DISSENY DEL JOC DE DAUS "CRAPS"

Christian Peter Winter, Francisco Muñoz Contreras

Segon curs d'Enginyeria Tècnica Industrial, Especialitat en Electrònica Industrial
christian@kpwinter.com
feco_m@msn.com

Resum— **"Craps" està present a tots els grans casinos del món i es un dels jocs de daus amb més èxit. Llavors, la proposta es dissenyar e implementar sobre una placa programable una versió digital d'aquest joc. Les eines necessàries per dur a terme aquesta tasca seran el programa de disseny MAX+plus II i la placa d'entrenament UP-1, tots dos productes de l'empresa Altera (una de les empreses líder en tecnologies de lògica programable).**

V. INTRODUCCIÓ AL JOC

Inicialment, el jugador fa l'aposta que cregui oportuna i llança els dos daus. L'objectiu es treure a la primera tirada (també anomenada tirada de sortida) un resultat total igual a 7 ó a 11 de manera que el jugador guanya dues vegades l'aposta feta. Es perd si es treu un "crap", es a dir, un 2, 3 ó 12.

Si el tirador obté un 4, 5, 6, 8, 9 ó 10 es diu que ha tret el seu "punt". A partir d'aquest moment, l'objectiu serà tornar a treure el mateix "punt" per recuperar l'aposta (per exemple si s'ha tret un 4, s'ha d'intentar tornar a treure un 4). Si es treu un 7 abans de treure el "punt", es perd (també es diu "crap").

Aquestes són les regles més bàsiques del joc. Per obtenir una informació més completa:

Link: http://es.wikipedia.org/wiki/Craps

VI. INTRODUCCIÓ A LA PLACA

La versió electrònica del joc consistirà en el maneig d'una placa programable (ex. Fig.1). El sistema constarà de tres polsadors: Dos per emular el llançament dels daus y un altre (de reset) per inicialitzar la partida. A més, els polsadors dels daus es podran pitjar de manera independent.

Fig. 1 Exemple d'una placa d'entrenament UP-1.

Els resultats es mostraran a dos displays de set segments (ex. Fig.2). La informació visualitzada en aquests displays seguirà una pauta predefinida de la següent forma:

Primerament es mostrarà "In" fins que es polsin els polsadors dels daus, moment en el qual s'observarà als dos displays una fluctuació aleatòria de valors entre 1 i 6 (les cares d'un dau) que al final s'estabilitza a un valor concret per a cada dau. Finalment, passats cinc segons mostrant els valors, es visualitzarà "Gn" si el jugador ha guanyat, "Cp" si ha perdut o "Pn" si ha tret un punt (en aquest cas s'haurà de tornar a polsar). Per reinicialitzar, bastarà amb polsar el botó de reset i el sistema tornarà al seu estat inicial.

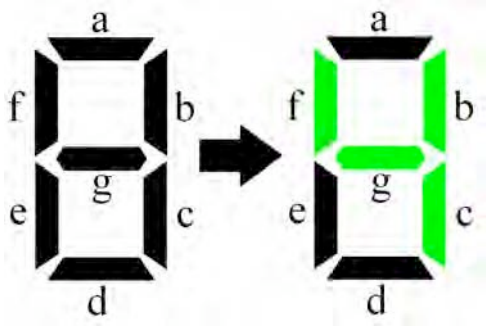

Fig.2 Exemple d'un display de set segments. S'observa que el display té distints segments (a, b, c, d, e, f i g) que, controlats individualment, donen lloc a multitud de combinacions (entre altres els números del 0 al 9).

VII. FUNCIONAMENT INTERN DEL DISSENY

C. Màquina d'estats

Tot disseny seqüencial es pot definir a partir d'una màquina d'estats. Aquesta màquina té per funció representar el funcionament intern del sistema a nivell d'estats y transicions assignant a cada estat les sortides corresponents y a cada transició les condicions específiques.

Exemple senzill: Tenim un sistema format per una bombeta y un interruptor. Per tant, tenim 2 estats: "bombeta apagada" y "bombeta encesa". Si estam a l'estat "bombeta apagada", la sortida "donar tensió a la bombeta" està a 0 (no donam tensió). Si ara polsem el interruptor (que és la condició), passarem al estat "bombeta encesa" estant la sortida a 1 (donam tensió).

La vertadera utilitat d'una màquina d'estats s'observa en sistemes més complexos en els quals ens topam amb desenes d'estats, sortides i condicions.

En el nostre cas, la màquina (Fig.3) ja es relativament complexa i encara que pot resultar difícil entendre-la completament, segueix essent la millor manera per representar de forma visual com funciona el disseny.

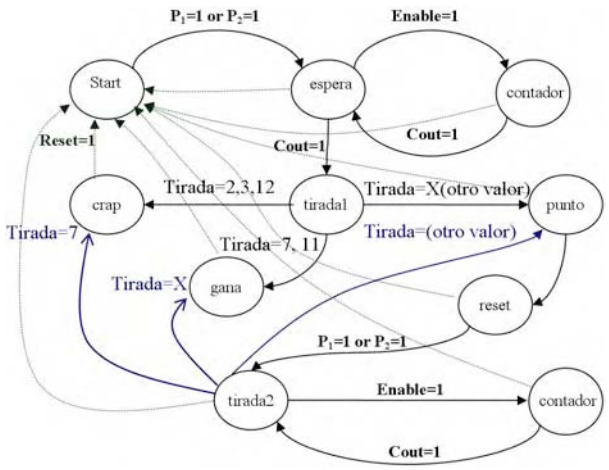

Fig. 3 Màquina d'estats que controla el funcionament del sistema. Podem observar que està formada per 10 estats (que són els 10 blocs que integren el disseny) i que les transicions depenen de diferents condicions a l'hora.

D. Disseny amb el programa

Una vegada ideada la màquina d'estats que defineix el funcionament del sistema, passam al disseny mitjançant un programa especial. Aquest programa (anomenat MAXplus II) es una eina molt potent i té, entre altres moltes funcions, un editor de codi (Fig.5), un editor gràfic (Fig.4), un compilador i un simulador. Per tant, es possible editar diferents blocs lògics a nivell de codi y llavors interconnectar-los amb l'editor gràfic. Finalment es possible compilar y simular el sistema conjunt y així verificar el funcionament correcte del disseny.

Els distints blocs creats (amb l'editor de codi) per implementar el joc de "Craps" son els següents:

- Comptadors de 1 a 6 (Fig.4, Fig.5) .
- Temporitzadors.
- Habilitadors.
- Multiplexors.
- Divisors de freqüència.
- Sumador.
- Controlador.
- Conversors BCD integrats.
- Pautes pseudo-aleatòries implementades.
- Màquina d'estats principal implementada.

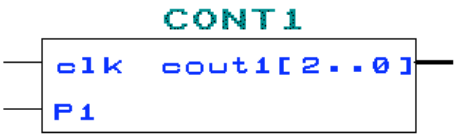

Fig. 4 Comptador editat textualment (Fig.5) e implementat a l'editor gràfic en forma de bloc.

E. Explicació breu de la funció de cada bloc

Per entendre correctament el funcionament del sistema complet, és necessari entendre la funció que té cada bloc de l'integrat.

Com que hi ha un total de 10 blocs diferents, una explicació exhaustiva de cada bloc seria excessiu. Per aquest motiu, s'ha optat per una explicació qualitativa.

1_Contador de 1 a 6:

En el període de temps que el polsador es troba polsat, el comptador està habilitat i compta a una freqüència de 25,175MHz (aquest es el seu rellotge intern). Tenint en compte que la freqüència es molt elevada, podem suposar que el valor que finalment es copia a la sortida (quan deixam de polsar) es completament aleatori.

2_Temporitzador:

La funció del temporitzador es comptar el temps que s'ha polsat el polsador i, quan s'ha amollat aquest, posar a la sortida un valor proporcional al temps comptat (temps multiplicat per 5). El valor màxim que es posarà a la sortida serà de 10 segons (si s'ha polsat el polsador 2 o més segons).

3_Habilitador:

Té per missió habilitar altres blocs depenent de si una variable interna es igual o no a una de les seves entrades. Principalment serveix per organitzar la tasca a executar.

4_Multiplexor:

Segons el valor que tingui a la entrada selectora, selecciona una de les seves entrades i la copia a la sortida. D'aquesta manera es selecciona la sortida del comptador o la pauta pseudo-aleatòria segons convingui.

```
--contador 6
ENTITY cont1 IS
    PORT(clk,P1: IN BIT;
         cout1:OUT INTEGER RANGE 1 TO 6);
END cont1;

ARCHITECTURE arch1 OF cont1 IS
BEGIN
    PROCESS (clk)
    VARIABLE n: INTEGER RANGE 1 TO 6;
        BEGIN
            IF (clk'EVENT AND clk='1') THEN
                IF P1='1' AND n<6 THEN
                    n:=n+1;
                ELSIF P1='1' AND n=6 THEN
                    n:=1;
                ELSE
                    cout1<= n;
                END IF;
            END IF;
        END PROCESS;
END arch1;
```

Inicialitzacions

Comptador 1-6 dependent de E1

Fig.5 Exemple de Codi. Text escrit amb l'editor de codi que implementa un comptador de 1 a 6 que s'habilita amb una entrada E1 i s'incrementa amb cada període de rellotge.

5_Divisor de freqüència:

Com ja ho diu el nom, divideix per 10 la freqüència que li entra. Com que la freqüència d'entrada es molt elevada (25,175MHz), resulta necessari dividir-la per poder dur a terme certes tasques.

6_Sumador:

Suma les seves entrades. Serveix per sumar els valors obtinguts en el llançament dels daus (emulat per el polsadors).

7_Controlador:
Necessari per saber si ja s'han polsat els polsadors. Segons la combinació (ningú polsat, un polsat, tots dos polsats) activa un senyal o un altre.

8_Conversor BCD integrat:
S'ocupa de controlar els displays de set segments de manera que es visualitza la informació correcta.

9_Pauta pseudo-aleatòria:
Abans d'estabilitzar-se el valor, aquest estarà fluctuant de manera aleatòria entre valors de 1 i 6. Realment, aquesta fluctuació no es aleatòria (està controlada per una màquina d'estats interna).

10_Màquina d'estats principal:
Controla tota la tasca de inici a fi. Segons les senyals que activa, el sistema fa una cosa o l'altra (entre altres, decideix si s'ha guanyat, perdut o tret un punt).

El procés que duu a terme el circuit es, en general, seqüencial però també hi ha tasques que es duen a terme de manera paral·lela. A nivell d'exemple:

L'entrada P1 (polsador 1) habilita el temporitzador, aquest l'habilitador que, en conseqüència, habilita la pauta aleatòria. Aquesta passa per el multiplexor y quan acaba, habilita el sumador y la màquina d'estats principal. Finalment, quan aquests han processat la informació, habiliten el conversor BCD obtenint així el resultat final. Es veu que és un procés totalment seqüencial. De manera paral·lela, es duu a terme el mateix procés per a l'entrada P2.

Aquest exemple fa clara la complexitat del disseny. No és d'estranyar que un sistema tan complex presenti certs errors. Per aquest motiu, el programa inclou dues funcions merament importants: el compilador i el simulador. Aquestes funcions comproven tot el codi i, en cas d'errors, informen al programador. Si no hi ha cap tipus d'error, ja es pot passar a l'últim pas que és la implementació sobre una placa.

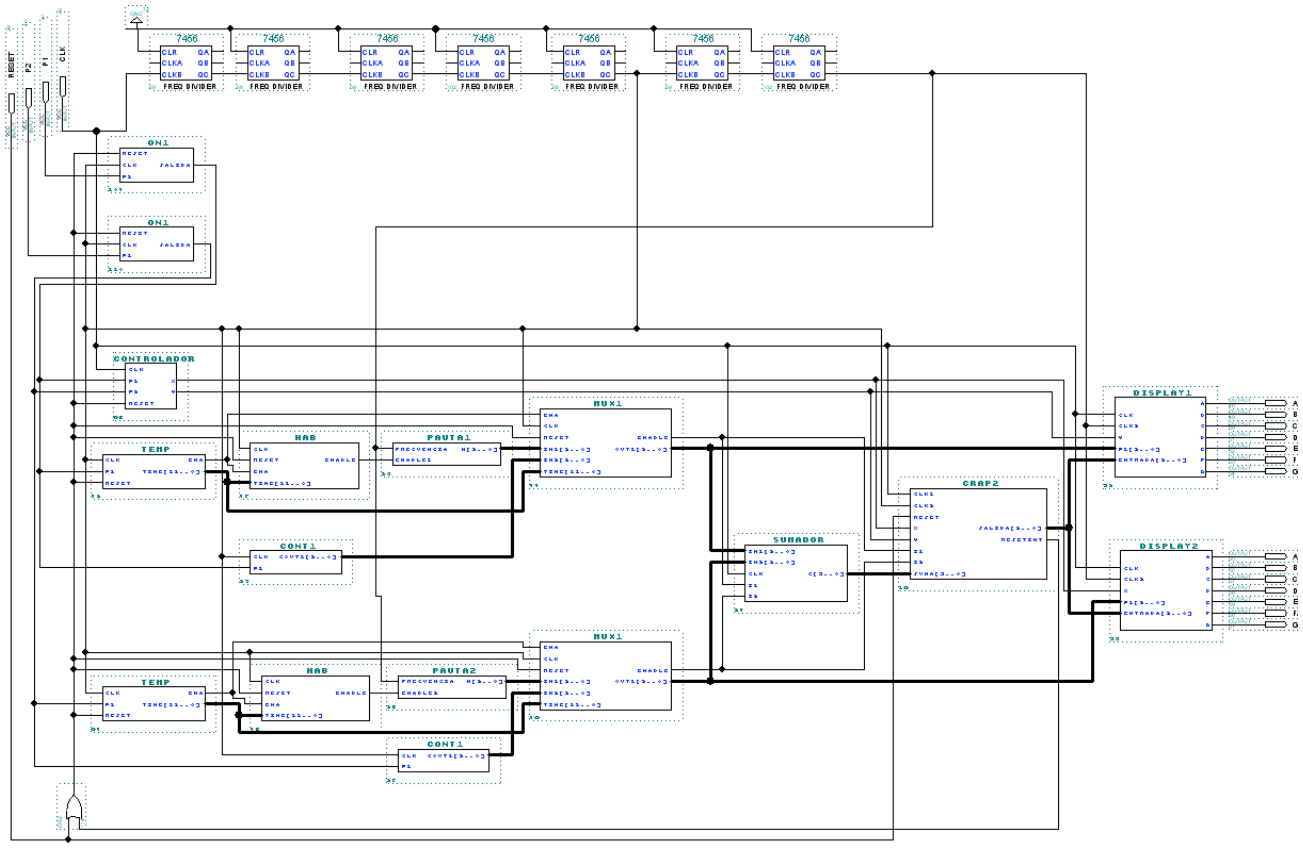

Fig. 6 Esquema del sistema complet (dibuixat amb l'editor gràfic)

F. Disseny final

Tenint tots els blocs dissenyats, es pot passar a la interconnexió d'aquests. Mitjançant l'editor gràfic podem "dibuixar" els blocs i , llavors, connectar-los entre ells amb diferents tipus de línies (d'un bit o de múltiples bits). Fet això, el disseny estarà complet (Fig. 6) y llest per ser simulat i implementat en la placa.

VIII. IMPLEMENTACIÓ FÍSICA

La implementació física es l'últim pas del disseny. En primer lloc es connectarà una "byte-blaster" (Fig. 7) com a medi de comunicació entre PC i placa. La resta és una configuració que es duu a terme pas a pas y ve explicada en el datashett (full de característiques) de la placa en qüestió.

Resumint, els passos a seguir son aquests:

- Assignar la placa →→→ EPF10K20RC240-4

- Compilar
- Assignació de pins
- Compilar (creació d'un fitxer .sof)
- Programar placa (amb la funció "programmer")

Seguits aquest passos, la placa ja es trobarà programada i llesta per a ser utilitzada, facin les seves apostes!

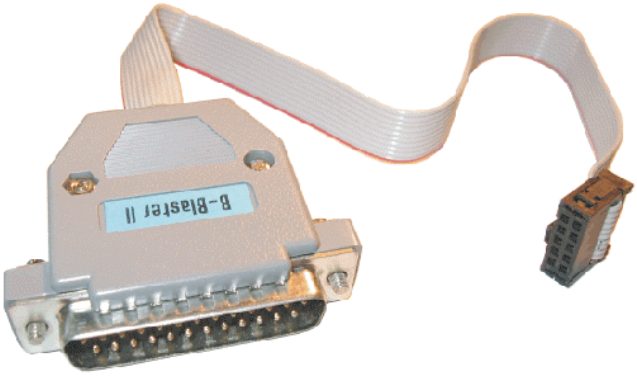

Fig. 7 Fotografia d'una "Byte-blaster". Aquest cable es connectarà entre el port paral·lel del PC i la placa programable (es la connexió física entre els dos components).

IX. CONCLUSIONS

Encara que el joc de Craps real pugui ser bastant més entretingut que no la seva versió digitalitzada, amb aquest disseny queda clar que, encara que una tasca sigui complexa, amb les eines adequades es possible implementar-la sense grans dificultats.

AGRAÏMENTS

Volem donar les gràcies a la Universitat de les Illes Balears per proporcionar-nos les eines de disseny, i la placa d'entrenament i al professor de l'assignatura "Sistemes electrònics digitals", Josep Lluís Rosselló Sanz que ens va ajudar en moments de dubte.