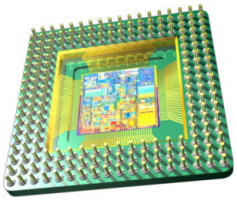

Lab 10 – Utilização do MIPS como sistema embutido numa FPGA

Neste trabalho de laboratório vamos continuar a trabalhar com o MIPS descrito em VHDL e implementado na FPGA Quartus II[1], introduzido no trabalho anterior, usando-o como um sistema embutido em circuitos digitais implementados na FPGA.

1. MIPS embutido utilizado como contador

No projecto dado, existe um esquemático MIPS_EMB que recorre ao módulo TOP_MIPSIO para embutir o MIPS num circuito digital (fig. 6).

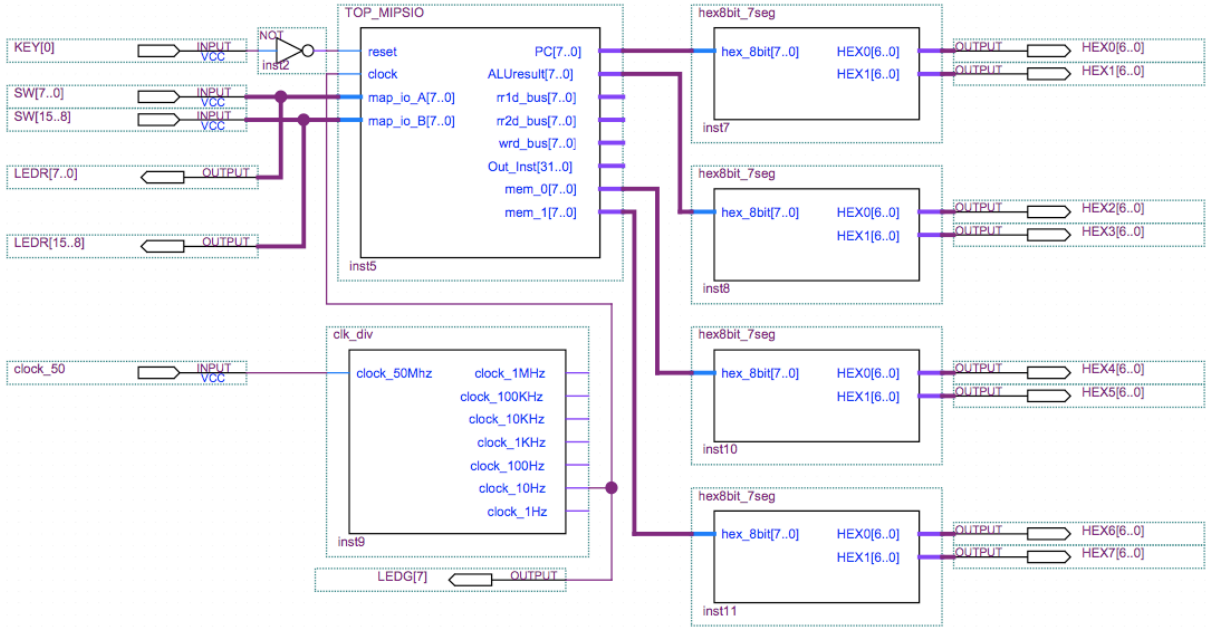

Fig. 6. Projecto com módulo TOP_MIPS_IO.VHD e componentes do UP Core.
O Módulo TOP_MIPSIO tem dois portos de entrada e saída da memória interna.

Neste esquemático o MIPS tem dois portos de entrada ligados aos interruptores da placa, e o PC, ALU, e portos de saída (MEM[0],MEM[1]) ligados a visores de 7 segmentos.

[1] Altera Quartus II disponível em: http://www.deec.uc.pt/~jlobo/altera/

Para carregar um registo com o estado dos interruptores basta ler as posições de memória 2 e 3, que se encontra "mapeada" para os portos A e B). Por exemplo fazendo:

```
lw    $6, 2($0)
lw    $7, 3($0)
```

ficamos com $6=SW[7..0] e $7=SW[15..8]. Para escrever nos portos de saída ligados aos visores de 7 segmentos basta escrever nas células de memoria MEM[0] e MEM[1], i.e:

```
sw    $6, 0($0)
sw    $7, 1($0)
```

Adapte o código feito no ponto 4 do Trabalho Laboratorial 10 para, com base no esquemático TOP_MIPS, por um MIPS a contar de 0-F no visor de 7 segmentos, com o outro a indicar o PC.

2. Luz do pátio com MIPS embutido

Programe um MIPS para integrar um circuito digital para controlar a iluminação do pátio de uma residência. A luz só deve acender quando: a entrada de um sensor de luz D é **0** (indicando que não é de dia), e a entrada de um sensor de movimento M é **1** (indicando movimento). Temos ainda a entrada de um interruptor I que quando é **1** indica que a luz deve ficar ligada, independentemente dos valores de D e M. Para ligar a luz deve-se colocar a saída L a **1**. A figura 7 mostra o diagrama de blocos do sistema a implementar.

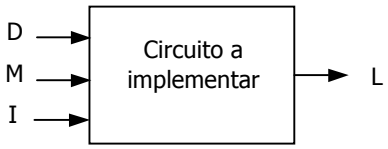

Fig. 7. Diagrama de blocos do controlador da luz do pátio.

O primeiro passo do projecto é capturar o comportamento do circuito por uma equação Booleana, que indica o valor da saída em função das entradas. Este comportamento pode ser descrito através da equação Booleana L=M.D'+I, em que D' representa "não D" (não é de dia), '.' E (*AND*) e '+' OU (*OR*).

O segundo passo é identificar o mapeamento entre as entradas e saídas do circuito e as entradas e saídas do processador. Neste caso deve considerar D=SW[0], M=SW[1] e I=SW[2], ou seja o porto A mapeado no endereço 2 fica com D,M e I nos 3 bits menos significativos, e a saída L fica ligada ao bit menos significativo do endereço de memória 0.

O terceiro passo é escrever um programa para o processador que assegure o funcionamento desejado, ficando em ciclo infinito a ler as entradas e escrever nas saídas.

3. MIPS embutido como temporizador I

A partir do código feito no ponto 1, programe o MIPS para contar de 0 a 9, com cada iteração do contador a demorar 10 ciclos de processador, e com o botão SW[0] a funcionar como *start/stop* do contador. Num novo esquemático[2], coloque um MIPS embutido a contar segundos controlado por SW[0], mantendo o *reset* (geral) em KEY[0].

4. MIPS embutido como temporizador II

Num novo esquemático, coloque dois MIPS embutidos a executar o mesmo código, cada um a contar uma casa decimal, para ter uma contagem de dois dígitos nos visores de 7 segmentos. Deve utilizar as saídas convenientes do módulo CLOCK_DIV para cada um dos processadores para contar segundos e décimas de segundo.

5. MIPS embutido como temporizador III *(opcional)*

Altere a programação dos processadores para ter contagem ascendente ou descendente em função de SW[0].

[2] não esquecer de indicar o novo esquemático como o principal do seu projecto (top-level entity), seleccionando essa opção com o botão do lado direito do rato sobre o ficheiro respectivo na janela do Project Navigator.

DE2_pin_assignments.csv

```
# Altera's DE2 board
# Cyclone II
# EP2C35F672C6
#
To,Location
# push-buttons
KEY[0],PIN_G26
KEY[1],PIN_N23
KEY[2],PIN_P23
KEY[3],PIN_W26
# Switches
SW[0],PIN_N25
SW[1],PIN_N26
SW[2],PIN_P25
SW[3],PIN_AE14
SW[4],PIN_AF14
SW[5],PIN_AD13
SW[6],PIN_AC13
SW[7],PIN_C13
SW[8],PIN_B13
SW[9],PIN_A13
SW[10],PIN_N1
SW[11],PIN_P1
SW[12],PIN_P2
SW[13],PIN_T7
SW[14],PIN_U3
SW[15],PIN_U4
SW[16],PIN_V1
SW[17],PIN_V2
# Red LEDs
LEDR[0],PIN_AE23
LEDR[1],PIN_AF23
LEDR[2],PIN_AB21
LEDR[3],PIN_AC22
LEDR[4],PIN_AD22
LEDR[5],PIN_AD23
LEDR[6],PIN_AD21
LEDR[7],PIN_AC21
LEDR[8],PIN_AA14
LEDR[9],PIN_Y13
LEDR[10],PIN_AA13
LEDR[11],PIN_AC14
LEDR[12],PIN_AD15
LEDR[13],PIN_AE15
LEDR[14],PIN_AF13
LEDR[15],PIN_AE13
LEDR[16],PIN_AE12
LEDR[17],PIN_AD12
# Green LEDs
LEDG[0],PIN_AE22
LEDG[1],PIN_AF22
LEDG[2],PIN_W19
LEDG[3],PIN_V18
LEDG[4],PIN_U18
LEDG[5],PIN_U17
LEDG[6],PIN_AA20
LEDG[7],PIN_Y18
LEDG[8],PIN_Y12
#
# 7 segments
# hex0
HEX0[0],PIN_AF10
HEX0[1],PIN_AB12
HEX0[2],PIN_AC12
HEX0[3],PIN_AD11
HEX0[4],PIN_AE11
HEX0[5],PIN_V14
HEX0[6],PIN_V13
# hex1
HEX1[0],PIN_V20
HEX1[1],PIN_V21
HEX1[2],PIN_W21
HEX1[3],PIN_Y22
HEX1[4],PIN_AA24
HEX1[5],PIN_AA23
HEX1[6],PIN_AB24
# hex2
HEX2[0],PIN_AB23
HEX2[1],PIN_V22
HEX2[2],PIN_AC25
HEX2[3],PIN_AC26
HEX2[4],PIN_AB26
HEX2[5],PIN_AB25
HEX2[6],PIN_Y24
# hex3
HEX3[0],PIN_Y23
HEX3[1],PIN_AA25
HEX3[2],PIN_AA26
HEX3[3],PIN_Y26
HEX3[4],PIN_Y25
HEX3[5],PIN_U22
HEX3[6],PIN_W24
# hex4
HEX4[0],PIN_U9
HEX4[1],PIN_U1
HEX4[2],PIN_U2
HEX4[3],PIN_T4
HEX4[4],PIN_R7
HEX4[5],PIN_R6
HEX4[6],PIN_T3
# hex5
HEX5[0],PIN_T2
HEX5[1],PIN_P6
HEX5[2],PIN_P7
HEX5[3],PIN_T9
HEX5[4],PIN_R5
HEX5[5],PIN_R4
HEX5[6],PIN_R3
# hex6
HEX6[0],PIN_R2
HEX6[1],PIN_P4
HEX6[2],PIN_P3
HEX6[3],PIN_M2
HEX6[4],PIN_M3
HEX6[5],PIN_M5
HEX6[6],PIN_M4
# hex7
HEX7[0],PIN_L3
HEX7[1],PIN_L2
HEX7[2],PIN_L9
HEX7[3],PIN_L6
HEX7[4],PIN_L7
HEX7[5],PIN_P9
HEX7[6],PIN_N9
# 50MHz clock
CLOCK_50,PIN_N2
# PS2
PS2_CLK,PIN_D26
PS2_DAT,PIN_C24
# VGA
VGA_R[9],PIN_E10
VGA_G[9],PIN_D12
VGA_B[9],PIN_B12
VGA_HS,PIN_A7
VGA_VS,PIN_D8
VGA_CLK, PIN_B8
VGA_BLANK, PIN_D6
```