Práctica 5

- Sistema para jugar al *Black Jack*

Práctica 5

- Desarrollar en VHDL un sistema para jugar al popular juego de cartas *Black Jack*.
  - i) El jugador comienza la partida (inicio ='1')
  - ii) El jugador pide cartas
    - Debe intentar acercarse a una puntuación de 21. Si se pasa, pierde la partida.
  - iii) El jugador puede decidir plantarse en cualquier momento. Pero si pide carta y su nueva puntuación es mayor que 21, pierde la partida.

Práctica 5

- Cómo se implementa el acceso aleatorio a la baraja:
  - Existe un contador que siempre está contando de 0 a 51
    - El contador tiene solo una señal de reset, y está contando siempre que reset='0'
  - El jugador, cuando pide una carta, lee el contenido del contador. El valor devuelto por el contador indica la posición de memoria donde se leerá la carta que el sistema da al jugador.
  - Una vez leída la carta esta se borra del taco
    - Se escribe cero en su posición
  - Cada carta numérica tiene su valor, y las figuras tienen valor 10.
  - El jugador tiene 1 baraja a su entera disposición, con 52 cartas (para cada palo, cartas 1-10 y 3 figuras que valen 10 puntos).

Práctica 5

- Simplificaciones que haremos en este sistema:
  - El jugador juega contra sí mismo, no hay banca.
  - Los ases no tienen doble valor (1 u 11); sólo valen 1.
  - No se pueden “separar cartas” cuando el jugador tenga 2 cartas iguales.

Práctica 5

- Gestión de la **memoria de cartas**:
  - Las cartas que quedan en la baraja del jugador se guardarán en una memoria.
  - Esta memoria tendrá 52 posiciones que deben inicializarse a los valores de estas cartas.
    - MEM(0) = 1; MEM(1) = 2; …, MEM(12) = 10;
    - MEM(13) = 1; ….
  - Cuando el sistema dé una carta al jugador, ésta se debe eliminar de la memoria.
    - El contenido de la posición de memoria seleccionada se actualizará a (others => '0').
  - Si cuando se selecciona una nueva carta el contador devuelve una dirección de memoria en la que hay guardado un 0 (carta asignada en una tirada anterior), el juego debe ***automáticamente*** pedir una nueva carta.

Práctica 5: Entity

- Entradas:
  - Reset.
  - Reloj.
  - Inicio (para comenzar el juego).
  - Jugar (para pedir carta).
  - Plantarse.
- Salidas:
  - Valor de la carta
  - Pierdo
  - Puntuación (acumulada por el jugador, 6 bits)

Práctica 5: Implementación en FPGA

- Entradas:
  - Reset (switch)
  - Reloj
  - Inicio (switch)
  - Jugar (pulsador)
  - Plantarse (pulsador)
- Salidas:
  - Valor de la carta (1 display 7-segmentos)
  - Pierdo (1 LED)
  - Puntuación (6 LEDs)

Los pulsadores son activos a baja: cuando se pulsan generan un '0'

Especificaciones

- Las cartas estarán almacenadas a partir de la posición 0x00 de una memoria SRAM **síncrona**.
- La memoria será de un solo puerto:
  - Se puede realizar de forma independiente una operación de lectura o una operación de escritura
- La definición de puertos es:
  - din, bus de datos de entrada
  - addr, bus de direcciones
  - we, write enable
  - dout, bus de datos de salida. El puerto de salida conserva el valor leído hasta que se realice una nueva lectura.

Memoria en VHDL

- Usad el wizard incluido en ISE 14.1 para generar memorias de un solo puerto.