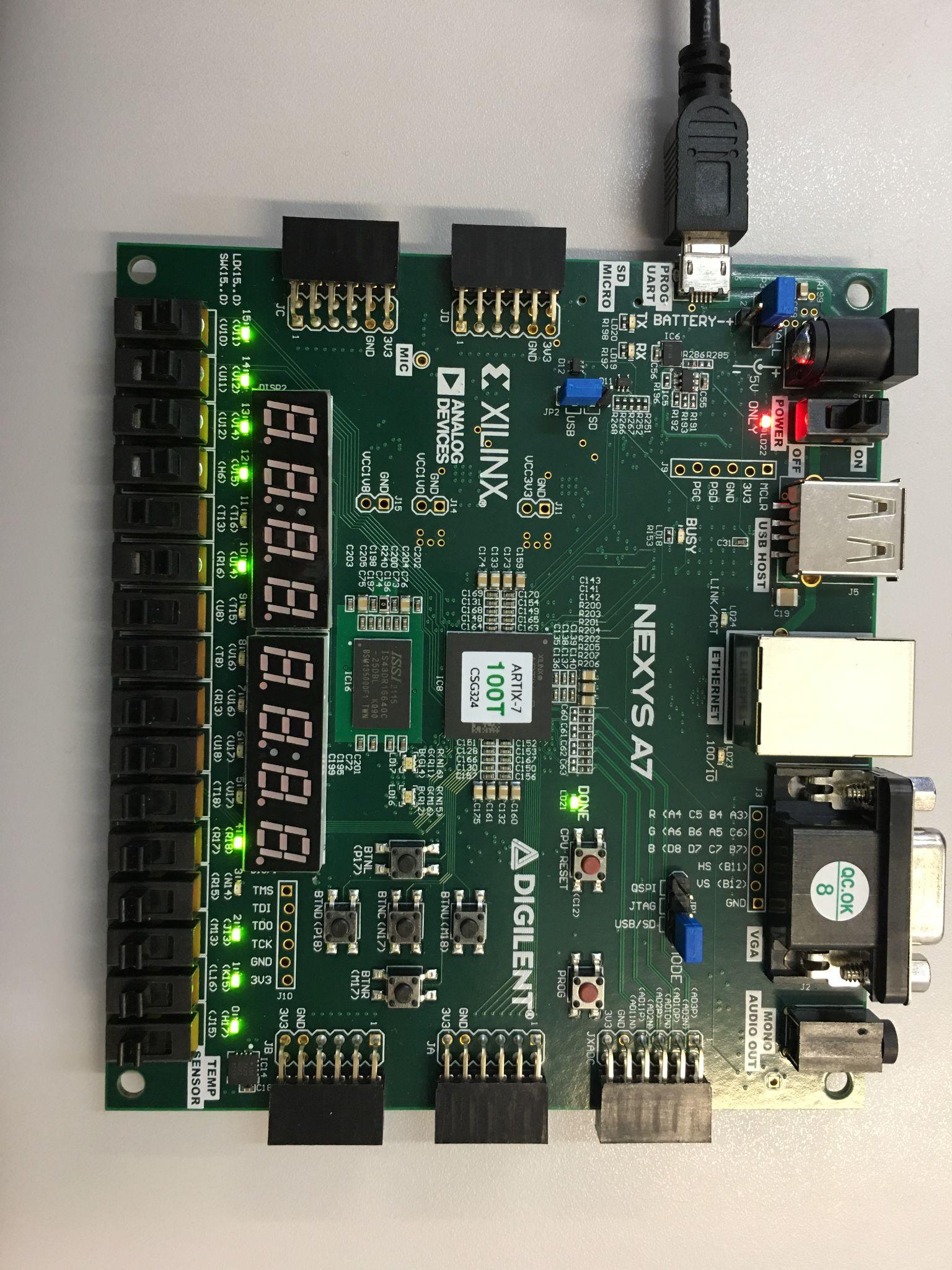


Figura 14 - Resultado prático do teste 3

1. **Conclusões**

Ao finalizar o projeto foi possível aplicar na prática o somador/ subtrator completo de 4 bits, bem como a análise e a detecção do overflow em determinados casos. Ademais, verificou-se a representação de um número negativo em binário, a qual ainda não havia sido apresentada durante o curso.