Departamento de Eletrónica Telecomunicações e Informática

8240 - Mestrado Integrado em Engenharia de Computadores e Telemática

Computação Reconfigurável

2021/2022

Projeto

Acelerar a verificação e contagem de palíndromos com um Coprocessador de Hardware

Autores:
Lúcia Sousa 93086
Raquel Pinto 92948

21 de Junho de 2022

Contents

1 Descrição do projeto

Este projeto é um sistema com um acelerador hardware para verificar se cada vetor de 32 bits, lido da memória (no total são 16 vetores de 32 bits cada um), é um palíndromo e contar quantos palíndromos existem na memória. Estas entradas não serão geradas aleatoriamente. São entradas em que umas são palíndromos e outras não.

Este sistema vai ter um módulo hardware (CheckCounterPalindrome, Figura 1) que tem uma interface AXI-Stream Slave para receber dados do MicroBlaze que envia dados para o novo módulo através da interface AXI-Stream Master. Tem também uma interface AXI-Stream Master para enviar dados ao Microblaze que os recebe através da sua interface AXI-Stream Slave. Por último, tem também uma interface AXI-Lite Slave que comunica com AXI Interconnect para realizar um contador acumulativo feito por software. Como pode ser visto na Figura 2.

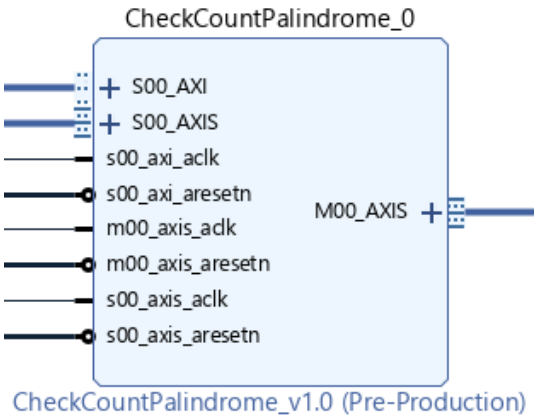

Figure 1: Modulo CheckCounterPalindrome.

A interface AXI-Stream Slave do nosso modulo após receber um vetor de 32 bits vai verificar se este vetor é um palíndromo ou não, devolvendo um vetor de 32 bits a 0000 0000 0000 0000 0000 0000 0000 0000 se não for palíndromo ou devolvendo um vetor de 32 bits a 0000 0000 0000 0000 0000 0000 0000 0001 caso o vetor de entrada seja um palíndromo.

Este resultado é enviado ao Microblaze através da saida AXI-Stream Master e é escrito no AXI Interconnect no endereço base. Assim consegue-se ter acesso ao resultado podendo-se iniciar o contador acumulativo. Se o resultado indicar que é um palíndromo, é lido o valor do contador guardado no AXI Interconnect no endereço base + 4, é incrementado uma unidade e é escrito o novo valor no AXI Interconnect no endereço base + 4.

No final são verificados os resultados obtidos e é comparado o desempenho do software e hardware, onde é de esperar que os resultados do software e do hardware sejam iguais e que o hardware seja mais rápido que o software.

2 Algoritmo de verificação de palíndromos em VHDL

O módulo PalindromoCheck foi implementado para verificar se um vetor de 32 bits é um palíndromo. Este contém uma entrada de 32 bits e uma saída de 32 bits.

O vetor que recebe como entrada é invertido, e se a palavra que entrou é igual ao seu inverso, é um palíndromo e é concatenado a 31 zeros o bit a 1 do resultado, caso não seja igual, são enviados 32 bits a zero.

3 Block Design

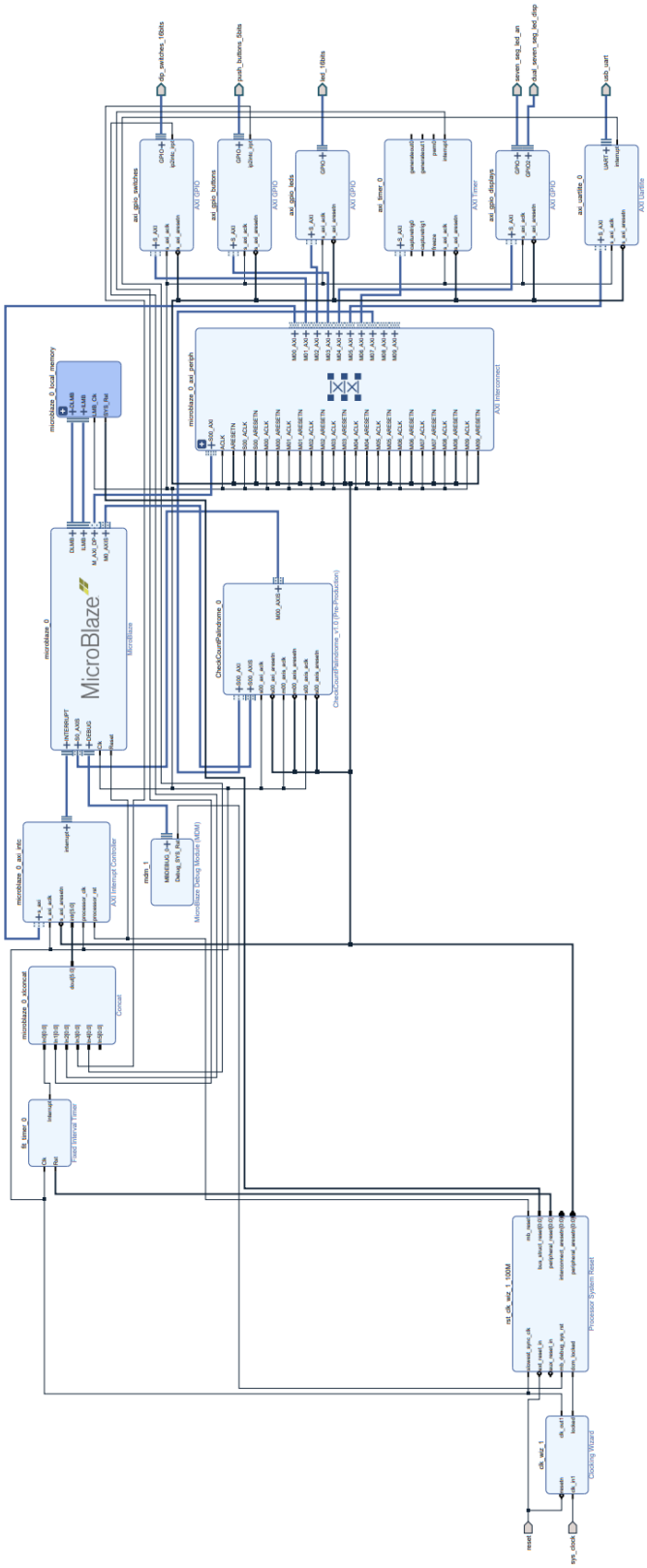

Figure 2: Block design.

4 Resultados

Neste projeto são enviados 16 vetores de 32 bits, estes vetores são guardadas no srcData. Os que são palíndromos estão armazenados nos índices 0, 3, 4, 6, 8, 11, 13 e 15, como indicado na Figura 3. Estes vetores são então testadas em software e hardware.

```
//palindromos = 9 -> indices 0, 3, 4, 6, 8, 9, 11, 13, 15
srcData[0] = 1704565158; //0110 0101 1001 1001 1001 1001 1010 0110
srcData[1] = 35648752;   //0000 0010 0001 1111 1111 0100 1111 0000
srcData[2] = 249224586;  //0000 1110 1101 1010 1101 1101 1000 1010
srcData[3] = 4294967295; //1111 1111 1111 1111 1111 1111 1111 1111
srcData[4] = 0;          //0000 0000 0000 0000 0000 0000 0000 0000
srcData[5] = 423077452;  //0001 1001 0011 0111 1010 0110 0100 1100
srcData[6] = 2702176389; //1010 0001 0000 1111 1111 0000 1000 0101
srcData[7] = 767213316;  //0010 1101 1011 1010 1011 1111 0000 0100
srcData[8] = 3536044875; //1101 0010 1100 0011 1100 0011 0100 1011
srcData[9] = 190593744;  //0000 1011 0101 1100 0011 1010 1101 0000
srcData[10] = 463115274; //0001 1011 1001 1010 1001 0100 0000 1010
srcData[11] = 3567625515;//1101 0100 1010 0101 1010 0101 0010 1011
srcData[12] = 520016216; //0001 1110 1111 1110 1101 0001 0101 1000
srcData[13] = 2483703849;//1001 0100 0000 1010 0101 0000 0010 1001
srcData[14] = 356733678; //0001 0101 0100 0011 0101 0010 1110 1110
srcData[15] = 463100376; //0001 1011 1001 1010 0101 1001 1101 1000
```

Figure 3: Palindromos enviados.

Em software, é enviado um inteiro de cada vez, armazenado no srcData, convertido em binário e verificado se é um palíndromo ou não. Para esta verificação é comparado cada bit numa posição com o seu bit na posição oposta. O resultado é armazenado num array, dstDataSW, que mais tarde é utilizado para calcular quantos palíndromos existem. O resultado obtido, como pode ser visto na Figura 4, foi de nove palíndromos.

Em hardware, um valor é enviado e no bloco criado, CheckCountPalindrome, é verificado se é palíndromo e se for é contado (como explicado anteriormente). Após verificar e contar os restantes inteiros, o resultado da contagem é devolvido e é possível ver que, como em software o resultado foi nove.

São comparados ambos os resultados, do hardware e do software, para verificar se deram resultados iguais.

```
Filling memory with pseudo-random data...

Memory initialization time: 4 microseconds

Resultado software counter -> 9

Software only time: 33328 microseconds
Resultado counter -> 9

Hardware assisted reverse endianness time: 23954 microseconds

Checking result: OK
```

Figure 4: Resultado do terminal.

Os tempos que estas verificações e contagens demoraram foram retirados e, como previsto, estas operações em software demoraram mais tempo, cerca de 1,39 vezes mais em comparação ao hardware.

5 Contribuição dos autores

Lúcia Sousa - 50%

Raquel Pinto - 50%

Este projeto está organizado em:

- Vivado - diretório **Palindrome**
- Vitis - diretório **CheckCounterPalindrome**