Memoria practica 3

Daniel Vilardell
Igor Yuziv

Índex

1 Part 1 **2**
1.1 Diseny Jerarquic . . . 2
1.2 ComptadorBCD . . . 3
1.3 ComparadorBCD . . . 4
1.4 Control . . . 5
1.5 Registres . . . 8
1.6 keygroup . . . 9
1.7 Leds . . . 11
1.8 Joc . . . 12
1.9 Joc Placa . . . 14

2 Part Extra **16**
2.1 ComptadorBCD Extra . . . 16
2.2 Registres Extra . . . 18
2.3 Keygroup Extra . . . 19
2.4 Control Extra . . . 21
2.5 Trampes . . . 23
2.6 Leds Extra . . . 24
2.7 Temporitzador . . . 25
2.8 Slow Timer . . . 27
2.9 Joc Extra . . . 28
2.10 Joc Extra Placa . . . 30

1 Part 1

1.1 Diseny Jerarquic

El diseny te l'objectiu de implementar un joc que es basa en endivinar nombres aleatoris amb la informacio de si el nombre entrat es mes gran o mes petit al nombre buscat. Un cop s'endevina s'ha de mostrar el nombre i es pot tornar a començar. Per tal de fer aixo hem creat un programa principal que consta dels seguents components: keygroup, control, comptadorBCD, comparadorBCD i regs_v.

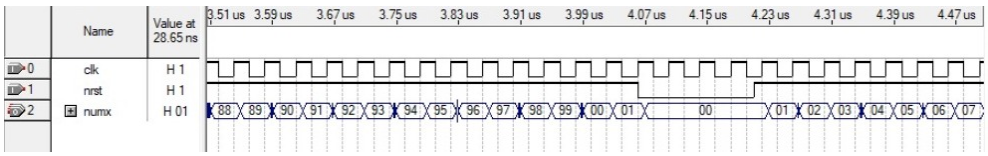
Quan es clica una tecla de la placa ho rep keygroup que informa a control si la entrada es un nombre, un asterisc o un coixinet. Control decideix, en funció del moment en que es trobi de la partida(inici, introduccio de dades o final) que fa. En el cas de estar en inici i si s'entra un asterisc el joc comença i es selecciona un nombre aleatori que no es mostra indicant a comptador que pari de contar. Despres si s'entren nombres es van emmagatzemant dins de regs_v i es van comparant amb el nombre introduit. Nomes quan es clica la tecla coixinet es mostra si el nombre es mes gran o no al buscat.

El component control es l'encarregat de gestionar les fases del joc i decidir que fer en cada moment mentres que comptador es el que decideix el nombre aleatori i comparador retorna si un nombre es mes gran que un altre, i regs de emmagatzemar els nombres introduits.

1.2 ComptadorBCD

El component comptador es el encarregat de decidir el nombre aleatori. Ho fa de la següent manera: Conte un contador que va augmentant cada flanc de pujada de clk, per tant molt rapidament, i nomes s'atura si ecnt es 0. Com que compta molt rapidament, podem "assegurar"que el nombre escollit sera aleatori. La entrada ecnt la rebrà de control en el moment en que estiguem a fase inicial i es pitgi la tecla asterisc.

```
                library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;

entity comptadorBCD is
        port (nrst, clk, ecnt : in std_logic;
                numx : out std_logic_vector(7 downto 0));
end comptadorBCD;

architecture compte of comptadorBCD is
        signal unitats, desenes : std_logic_vector (3 downto 0);

begin
        process(clk, nrst)
        begin
            if nrst = '0' then desenes <= "0000";
                        unitats <= "0000";
            elsif clk' event and clk='1' then
                if ecnt = '1' then
                if desenes = "1001" and unitats = "1001" then desenes <= "0000";
                        unitats <= "0000";
                elsif unitats = "1001" then desenes <= desenes +1;
                        unitats <= "0000";
                else unitats <= unitats+1;
                end if;
            end if;
        end if;
end process;
numx (7 downto 4) <= desenes;
numx (3 downto 0) <= unitats;

end compte;
```



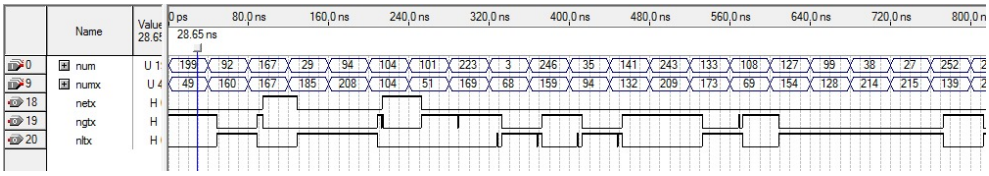
Podem veure que la simulació ja que quan arriba a 99 torna a començar al 0 i segueix contant.

Flow Status	Successful - Tue Dec 01 16:10:28 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	comptadorBCD
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	12 / 33,216 (< 1 %)
Total combinational functions	12 / 33,216 (< 1 %)
Dedicated logic registers	8 / 33,216 (< 1 %)
Total registers	8
Total pins	11 / 475 (2 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.3 ComparadorBCD

El bloc comparador també es ben senzill, el que fa es mirar entre dos nombres d'entrada quin es mes gran i donar això a la sortida. Si num > numx aleshores ngtx (n greater than x) serà 1, sin num < numx nltx (n less than x) serà 1 i finalment si son iguals netx(number equal to x) serà 1.

```
        library ieee;
use ieee.std_logic_1164.all;

entity comparadorBCD is
        port (numx,num : in std_logic_vector (7 downto 0);
                        ngtx,nltx,netx: out std_logic);
end comparadorBCD;

architecture comparador of comparadorBCD is
        begin
        ngtx <= '1' when num > numx else '0';
        netx <= '1' when num = numx else '0';
        nltx <= '1' when num < numx else '0';
end comparador;
```



Podem veure que es comporta com hem comentat que faria. En el nostre joc els nombres seran de nomes dos digits, tot i això el component funciona per alguns nombres de 3 digits.

Flow Status	Successful - Thu Dec 17 12:07:40 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	comparadorBCD
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	21 / 33,216 (< 1 %)
Total combinational functions	21 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	19 / 475 (4 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.4 Control

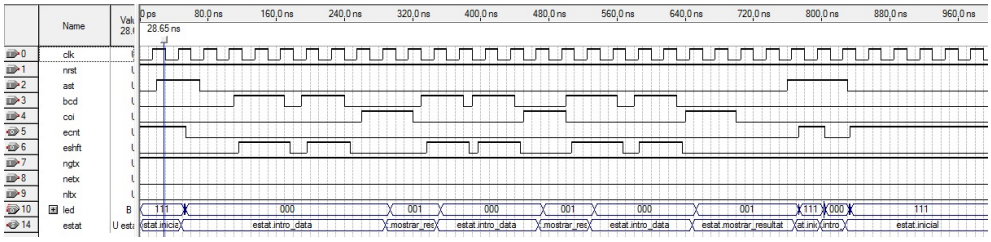
El bloc control es l'encarregat de organitzar les fases del joc i el mes complicat de tots. Aquest bloc conté tres estats. Inicial, Introducció de

dades i mostrar resultats. Com a entrades rep si la tecla premuda es un asterisc, un nombre bcd o el coixinet junt amb la relació entre els nombres que s'estan comparant.

- **Inicial:** Per defecte la sortida que indica al contador si seguir contant esta desactivada. Si la entrada es un asterisc indica la atura del comptador posant el bit ecnt a 0 i també canvia l'estat a introduccio de dades.
- **Introducció de dades:** En aquest estat si la entrada es un nombre bcd, s'indica via la sortida eshft que s'actualitzi el valor del digit dins de regs, on s'emmagatzemen aquests. Si es clica l'asterisc es torna al estat inicial. Si es clica coixinet s'indica a partir de la sortida led i amb la informacio de les entrades ngtx nltx netx si el valor introduit es major, menor o igual i es canvia el estat a mostrar resultats.
- **Mostrar resultats:** En aquest estat control no gestiona res, simplement indica a la sortida eshft que s'han de mostrar els resultats per pantalla (si es major, menor o igual).

```
                library ieee;
use ieee.std_logic_1164.all;

entity control is
        port (nrst,clk,bcd,ast,coi,ngtx,netx,nltx : in std_logic;
                ecnt, eshft : out std_logic;
                led : out std_logic_vector(2 downto 0));
        end control;

architecture arcControl of control is
        type maquina is (inicial, intro_data, mostrar_resultat);
        signal estat: maquina;
begin
        process(clk, nrst) begin
        if nrst = '0' then estat <= inicial;
        elsif (clk'event and clk ='1') then
            case estat is
            when inicial => if ast = '1' then estat <= intro_data; end if;
            when intro_data => if coi = '1' and ast = '0' then estat <= mostrar_resultat;
                        elsif ast = '1' then estat <= inicial; end if;
            when mostrar_resultat => if ast = '1' then estat <= inicial;
                                elsif netx = '1' then estat <= mostrar_resultat;
                                elsif bcd = '1' then estat <= intro_data;
                                end if;
            end case;
        end if;
end process;

ecnt <= '1' when estat = inicial else '0';
eshft <= '1' when (estat = intro_data or estat = mostrar_resultat) and bcd = '1' else '0';

led <= "111" when estat = inicial
    else "100" when estat = mostrar_resultat and nltx = '1'
        and netx= '0' and ngtx = '0'
    else "010" when estat = mostrar_resultat and netx = '1'
        and nltx = '0' and ngtx = '0'
    else "001" when estat = mostrar_resultat and ngtx = '1'
```

```
        and nltx = '0' and netx = '0'
    else "000";
end arcControl;
```



Flow Status	Successful - Thu Dec 17 12:13:17 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	control
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	11 / 33,216 (< 1 %)
Total combinational functions	11 / 33,216 (< 1 %)
Dedicated logic registers	3 / 33,216 (< 1 %)
Total registers	3
Total pins	13 / 475 (3 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.5 Registres

Registres funciona igual que a la ultima practica, es un mòdul seqüencial síncron que te com a finalitat carregar i memoritzar els digits introduits opA i opB. Aquest, si la entrada intro es 1 i clk esta en el flanc de pujada i nrst esta activat, actualitzara els valors de opA i opB, posant a opA el valor entrat per keycode i a opB el antic valor de opA.

```
library ieee;
use ieee.std_logic_1164.all;

entity regs_v is
        port(clk, nrst, intro : in std_logic;
                keycode : in std_logic_vector(3 downto 0);
                opA, opB: out std_logic_vector(3 downto 0));
end regs_v;

architecture arq of regs_v is
signal a, b : std_logic_vector(3 downto 0);
begin
process (clk, nrst)
begin
        if(nrst = '0') then a <= "0000"; b <= "0000";
        elsif(nrst='1') and (clk'event and clk = '1') and (intro = '1') then
                        b <= a;
                        a <= keycode;
        end if;
end process;
opA <= a;
opB <= b;

end arq;
```

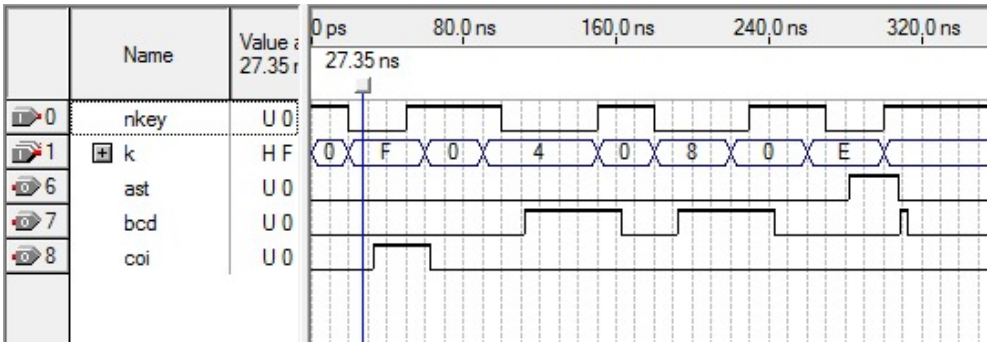
Flow Status	Successful - Thu Dec 17 18:29:47 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	regs_v
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	9 / 33,216 (< 1 %)
Total combinational functions	1 / 33,216 (< 1 %)
Dedicated logic registers	8 / 33,216 (< 1 %)
Total registers	8
Total pins	15 / 475 (3 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.6 keygroup

Aquest component també es igual a la practica anterior i te com a finalitat indicarnos si la tecla premuda es un nombre bcd, un asterisc o un coixinet.

```
        library ieee;
use ieee.std_logic_1164.all;

entity keygroup_v is
        port(nkey : in std_logic;
                k : in std_logic_vector(3 downto 0);
                bcd, ast, coi : out std_logic);
end keygroup_v;

architecture arq of keygroup_v is
begin
process (nkey, k)
begin
        if (nkey = '0' and (k = "0000" or k = "0001" or k = "0010" or
                                k = "0011" or k = "0100" or k = "0101" or
                                k = "0110" or k = "0111" or k = "1000" or
                                k = "1001")) then bcd <= '1'; ast <= '0'; coi <= '0';
        elsif(nkey = '0' and k = "1110")
                        then bcd <= '0'; ast <= '1'; coi <= '0';
        elsif(nkey = '0' and k = "1111")
                        then bcd <= '0'; ast <= '0'; coi <= '1';
```

```
        elsif(nkey = '1') then bcd <= '0'; ast <= '0'; coi <= '0';
        end if;
end process;
end arq;
```



Veiem que funciona ja que s'activa coi quan la entrada es F, bcd quan la entrada es 4 i 8 i ast quan la entrada es E.

Flow Status	Successful - Thu Dec 17 18:26:49 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	keygroup_v
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	10 / 33,216 (< 1 %)
Total combinational functions	10 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	8 / 475 (2 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.7 Leds

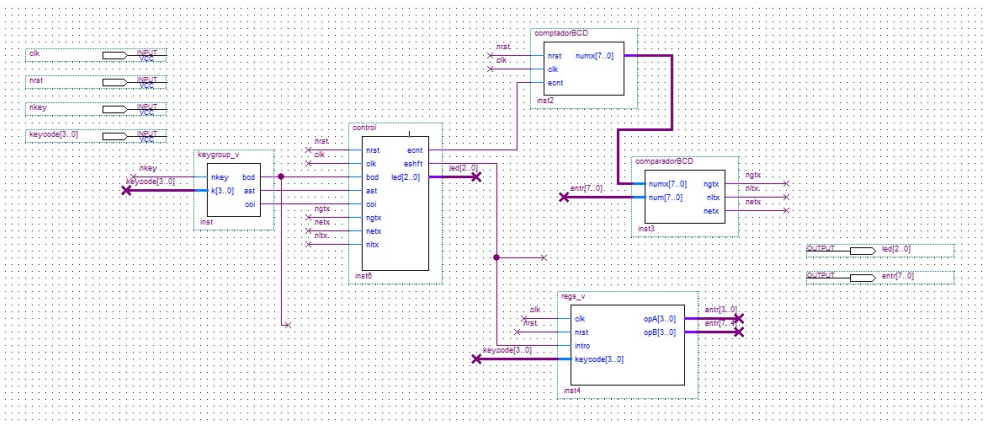
El component leds passa la informació que rep del programa principal a la representació esperada per a la practica, es a dir dels tres bits a la representació de 8 bits que iluminara 8 leds.

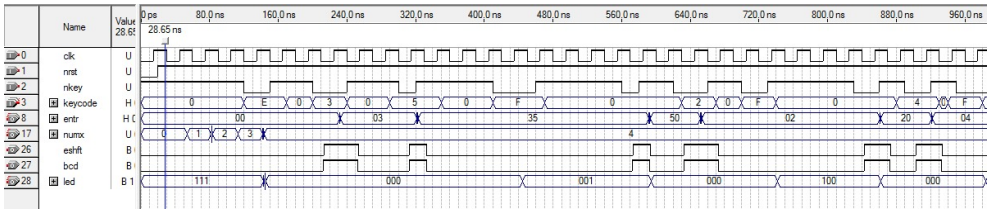
```
        library ieee;
use ieee.std_logic_1164.all;

entity leds is
        port(led: in std_logic_vector(2 downto 0);
                led_green : out std_logic_vector(7 downto 0));
end leds;

architecture arq of leds is
begin
        led_green <= "11111111" when led = "111" else
                                "11110000" when led = "100" else
                                "00001111" when led = "001" else
                                "00111100" when led = "010" else
                                "00000000" when led = "000";

end arq;
```

Flow Status	Successful - Thu Dec 17 21:11:47 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	leds
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	16 / 33,216 (< 1 %)
Total combinational functions	16 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	11 / 475 (2 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.8 Joc

Aquest es el programa principal breument explicat al inici de la practica. Consta de tots els components mencionats anteriorment i els ajunta per a controlar la part principal del joc. La entrada es passa per keygroup que ens diu quin tipus d'entrada es, que s'envia a control. En funció de en quin estat control es trobi aturara el contador, mostrara si el valor introduit es mes gran o igual o mes petit o actialitzarà el valor de regs. Com a entrades rebrà si s'esta clicant una tecla, quina tecla s'esta pitjant, nrst i clk. Com a sortida indicarà quins leds iluminar junt amb el nombre introduit per l'usuari.





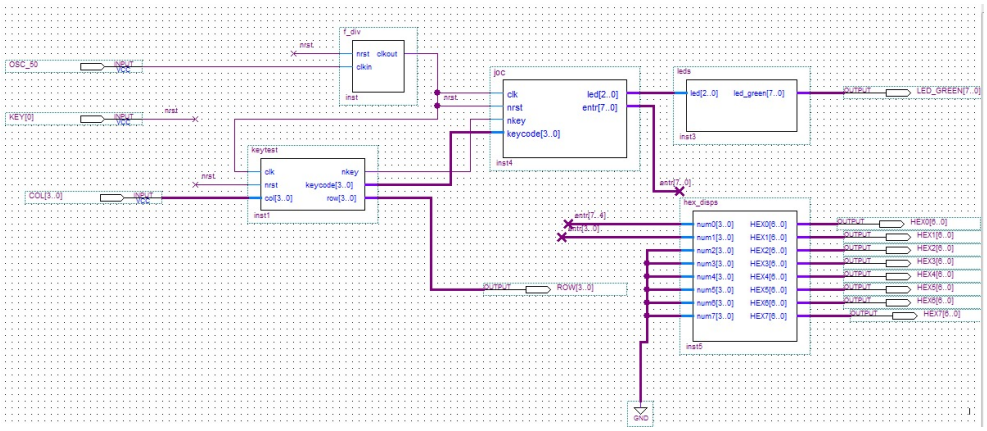
Podem veure que funciona ja que estem simulant la seguent partida: Primer es prem la tecla E que ens comença la partida i fixa el nombre numx a endevinar. Despres s'introdueixen els nombres 3 i 5 que formen el 35 dins de entr. Es prem la tecla F i la sortida led es 001, per tant indica que el nombre es massa gran. Despres s'introdueix el nombre 0 i dos que forma el 02 i es prem la tecla F. La sortida es 4, es a dir 100, que indica que el nombre es massa gran. Finalment prenem el nombre 04 i prenem la tecla F que ens mostra la sortida 010 i per tant esta encertat.

Flow Status	Successful - Thu Dec 17 18:36:42 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	joc
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	55 / 33,216 (< 1 %)
Total combinational functions	55 / 33,216 (< 1 %)
Dedicated logic registers	19 / 33,216 (< 1 %)
Total registers	19
Total pins	18 / 475 (4 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

1.9 Joc Placa

Un cop ja hem testejat tots els components individualment amb simulacións ho ajuntem tot a un diagrama de blocs amb l'objectiu de mostrarho i interectuar a la placa.

Aquest diagrama funciona de la seguent manera, keytest s'encarrega de indicar quina tecla s'està prement i la introdueix al programa principal indicat anteriorment. Aquest ens indicara el nombre entrat per l'usuari a mostrar per la sortida entr i els leds a iluminar led. El component leds que s'encarregara de transformar la sortida led de 3 bits a una sortida de 7 que indicarà els leds a iluminar. El component hex_disp transforma els nombres de la entrada a 7 segments per a mostrar a la placa els valors.


f_div
nrst
clkout
clkin
inst
OSC_50
INPUT
KEY[0]
INPUT
nrst
keytest
clk
nrst
col[3..0]
nkey
keycode[3..0]
row[3..0]
inst1
COL[3..0]
INPUT
nrst
joc
clk
nrst
nkey
keycode[3..0]
led[2..0]
entr[7..0]
inst4
leds
led[2..0]
led_green[7..0]
inst3
OUTPUT
LED_GREEN[7..0]
entr[7..0]
hex_disps
entr[7..4]
entr[3..0]
num0[3..0]
num1[3..0]
num2[3..0]
num3[3..0]
num4[3..0]
num5[3..0]
num6[3..0]
num7[3..0]
HEX0[6..0]
HEX1[6..0]
HEX2[6..0]
HEX3[6..0]
HEX4[6..0]
HEX5[6..0]
HEX6[6..0]
HEX7[6..0]
inst5
OUTPUT
ROW[3..0]
GND

2 Part Extra

Un cop tenim el joc basic hem decidit ampliarlo de la seguent forma. Primer hem introduit un altre digit al joc, es a dir, en contes d'adivinar un nombre de 2 digits s'ha d'endivinar un nombre de 3. Com que això podria arribar a ser aburrit hem havilitat una forma de fer trampes: pitjant la tecla B es pot veure el valor de les desenes del nombre buscat i la tecla A per a veure les unitats. Aixi doncs nomes faltaria un nombre per a adivinar, el de les centenes. A mes d'això hem havilitat un sistema que fa un conta enrere i un contador de punts a base de leds per tal de competir a contrarellotge i intentar fer els maxims punts en un temps donat.

Per a fer aquesta part hem hagut de fer moltes modificacions a la primera part del progecte, algunes d'aquestes son les seguents. Hem modificat contador per a que conti fins a les centenes, també hem modificat keygroup per a admetre les entrades A i B. A regs ara hi guardarem 3 nombres, unitats, desenes i centenes. Control tindrà ara dos nous estats, trampaA i trampaB, que indicaran si es mostra el digit de les desenes o el de les centenes. A leds se li afegirà el control de punts que es porta que ens marcarà quants leds iluminar i quins en funció de la entrada. També crearem un nou modul trampa que treurà el nombre a mostrar en funció de si estem en estat trampaA o trampaB. Aprofitant el modul f_div crearem un component slow_timer que ens permetrà ampliar el temps de rellotge fins a segons i aixi poder, amb temporitzador, que internament funciona igual que un contador, poder fer el contador de segons. Finalment ho posarem tot dins de un programa principal que gestionara gran part de les funcions.

Per a posarho a la placa aprofitarem el component anterior afegintli algunes variacions com el slow timer junt amb el temporitzador o el modul trampes.

2.1 ComptadorBCD Extra

Aquest contador funciona igual que el anterior pero s'ha afegit una variable per a contar les centenes, fent aixi un contador de 3 xifres.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;

entity comptadorBCD_extra is
        port (nrst, clk, ecnt : in std_logic;
                numx : out std_logic_vector(11 downto 0));
end comptadorBCD_extra;
```

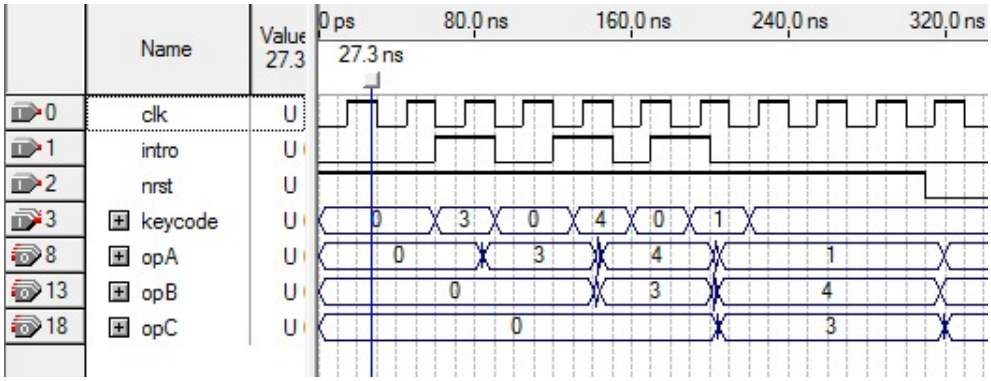
```
architecture compte of comptadorBCD_extra is
        signal unitats, desenes, centenes : std_logic_vector (3 downto 0);

begin
        process(clk, nrst)
        begin
            if nrst = '0' then desenes <= "0000";
                        unitats <= "0000"; centenes <= "0000";
            elsif clk' event and clk='1' then
                if ecnt = '1' then
                if desenes = "1001" and unitats = "1001" and centenes = "1001"
                        then desenes <= "0000"; unitats <= "0000"; centenes <= "0000";
                elsif desenes = "1001" and unitats = "1001" then centenes <= centenes + 1;
                        desenes <= "0000"; unitats <= "0000";
                elsif unitats = "1001" then desenes <= desenes + 1;
                        unitats <= "0000";
                else unitats <= unitats+1;
                end if;
            end if;
        end if;
end process;
numx (11 downto 8) <= centenes;
numx (7 downto 4) <= desenes;
numx (3 downto 0) <= unitats;

end compte;
```

Flow Status	Successful - Thu Dec 17 19:05:57 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	comptadorBCD_extra
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	20 / 33,216 (< 1 %)
Total combinational functions	20 / 33,216 (< 1 %)
Dedicated logic registers	12 / 33,216 (< 1 %)
Total registers	12
Total pins	15 / 475 (3 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.2 Registres Extra

Aquest component es el mateix que el anterior el que se li ha afegit una variable de sortida opC per a guardar el 3r valor.

```
        library ieee;
use ieee.std_logic_1164.all;

entity regs_extra is
        port(clk, nrst, intro : in std_logic;
                keycode : in std_logic_vector(3 downto 0);
                opA, opB, opC: out std_logic_vector(3 downto 0));
end regs_extra;

architecture arq of regs_extra is
signal a, b, c : std_logic_vector(3 downto 0);
begin
process (clk, nrst)
begin
        if(nrst = '0') then a <= "0000"; b <= "0000"; c <= "0000";
        elsif(nrst='1') and (clk'event and clk = '1') and (intro = '1') then
                        c <= b;
                        b <= a;
                        a <= keycode;
        end if;
end process;
opA <= a;
opB <= b;
opC <= c;

end arq;
```



Podem veure que funciona ja que ens guarda els tres valors a mida que es van introduint.

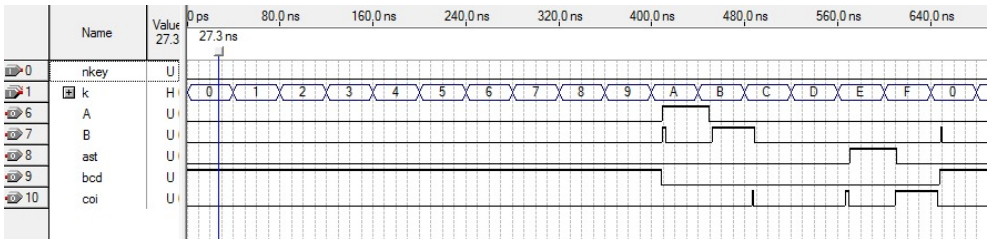
Flow Status	Successful - Thu Dec 17 19:01:16 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	regs_extra
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	13 / 33,216 (< 1 %)
Total combinational functions	1 / 33,216 (< 1 %)
Dedicated logic registers	12 / 33,216 (< 1 %)
Total registers	12
Total pins	19 / 475 (4 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.3 Keygroup Extra

Funciona igual que el anterior, nomes se li ha afegit la entrada A i la entrada B

```
library ieee;
use ieee.std_logic_1164.all;

entity keygroup_extra is
        port(nkey : in std_logic;
                k : in std_logic_vector(3 downto 0);
                bcd, ast, coi, A, B : out std_logic);
end keygroup_extra;

architecture arq of keygroup_extra is
begin
process (nkey, k)
begin
        if (nkey = '0' and (k = "0000" or k = "0001" or k = "0010" or
                k = "0011" or k = "0100" or k = "0101" or
                k = "0110" or k = "0111" or k = "1000" or
                k = "1001")) then bcd <= '1'; ast <= '0'; coi <= '0'; A <= '0'; B <= '0';
        elsif(nkey = '0' and k = "1010") then bcd <= '0'; ast <= '0'; coi <= '0';
                A <= '1'; B <= '0';
        elsif(nkey = '0' and k = "1011") then bcd <= '0'; ast <= '0'; coi <= '0';
                A <= '0'; B <= '1';
        elsif(nkey = '0' and k = "1110") then bcd <= '0'; ast <= '1'; coi <= '0';
                A <= '0'; B <= '0';
```

```
        elsif(nkey = '0' and k = "1111") then bcd <= '0'; ast <= '0'; coi <= '1';
                A <= '0'; B <= '0';
        elsif(nkey = '1') then bcd <= '0'; ast <= '0'; coi <= '0'; A <= '0'; B <= '0';
        else bcd <= '0'; ast <= '0'; coi <= '0'; A <= '0'; B <= '0';
        end if;
end process;
end arq;
```



Veiem que funciona ja que quan es prem la tecla A s'activa la sortida A i quan es prem la tecla B s'activa la sortida B.

Flow Status	Successful - Thu Dec 17 19:07:41 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	keygroup_extra
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	7 / 33,216 (< 1 %)
Total combinational functions	7 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	10 / 475 (2 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.4 Control Extra

La unica cosa que s'ha canviat en aquest component es que s'han afegit els estats trampaA i trampaB que s'entra pitjant les tecles A i B desde qualsevol altre estat. Si es pitja la tecla A la sortida led sera 110 i s'enviara a trampa per la entrada tr que indicara que es mostrin les unitats. Si es pitja la tecla B la sortida sera 011 i es mostraran les desenes.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity control_extra is
        port (nrst,clk,bcd,ast,coi,A,B,ngtx,netx,nltx : in std_logic;
                ecnt, eshft : out std_logic;
                led : out std_logic_vector(2 downto 0);
                tot : out std_logic_vector(3 downto 0));
        end control_extra;

architecture arcControl of control_extra is
        type maquina is (inicial, intro_data, mostrar_resultat, trampa_A, trampa_B);
```

```
        signal estat: maquina;
        signal t : std_logic_vector(3 downto 0);
begin
        process(clk, nrst) begin
        if nrst = '0' then estat <= inicial;
        elsif (clk'event and clk ='1') then
            case estat is
            when inicial => if ast = '1' then estat <= intro_data; end if;
            when intro_data => if coi = '1' and ast = '0' then estat <= mostrar_resultat;
                                    elsif ast = '1' then estat <= inicial;
                                    elsif A = '1' then estat <= trampa_A;
                                    elsif B = '1' then estat <= trampa_B; end if;
            when mostrar_resultat => if ast = '1' then estat <= inicial;
                                elsif bcd = '1' then estat <= intro_data;
                                elsif A = '1' then estat <= trampa_A;
                                elsif B = '1' then estat <= trampa_B;
                                end if;
                when trampa_A => if ast = '1' then estat <= inicial;
                                elsif bcd = '1' then estat <= intro_data;
                                elsif B = '1' then estat <= trampa_B;
                                end if;
                when trampa_B => if ast = '1' then estat <= inicial;
                                elsif bcd = '1' then estat <= intro_data;
                                elsif A = '1' then estat <= trampa_A;
                                end if;
                end case;
                if(netx = '1' and estat = intro_data)
                        then t <= t + 1; estat <= inicial; end if;
        end if;
end process;
tot <= t;

ecnt <= '1' when estat = inicial else '0';
eshft <= '1' when (estat = intro_data or estat = mostrar_resultat) and bcd = '1'
                else '0';

led <= "111" when estat = inicial
    else "100" when estat = mostrar_resultat and nltx = '1'
        and netx= '0' and ngtx = '0'
    else "010" when estat = mostrar_resultat and netx = '1'
        and nltx = '0' and ngtx = '0'
    else "001" when estat = mostrar_resultat and ngtx = '1'
        and nltx = '0' and netx = '0'
    else "110" when estat = trampa_A
    else "011" when estat = trampa_B
    else "000";
end arcControl;
```

Flow Status	Successful - Thu Dec 17 19:09:23 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	control_extra
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	50 / 33,216 (< 1 %)
Total combinational functions	50 / 33,216 (< 1 %)
Dedicated logic registers	9 / 33,216 (< 1 %)
Total registers	9
Total pins	19 / 475 (4 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.5 Trampes

El bloc trampes rep dos entrades, tr que indicara si mostrar les desenes o unitats i num, que ens dona el nombre en bcd de tres xifres. Tindrà una sortida, mostra que ens retornara el nombre que veiem al fer trampes. Si tr es 110 es mostrara les unitats i si tr es 011 es mostrara les desenes.

```
library ieee;
use ieee.std_logic_1164.all;

entity trampes is
        port(tr : in std_logic_vector(2 downto 0);
                num: in std_logic_vector(11 downto 0);
                mostra: out std_logic_vector(3 downto 0));
end trampes;

architecture arq of trampes is
begin
process (tr)
begin
        if tr = "110" then mostra <= num(3 downto 0);
        elsif tr = "011" then mostra <= num(7 downto 4);
        else mostra <= "0000";
        end if;
end process;
end arq;
```

Flow Status	Successful - Thu Dec 17 19:18:19 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	trampes
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	6 / 33,216 (< 1 %)
Total combinational functions	6 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	19 / 475 (4 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.6 Leds Extra

A leds rebrem el nombre de partides guanyades per la entrada num. En funció de això decidirem quins leds encendre i led_level variarà.

```
        library ieee;
use ieee.std_logic_1164.all;

entity led_extra is
        port(led: in std_logic_vector(2 downto 0);
                num : in std_logic_vector(3 downto 0);
                led_green : out std_logic_vector(7 downto 0);
                led_level : out std_logic_vector(7 downto 0));
end led_extra;

architecture arq of led_extra is
begin
        led_green <= "11111111" when led = "111" else
                                "11110000" when led = "100" else
                                "00001111" when led = "001" else
                                "00111100" when led = "010" else
                                "00000000" when led = "000" else
                                "01010101" when led = "110" else
                                "10101010" when led = "011";
        led_level <= "00000000" when num = "0000" else
```

```
                                    "10000000" when num = "0001" else
                                    "11000000" when num = "0010" else
                                    "11100000" when num = "0011" else
                                    "11110000" when num = "0100" else
                                    "11111000" when num = "0101" else
                                    "11111100" when num = "0110" else
                                    "11111110" when num = "0111" else
                                    "11111111" when num = "1000";

end arq;
```

Flow Status	Successful - Thu Dec 17 19:19:14 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	led_extra
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	43 / 33,216 (< 1 %)
Total combinational functions	43 / 33,216 (< 1 %)
Dedicated logic registers	0 / 33,216 (0 %)
Total registers	0
Total pins	23 / 475 (5 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.7 Temporitzador

El temporitzador tindra la mateixa estructura que un contador. Començara al nombre 99 amb les desenes a 9 i les unitats a 9, i anira descontant una unitat cada flanc de pujada de clk. Reusarem el codi de contador i per tant la entrada que decidia si contar o no hi posarem una font de tensió activa. Les sortides seran les desenes i les unitats.

```
        library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;

entity temporitzador is
        port (nrst, clk, ecnt : in std_logic;
                numx : out std_logic_vector(7 downto 0));
end temporitzador;

architecture compte of temporitzador is
        signal unitats, desenes : std_logic_vector (3 downto 0);

begin
        process(clk, nrst)
        begin
            if nrst = '0' then desenes <= "1001";
                        unitats <= "1001";
            elsif clk' event and clk='1' then
                if ecnt = '1' then
                if desenes = "0000" and unitats = "0000" then desenes <= "0000";
                        unitats <= "0000";
                elsif unitats = "0000" then desenes <= desenes -1;
                        unitats <= "1001";
                else unitats <= unitats-1;
                end if;
            end if;
        end if;
end process;
numx (7 downto 4) <= desenes;
numx (3 downto 0) <= unitats;

end compte;
```

Flow Status	Successful - Thu Dec 17 19:20:20 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	temporitzador
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	13 / 33,216 (< 1 %)
Total combinational functions	13 / 33,216 (< 1 %)
Dedicated logic registers	8 / 33,216 (< 1 %)
Total registers	8
Total pins	11 / 475 (2 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.8 Slow Timer

Aquest component es igual que f_div canviant la constant M, que es la que ajusta el que temps que reduim el clk. Ho hem ajustat empiricament mirant a la placa el nombre que ens donava una mesura de segon mes precisa que hem arribat a veure que es M = 750. La sortida clk la enviarem a temporitzador per a fer el conta enrere.

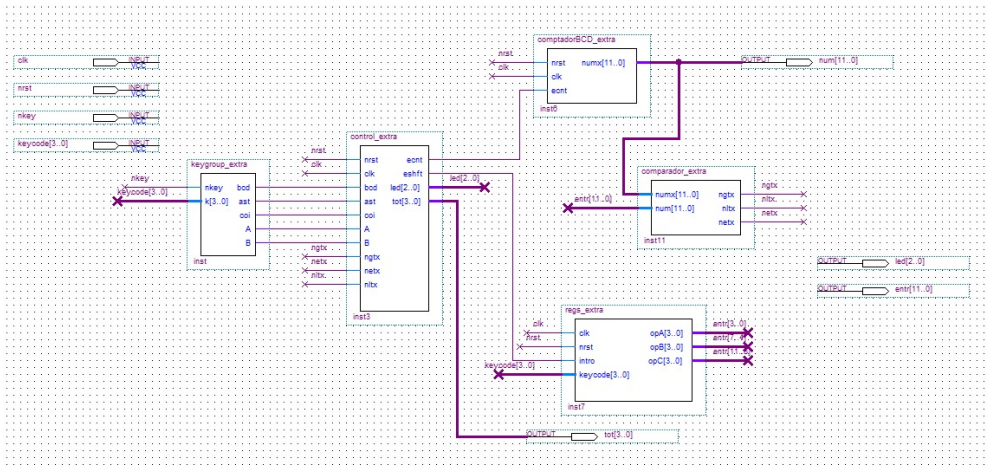
```
        -- Frequency divider by M
-- D = output duty cycle in %
-- version DD-1.0 - march 2011

library ieee;
use ieee.std_logic_1164.all;

entity slow_timer is
  port( nrst,clkin : in std_logic;
        clkout : out std_logic );
end slow_timer;

architecture dni of slow_timer is
constant M : integer :=750;
```

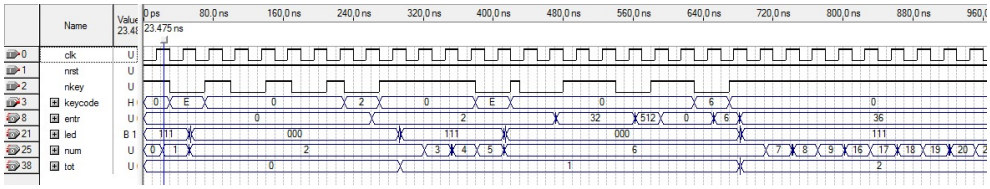
```
constant D : integer :=50;
constant n : integer :=D*M/100;
signal q : integer range 0 to M-1;
begin
  process(clkin,nrst) begin
     if nrst='0' then clkout <= '0'; q <= 0;
     elsif clkin'event and clkin='1' then
         if q < M-1 then q <= q+1;
            else q <= 0; end if;
         if q < n then clkout <= '1';
            else clkout <= '0'; end if;
     end if;
  end process;
end dni;
```

Flow Status	Successful - Thu Dec 17 19:31:34 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	slow_timer
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	16 / 33,216 (< 1 %)
Total combinational functions	16 / 33,216 (< 1 %)
Dedicated logic registers	11 / 33,216 (< 1 %)
Total registers	11
Total pins	3 / 475 (< 1 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.9 Joc Extra

Joc extra te la mateixa estructura que joc, pero se li afegeix la sortida tot que ens donara el nombre a mostrar en el cas que es facin trampes. La sortida num no s'utilitza, nomes la hem fet servir per a debuguejar el programa.

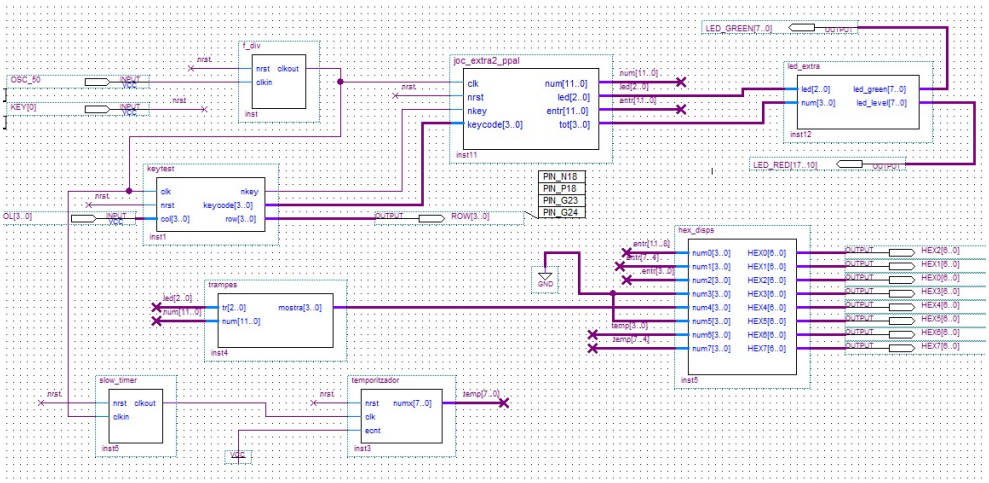

clk
nrst
nkey
keycode[3..0]
keygroup_extra
inst
control_extra
inst3
comptadorBCD_extra
inst6
comparador_extra
inst11
regs_extra
inst7
OUTPUT
num[11..0]
led[2..0]
entr[11..0]
tot[3..0]


Name
clk
nrst
nkey
keycode
entr
led
num
tot
23.475 ns

Flow Status	Successful - Thu Dec 17 19:22:34 2020
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	practica3
Top-level Entity Name	joc_extra2
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	291 / 33,216 (< 1 %)
Total combinational functions	291 / 33,216 (< 1 %)
Dedicated logic registers	86 / 33,216 (< 1 %)
Total registers	86
Total pins	82 / 475 (17 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

2.10 Joc Extra Placa

Per a acabar tenim el joc posat a la placa. Per a fer aixo hem reusat el fet per la part 1 i hi hem afegit tres moduls: slow_timer, temporitzador i trampes. També hem actualitzat els blocs del programa principal i el de leds. El bloc trampes ens donara la xifra a mostrar per pantalla en el cas que control dins de ppal estigui en algun dels estats trampaA o trampaB. El bloc temporitzador rebra el clk modificat de slow_timer i anira descomptant xifres segon per segon. Finalment leds_extra ens donara, a mes de si el nombre es mes gran, igual o mes petit, el nombre de partides que portem guanyades.


f_div
nrst
clkout
clkin
inst
OSC_50
INPUT
KEY[0]
nrst2
joc_extra2_ppal
clk
nrst
nkey
keycode[3..0]
num[11..0]
led[2..0]
entr[11..0]
tot[3..0]
inst11
num[11..0]
led[2..0]
entr[11..0]
led_extra
led[2..0]
num[3..0]
led_green[7..0]
led_level[7..0]
inst12
LED_GREEN[7..0]
OUTPUT
LED_RED[17..10]
keytest
clk
nrst
col[3..0]
nkey
keycode[3..0]
row[3..0]
inst1
OL[3..0]
BIDIR
OUTPUT
ROW[3..0]
PIN_N18
PIN_P18
PIN_G23
PIN_G24
GND
hex_disps
num0[3..0]
num1[3..0]
num2[3..0]
num3[3..0]
num4[3..0]
num5[3..0]
num6[3..0]
num7[3..0]
HEX0[6..0]
HEX1[6..0]
HEX2[6..0]
HEX3[6..0]
HEX4[6..0]
HEX5[6..0]
HEX6[6..0]
HEX7[6..0]
inst5
entr[11..8]
entr[7..4]
entr[3..0]
temp[3..0]
temp[7..4]
trampes
led[2..0]
num[11..0]
tr[2..0]
num[11..0]
mostra[3..0]
inst14
slow_timer
nrst
clkout
clkin
inst6
temporitzador
nrst
clk
eont
numx[7..0]
temp[7..0]
inst3
VCC