

DISEÑO DE UNA MÁQUINA EXPENDEDORA

**PRINCIPIOS DE DISEÑO DE
SISTEMAS DIGITALES 23-24**

MEMORIA DEL PROYECTO

**IKER FERNÁNDEZ & DANIEL TALMACI
GRUPO 01_04**

ÍNDICE

DESCRIPCIÓN DEL PROYECTO.....	3
MEMORIA	
DÍA 1, 2 y 3	4
DÍA 3 Y 4	5
DÍA 5 Y 6	6
DÍA 7 Y 8	7
PROYECTO FINAL	
TEMPORIZADOR	8
VHDL COMPONENTES TEMP Y UNIDAD DE CONTROL ...	9
SUMADOR	
ESQUEMA Y VALOR MONEDA	10
VHDL COMPONENTES	11
BLOQUE COMPARADOR	12
MÁQUINA EXPENDEDORA - PROYECTO DEFINITIVO ...	13
CONCLUSIÓN.....	14

PROYECTO

En esta tarea se nos pide, por parejas, realizar el diseño e implementación de una máquina expendedora en la que se podrán introducir monedas de 5, 10, 25 céntimos, la máquina no devolverá cambios. La máquina dispone de los siguientes productos, que son; la insignia y el gorro de lana con pom-pom.

El precio de los productos es el siguiente:

- Insignia: 15 céntimos
- Gorrito de lana con pom-pom: 25 céntimos

Si la suma de las monedas introducidas es igual o supera los 15 céntimos y es menor que 25 céntimos saldrá la insignia, si la suma de las monedas introducidas es igual o mayor a 25 saldrá el gorrito de lana.

La máquina expendedora se quedará las monedas introducidas al pasar 30 segundos, si con el valor introducido no se ha podido conseguir ni uno de los productos disponibles, de esta manera perdiendo el dinero introducido.

El proyecto será desarrollado y programado mediante el programa de ISE, utilizando una máquina virtual para acceder a este y poder realizar el trabajo.

MEMORIA

Día 1 : 12/09/23

- **Plan de Trabajo:** Este día comenzaremos con el programa necesario para trabajar con nuestra placa Nexys. Haremos la simulación de puertas lógicas utilizando los interruptores.
- **Trabajo Logrado:** Hemos aprendido a crear un nuevo proyecto en ISE, editar el esquema y los pasos necesarios a seguir para implementar el diseño creado en nuestra tarjeta Nexys. Al finalizar hemos observado el funcionamiento de nuestro sistema utilizando los interruptores y los LED.
- **Problemas y dificultades:** Hoy no hemos tenido ningún inconveniente.

Día 2 : 27/09/23

- **Plan de Trabajo:** Haremos el diseño del sumador que llevará implementado nuestra máquina expendedora.
- **Trabajo Logrado:** Hemos conseguido implementar en nuestra tarjeta el sumador o restador en este caso dependiendo de lo indicado, en este caso con 5 bits para realizar estas operaciones en complemento a dos, utilizando el bit más significativo como bit de signo.
- **Problemas y dificultades:** Este día no hemos encontrado dificultades en el avance del proyecto.

Día 3 : 10/10/23

- **Plan de Trabajo:** Hoy crearemos el acumulador, uniendo el sumador y el registro, de esta manera logrando que nuestra máquina recuerde la cantidad de monedas que tiene dentro, sumando las monedas introducidas a las que ya están dentro.

- **Trabajo Logrado:** Hemos obtenido mantener el dato introducido utilizando el registro y combinándolo con el sumador a 5 bits para seguir sumando valores al guardado anteriormente (los 5 bits solo son necesarios para esta práctica, para las siguientes utilizaremos uno de 4 bits). También, mediante la señal Reloj, hemos hecho que al activar a 0 esta señal el acumulador vuelva a 0, de esta manera haciendo un “Reset” del sistema.
- **Problemas y dificultades:** Se ha hecho un poco laborioso hacer el cambio del sumador a 5 bits ya que el dado estaba a 4, pero lo hemos solucionado fácilmente.

Día 4 : 25/10/23

- **Plan de Trabajo:** Hoy comenzaremos a combinar el acumulador con la selección de moneda, para saber el valor introducido en la máquina y finalmente utilizar el comparador para saber si con la cantidad de monedas es posible conseguir alguno de los artículos disponibles de la máquina expendedora.
- **Trabajo Logrado:** Hemos conseguido realizar la selección de moneda donde solo podemos introducir 3 valores que son las monedas posibles (5, 10, 20 cents.), estas solo podrán ser introducidas de una en una. Después hemos comenzado con el comparador el que viendo el valor del acumulador dirá si es posible obtener uno de los dos productos, asegurándonos que solo se pueda obtener un artículo a la vez.
- **Problemas y dificultades:** Hemos tenido dificultades a la hora de sintetizar e implementar nuestro diseño lo que ha hecho ralentizar nuestro procedimiento de trabajo y influido en el no haber podido acabar la Práctica 4.

Día 5 : 08/11/23

- **Plan de Trabajo:** Acabaremos con la Práctica 4 haciendo uso de un divisor de frecuencia.
- **Trabajo Logrado:** Finalmente hemos aplicado el divisor de frecuencia y asegurado del funcionamiento de nuestra máquina, haciendo que para introducir una moneda haya que esperar un tiempo de 0,5 segundos.
- **Problemas y dificultades:** En el día de hoy no se han presentado dificultades.

Día 6 : 22/11/23

- **Plan de Trabajo:** Hoy haremos la unidad de control que llevará implementada nuestra máquina.
- **Trabajo Logrado:** Hemos logrado unir nuestra unidad de proceso con la unidad de control de esta manera asegurándonos del buen funcionamiento de nuestra máquina. Con esto hemos conseguido asegurarnos que la máquina empieza con un valor de 0 monedas introducidas, que solo cambie el registro al introducir nuevas monedas y que aparezca el producto que es posible comprar con el valor de las monedas introducidas, siempre teniendo en cuenta las condiciones iniciales pedidas. Además, nos ha dado tiempo a crear el esquema del temporizador.
- **Problemas y dificultades:** Hoy no hemos tenido problemas para afrontar la práctica.

Día 7 : 29/11/23

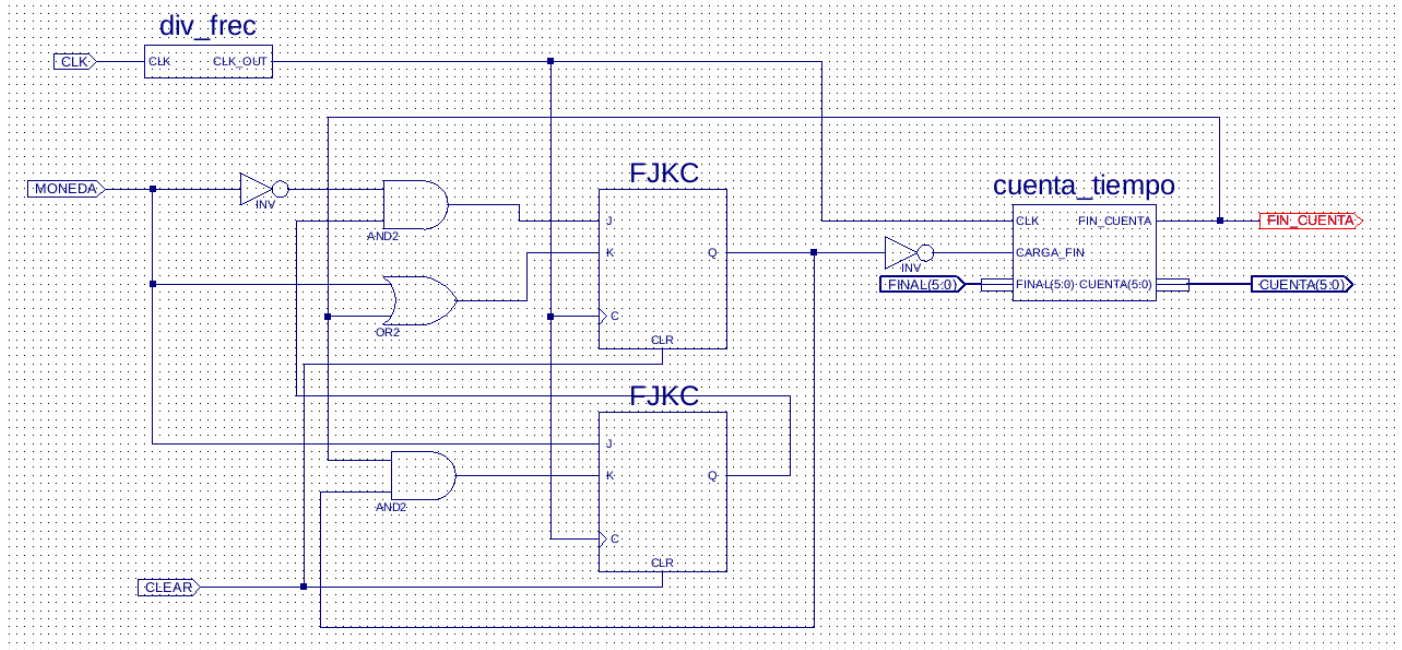
- **Plan de Trabajo:** Antes de finalizar con el proyecto implementaremos el temporizador a nuestro trabajo ya obtenido anteriormente durante estos días.
- **Trabajo Logrado:** Hemos logrado implementar con facilidad el diseño del temporizador a nuestra máquina expendedora.
- **Problemas y dificultades:** No se han presentado dificultades en el día de hoy.

Día 8 : 01/12/23

- **Plan de Trabajo:** En el último día tenemos intención de una vez obtenido el funcionamiento casi completo de la máquina, conseguir que cuando la máquina no detecta ninguna moneda introducida durante 30 segundos devuelva las monedas si no son suficientes para obtener ningún producto.
- **Trabajo Logrado:** Finalmente hemos conseguido dar por finalizado el proyecto consiguiendo la máquina expendedora con las funcionalidades pedidas desde un principio, habiendo implementado el último cambio para la devolución de monedas después de 30 segundos sin haber ni un cambio.
- **Problemas y dificultades:** Ha sido un poco problemático conseguir que el tiempo sea 30 segundos, porque desconocíamos que podía utilizarse el sistema hexadecimal en las constantes.

DISEÑO FINAL

TEMPORIZADOR



El temporizador se activa cuando la entrada **MONEDA** es 1, es decir, cuando entra cualquier tipo de moneda.

El **DIVISOR DE FRECUENCIA**, hace que cada período del **CLOCK** valga 0,5 segundos.

Cuando se cumple que **MONEDA** es 1, empieza a sumar hasta 30 (Que se ha especificado en la entrada **FINAL**) puesto que queremos que cuando pasen 30 segundos **FIN CUENTA** se active.

Si la entrada **CLEAR** se pone a 1, la cuenta se reiniciará y parará.


```

entity div_frec is
  Port ( CLK : in  STD_LOGIC;
        CLK_OUT : out  STD_LOGIC);
end div_frec;

architecture Behavioral of div_frec is
  constant FINAL: integer:= 10#25000000#;
  signal CUENTA: std_logic_vector(24 downto 0);
  signal Z: std_logic;

begin
  CLK_OUT  <= Z;

  CONTAR: process (CLK)
  begin
    if (CLK'EVENT AND CLK = '1') THEN
      CUENTA  <= CUENTA + 1;
      if (CUENTA = FINAL) then
        CUENTA  <= (OTHERS => '0');
        Z      <= NOT Z;
      end if;
    end if;
  end process;
end div_frec;

```

VHDL del DIVISOR DE FRECUENCIA

```

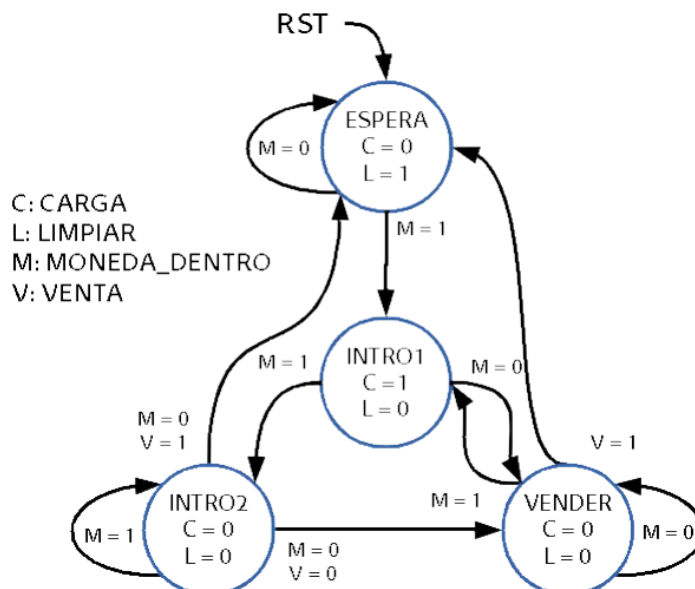
entity cuenta_tiempo is
  Port ( CLK, CARGA_FIN : in  STD_LOGIC;
        FINAL : in  STD_LOGIC_VECTOR (5 downto 0);
        FIN_CUENTA : out  STD_LOGIC;
        CUENTA : out  STD_LOGIC_VECTOR (5 downto 0)
end cuenta_tiempo;

architecture Behavioral of cuenta_tiempo is
  signal TIEMPO: STD_LOGIC_VECTOR (5 downto 0);
begin
  FIN_CUENTA<= '1' when TIEMPO = 0 else '0';
  CUENTA <= TIEMPO;
  CONTADOR: process (CLK, FINAL, CARGA_FIN)
  begin
    if (CARGA_FIN = '1') then TIEMPO <= FINAL;
    elsif(CLK'EVENT AND CLK = '1') then
      TIEMPO<= TIEMPO - 1;
      if TIEMPO = 0 then
        TIEMPO <= FINAL;
      end if;
    end if;
  end process;
end cuenta_tiempo;

```

VHDL del CUENTA TIEMPO

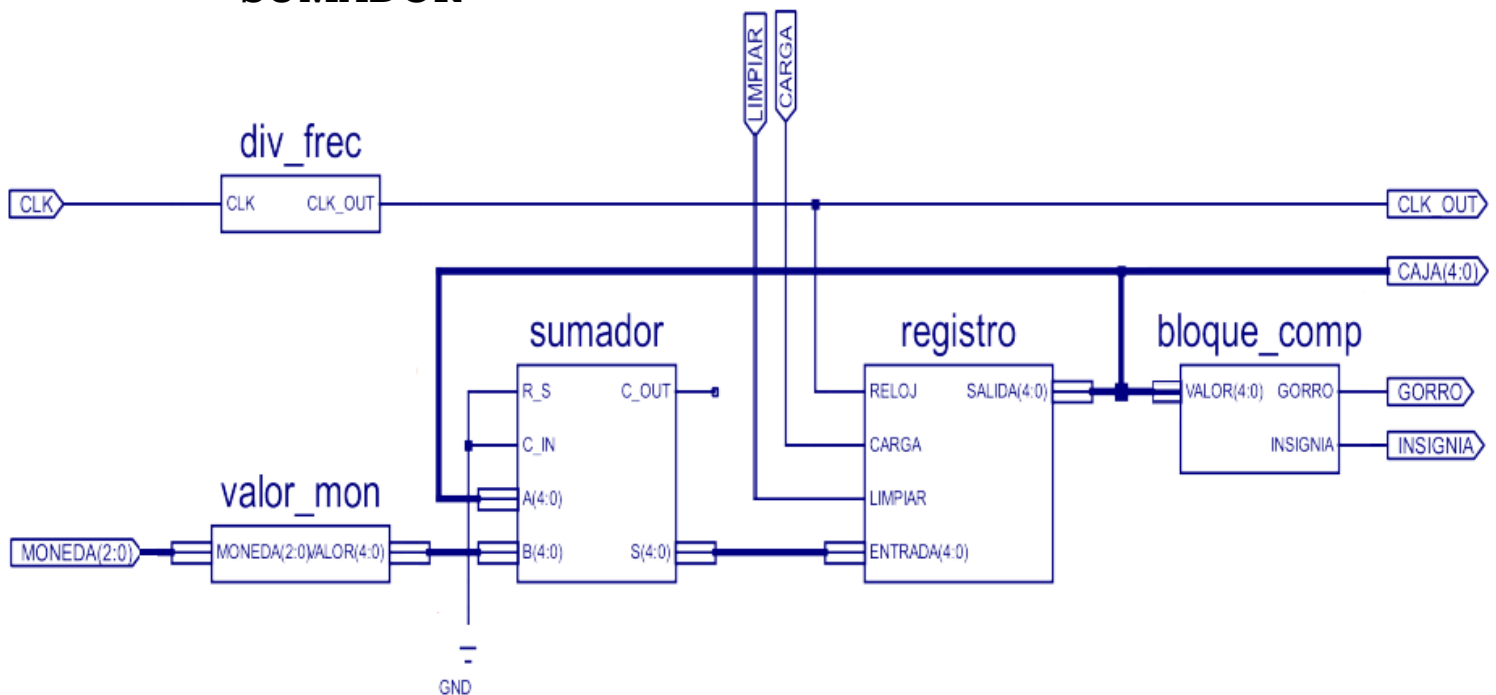
UNIDAD DE CONTROL



La unidad de control detecta si ha habido alguna **VENTA**, para así **LIMPIAR** la **CAJA**. Para introducir un valor en el **REGISTRO**, **CARGA** debe mantenerse durante 0,5 segundos.

Diagrama de estados de la UNIDAD DE CONTROL

SUMADOR



El **VALOR MONEDA**, da un valor de 5, 10 o 20 a la moneda insertada.

```

entity valor_mon is
  Port ( MONEDA : in  STD_LOGIC_VECTOR (2 downto 0);
        VALOR : out  STD_LOGIC_VECTOR (4 downto 0));
end valor_mon;
architecture Behavioral of valor_mon is
begin
  with MONEDA select
    VALOR<=
      "10100"    when "100",
      "01010"    when "010",
      "00101"    when "001",
      "00000"    when others;
end Behavioral;
  
```

VHDL del VALOR MONEDA

El **SUMADOR RESTADOR**, suma y resta las monedas insertadas. La suma se almacena en el **REGISTRO**.

```

entity SUM_REST is
  Port ( A : in  STD_LOGIC_VECTOR (3 downto 0);
        B : in  STD_LOGIC_VECTOR (3 downto 0);
        C_IN : in  STD_LOGIC;
        R_S : in  STD_LOGIC;
        S : out  STD_LOGIC_VECTOR (3 downto 0);
        C_OUT : out  STD_LOGIC);
end SUM_REST;

architecture Behavioral of SUM_REST is
  signal A_IN, B_IN, S_OUT: STD_LOGIC_VECTOR (4 DOWNTO 0);
begin
  A_IN<='0'&A;
  B_IN<= '0'&B;
  S <= S_OUT(3 downto 0);
  C_OUT<= S_OUT(4);

  SUMA: process (R_S, A_IN, B_IN, C_IN)
  begin
    case R_S is
      when '0'      =>      S_OUT<= A_IN + B_IN + C_IN;
      when others => S_OUT<= A_IN - B_IN + C_IN;
    end case;
  end process;
end Behavioral;

```

VHDL del SUMADOR RESTADOR

```

entity REGISTRO is
port(
  reloj, carga, limpiar:      in std_logic;
  entrada:                    in std_logic_vector(4 downto 0);
  salida:                      out std_logic_vector(4 downto 0)
);
end REGISTRO;

architecture arch of REGISTRO is

  signal dato: std_logic_vector (4 downto 0);

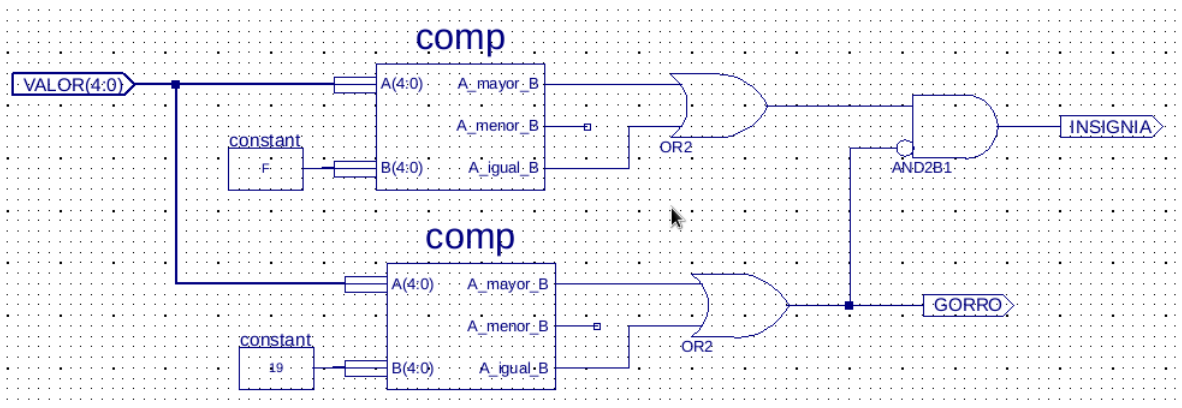
begin

  salida <= dato;

  process(reloj,limpiar)
  begin
    if (limpiar='1') then
      dato <=(others=>'0');
    elsif (reloj'event and reloj='1') then
      if carga = '1' then
        dato <= entrada;
      end if;
    end if;
  end process;
end arch;

```

VHDL del REGISTRO

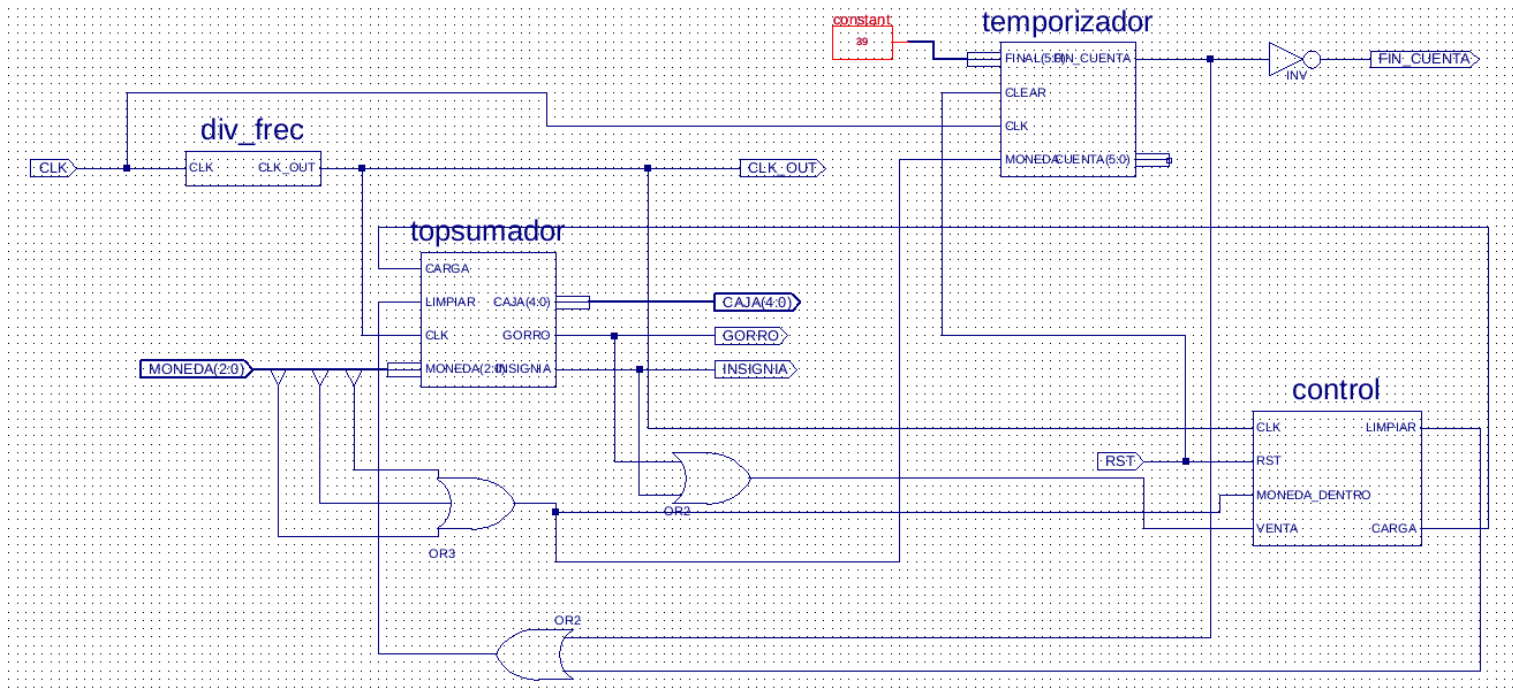


Esquema del BLOQUE COMPARADOR

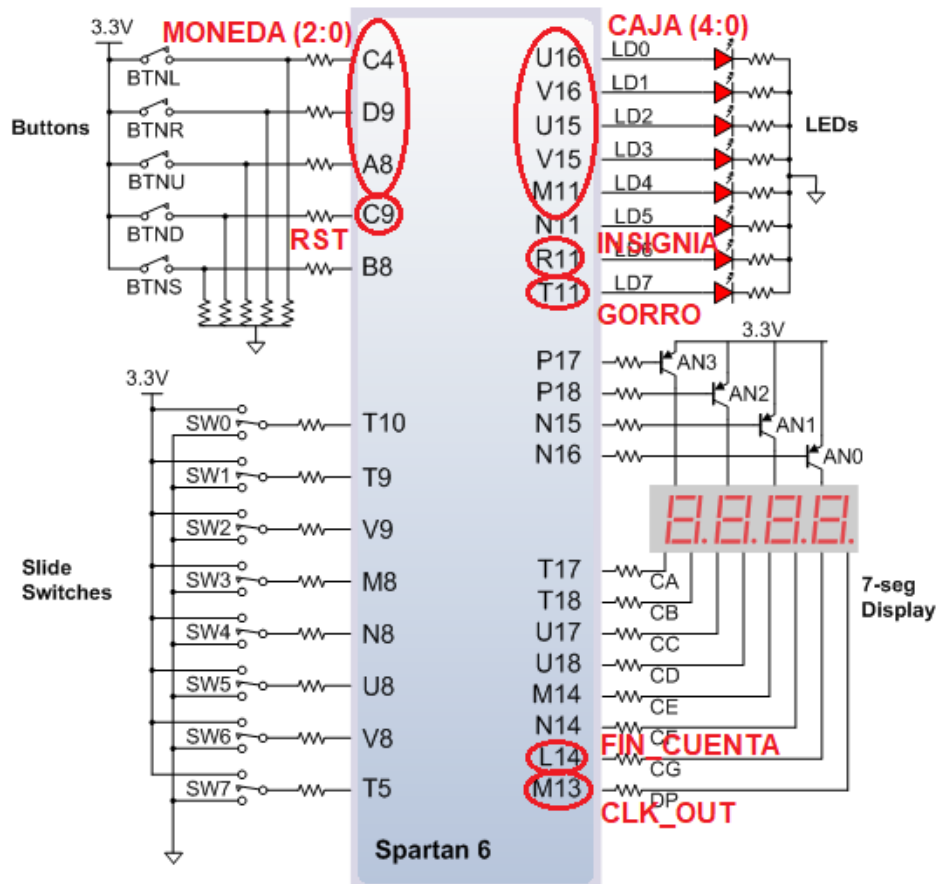
El **BLOQUE COMPARADOR** identifica si se ha igualado o superado el precio de la insignia o el gorro.

Una vez conectados todos estos componentes, obtenemos el **SUMADOR**. Tiene como salida la **CAJA**, donde se señala el número de dinero que ya se ha metido. También tiene como salidas, **INSIGNIA Y GORRO**, que son los productos a vender.

MÁQUINA EXPENDEDORA - PROYECTO DEFINITIVO



ASIGNACIÓN DE PINES



CONCLUSIÓN

Una vez logrado el sistema digital deseado, hemos conseguido el objetivo de nuestro proyecto, la máquina expendedora, cuya especificación está redactada al principio del informe.

Mediante este proyecto hemos implementado muchos de los conceptos adquiridos en clase, de esta manera llevando lo teórico a un terreno práctico, donde todos estos aprendizajes se ven útiles en problemas de la vida cotidiana.

A parte de trabajar conceptos aprendidos a lo largo de la asignatura, a su vez, se logra mejorar el trabajo en equipo, en este caso por parejas, ayudándonos mutuamente a avanzar en el proyecto y sustentando el aprendizaje en equipo, de esta manera afrontando las diferentes adversidades de manera más amena.

Por lo general, estamos contentos del trabajo realizado y como hemos logrado llegar al final de este proyecto. Además de haber podido aplicar los conceptos adquiridos de una manera práctica.