Desenvolvimento de um circuito em *Handel-C* para representação de grafos

Gustavo Corrente

Resumo **– Este artigo visa a descrição do circuito criado para a representação de grafos com 25 vértices, baseado numa *FPGA Spartan-II XC2S200, que faz parte da placa* de desenvolvimento *RC100*, produzida pela *Celoxica*. Para a interacção com o circuito foram usados o rato e o monitor. A especificação do circuito foi na linguagem *Handel-C* num ambiente *DK2*, ambos desenvolvidos pela *Celoxica* [1].**

Abstract **– This paper describes a circuit to be designed for processing graphs with up to 25 vertices. The circuit has been implemented in a *FPGA Spartan-II XC2S200* (the development board *RC100* of *Celoxica*). A mouse and a monitor have been used for interactions with the circuit. The design has been performed in Celoxica [1] *DK2* environment from a specification in *Handel-C*.**

I. INTRODUÇÃO

O projecto aqui descrito, foi desenvolvido no âmbito da disciplina Computação Reconfigurável [2], do 4º ano do Curso de Engenheria de Computadores e Telemática, leccionada pelo professor Valery Sklyarov.

A. Objectivos

O presente projecto visa, fundamentalmente, os seguintes objectivos:

- Aprender a linguagem *Handel-C* [3]
- Aprender a criar um modelo adequado à solução pretendida e ao *hardware*;
- Responder com sucesso à especificidade dos dispositivos periféricos tais como: rato e monitor VGA;
- Interiorizar técnicas de optimização e de reutilização dos recursos da *FPGA*;

B. Descrição global do projecto

O circuito desenvolvido permite a representação de grafos com 25 vértices, de uma forma muito particular. Cada vértice tem uma cor, mas vértices ligados entre si, não podem ter a mesma cor. A atribuição das cores é de uma forma optimizada, sendo o objectivo de colorir os vértices com o menor número de cores possível de modo a satisfizer a regra atrás descrita. Também foi desenvolvido um programa para o *PC*, em C++/QT [4], para facilitar a geração de ficheiros para teste.

C. Características principais da placa RC100

A placa *RC100* [5] tem como componente reconfigurável principal a *FPGA* da família *Spartan-II* da *Xilinx*, com 200.000 portas de sistema.
Esta *FPGA* tem ligação directa com:

- Oscilador de cristal de 80MHz;
- Dois blocos de memória *RAM* síncrona estática (*SSRAM*) com 256K palavras de 36bits cada;
- *Flash RAM* de tamanho 64Mbits;
- Vídeo *DAC* (VGA 24Bits);
- Descodificador da entrada de vídeo;
- Porta *PS/2;*
- *LED*s;
- 2 *Displays* de 8 segmentos;
- Um barramento de expansão.

II. ORGANIZAÇÃO BÁSICA DO PROJECTO

Este projecto dividiu-se em duas partes: o circuito desenvolvido em *handel-C* (*FPGA side*) e um programa devolvido em C++/QT (*PC side*).

Através do programa do *PC* podemos criar vários ficheiros de teste, que podem ser transferidos para a memória *flash* da *FPGA*. Essa transferência pode ser efectuada através do programa *FTU*. Esta arquitectura está ilustrada na *Figura 1*.

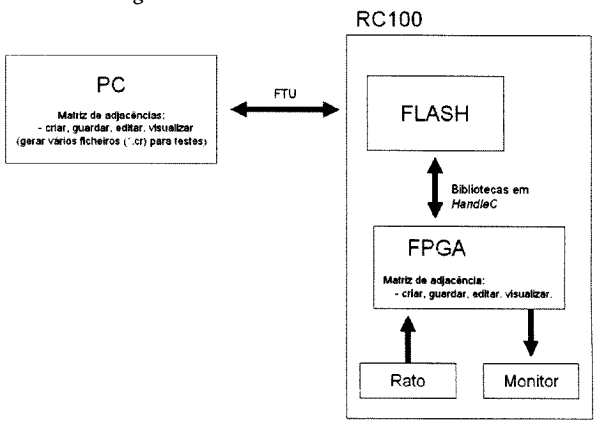

Figura 1 – Arquitectura global do projecto.

Já na *Figura 2* podemos ver os dispositivos usados que possibilitaram a visualização e edição do grafo sob a forma da sua matriz de adjacência.

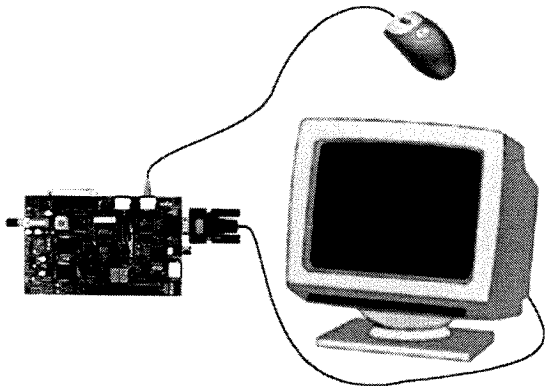

Figura 2 – Placa RC100 e dispositivos usados para a interacção

A *Figura 3* demonstra o funcionamento do *software* quer a nível da visualização, quer a nível de interacção com o utilizador.

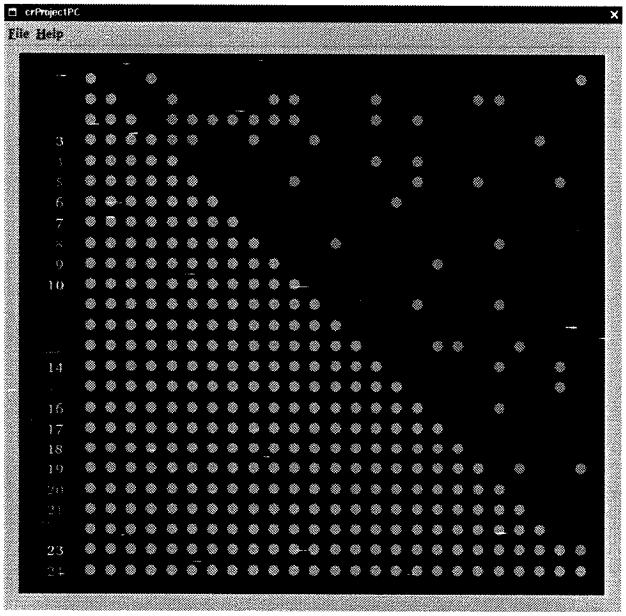

Figura 3 – Programa desenvolvido para o PC.

III. ESTRATÉGIAS ADOPTADAS

Para tirar partido de uma das principais características de *FPGA*, de se poder executar mais de uma instrução em simultâneo, foi adoptada a seguinte estratégia. Partilhar as variáveis (edgeMatrix, colorVector) entre os vários processos que estão a correr em paralelo. O esquema de funcionamento e os principais blocos de funcionamento estão resumidos na **Figura 4**.

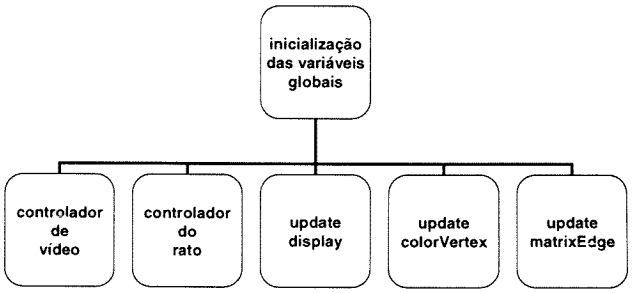

Figura 4 – Principais blocos de funcionamento

No *main* temos a correr em paralelo, cinco procedimentos expansíveis *in-line*: controlo do vídeo e do rato, ambas são bibliotecas [6] da *RC100*, controlo do *display*, controlo do vector onde está armazenada a informação sobre a cor de cada vértice e controlo da matriz de adjacências. Este funcionamento encontra-se ilustrado no bloco de código da **Figura 5.**

```
par
{
    RC100VideoDriver(&Video);
    RC100PS2MouseDriver(&Mouse, RC100_MOUSE_PORT);
    Display( &Video , &Mouse );
    UpdateColorVector();
    UpdateMatrixEdge( &Mouse );
}
```

Figura 5 – Bloco de código

No bloco de código descrito na **Figura 6** podemos ver esboço do algoritmo usado para seleccionar a cor de cada vértice.

```
for( int y = 0 ; y < 25 ; y++ )
    for( int x = y+1 ; x < 25 ; x++ )
        if( edgeMatrix[x][y] == '1' )
            if( colorVector[y] == colorVector[x] )
                colorVector[x]++;
```

Figura 6 – Bloco de código

Para o armazenamento da informação relativa à cor de cada vértice for usado um array *25* elementos (porque os grafos tem 25 vértices) de *unsigned int 24* porque cada *pixel* tem uma profundidade de 24 *bits*. A representação da matriz de adjacências do grafo é implementada através de um *array* de 300 elementos de *unsigned int 1*. Os 300 elementos são suficientes visto que a diagonal principal e o triângulo inferior não são usados, visto que representam arestas repetidas e arestas de um vértice para ele mesmo. A actualização da cor dos vértices é feita através do procedimento updateColorVertex que com base na variável edgeMatrix (implementação da matriz de adjacências). Já a edgeMatrix é alterada com base no *click* do rato, com um simples *click* podemos alternar entre 0 e 1, ou seja, alternamos entre "*há aresta*" e "*não há aresta*".

IV. CONCLUSÕES

A linguagem *Handel-C* revelou-se poderosa, bem como fácil de aprender. Apesar de uma grande optimização nas estruturas de dados, pôde-se comprovar que a implementação de memórias e estruturas de dados grandes devem ser postas nos blocos de *RAM* externos à *FPGA,* porque quando estão implementados na *FPGA* ocupam de uma forma exponencial as *slices* disponíveis. O uso da memória *RAM* iria contribuir para uma menor utilização dos recursos da *FPGA*, que de 89% passaria para 17% de utilização de *slices.*

AGRADECIMENTOS

O autor agradece ao Professor Valery Sklyarov e pela ajuda prestada na elaboração deste artigo.

REFERÊNCIAS

[1] Site: http://www.celoxica.com, on-line Agosto de 2004

[2] Site: http://webct.ua.pt, 2º Semestre, disciplina Computação Reconfigurável, on-line Agosto de 2004

[3] "Handel-C Language Reference Manual", Celoxica, 2003

[4] Site: http://www.trolltech.com, on-line Agosto de 2004

[5] "RC100 Hardware Manual", Celoxica, 2003

[6] "RC100 Function Library Reference", Celoxica, 2003