UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DEL CONTROL ELECTRÓNICO DIGITAL DEL MOTOR ELÉCTRICO EN UNA CINTA DE CORRER

AUTOR: CAMPOS RODRÍGUEZ, LUCAS MIGUEL

TUTOR: ESTEVE BOSCH, RAÚL

COTUTORA: ROMERO PÉREZ, LUCÍA

Curso Académico: 2013-14

Índice General

Memoria

Memoria

1 Objetivo del Trabajo Fin de Grado

Durante el presente trabajo se pretende diseñar el control digital síncrono del motor DC presente en una cinta de correr utilizando una FPGA Altera Cyclone IV EP4CE22F17C6N.

El objetivo del proyecto será un sistema que permita controlar la puesta en marcha, apagado, selección de programas y velocidad del motor de una cinta de correr.

La comunicación con el usuario se establecerá mediante una botonera y un display. El usuario seleccionará la puesta en marcha, el apagado, la velocidad de la cinta y distintos programas de ejercicio mediante la botonera. A su vez, en el display se mostrará el programa de ejercicio actual y la velocidad de la cinta.

Para ello, se utilizará una tarjeta DE0-Nano, un display LCD HITACHI HD 44780U y una etapa de potencia en puente en H. También se hará uso de un programa que controla los disparos de la etapa de potencia.

En primer lugar se diseñará la lógica del circuito para el control de la referencia de velocidad en los distintos programas y el control del display. A continuación el diseño se programará utilizando el software Quartus II y se simulará con ModelSim. Posteriormente se implementará el diseño en la tarjeta.

Finalmente se realizarán las pruebas en la DE0-Nano, que está conectada a la botonera, display y a la etapa de potencia en H que controla el motor DC y se depurarán posibles fallos.

2 Antecedentes, motivación y justificación

2.1 Motor eléctrico

Werner von Siemens patentó la dinamo en 1866, con esto contribuyó al inicio y desarrollo de los motores eléctricos que hoy en día se conocen.

Los motores eléctricos son máquinas que transforman la energía eléctrica en energía mecánica. Esta transformación se consigue mediante los campos magnéticos formados en las bobinas de los motores.

2.1.1 Motor eléctrico de corriente continua

El motor de corriente continua se compone principalmente de dos partes, el estator y el rotor. En el estator encontramos los devanados principales de la máquina (también llamados polos), pudiendo ser de imanes permanentes o devanados con hilo de cobre sobre núcleo de hierro (figura 2.1). El rotor también posee un devanado alrededor del núcleo, alimentado con corriente continua.

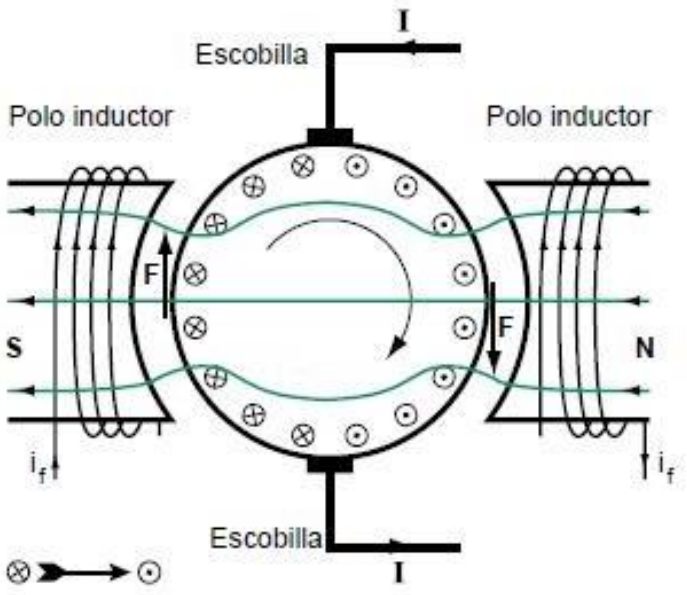

Figura 2.1 Motor de corriente continua

Cuando un conductor (devanado del rotor) se expone a un campo magnético, se ve sometido a una fuerza perpendicular al plano formado por la dirección de la corriente y el campo magnético, teniendo en cuenta la regla de la mano derecha.

$$F = B \cdot l \cdot I$$

Siendo:

- **F:** Fuerza en newtons

- **I:** Intensidad que recorre el conductor en amperios
- **l:** Longitud del conductor en metros
- **B:** Densidad del campo magnético o densidad de flujo en teslas

Uno de los grandes inconvenientes de estas máquinas es el alto mantenimiento que necesitan. Con esto se refiere al cambio de carbones, pulido de colector, limpieza de motores y alto desgaste. Sin embargo, en los motores de corriente alterna el mantenimiento es prácticamente nulo, y la duración de los mismos suele ser de 3 a 4 veces mayor. Pese a estos inconvenientes, los motores de corriente continua siguen predominando en cuanto a cintas de correr se refiere. Esto es debido a su mejor respuesta ante amplias variaciones de carga y velocidad de giro.

2.2 Control de los motores de corriente continua

Para controlar el par y la velocidad de giro de un motor de continua se necesita poder variar la tensión que le llega al motor.

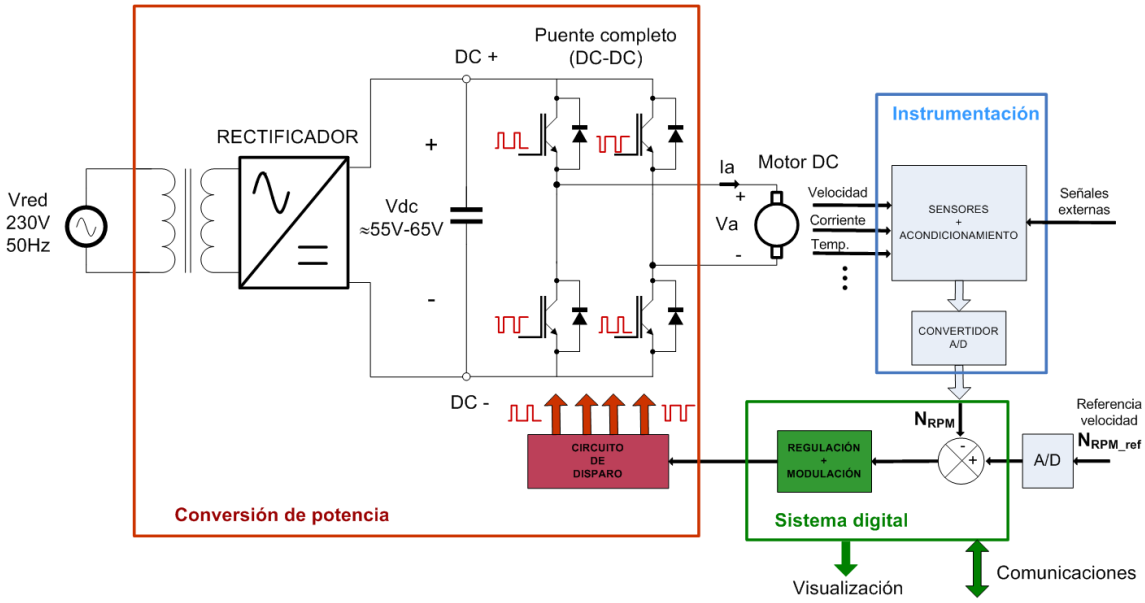

Figura 2.2 Diagrama de bloques de un variador de velocidad para un motor de corriente continua

En la figura 2.2 se muestra el diagrama de bloques de un variador de velocidad para un motor de corriente continua. En primer lugar se puede observar el puente en H (conjunto formado por los 4 transistores), mediante el cual se puede variar el voltaje aplicado al motor. Para gestionar la apertura o cierre de estos transistores se utiliza la modulación de ancho de pulso (PWM).

2.2.1 Puente en H

El puente en H es un circuito eléctrico que permite a un motor eléctrico de corriente continua variar la velocidad y el sentido de giro.

Como se puede observar en la figura 2.3, para hacer girar al motor en sentido directo se deberían activar los transistores S1 y S4; así, para hacer girar el motor en sentido inverso se deberían activar los transistores S2 y S3. Si se quiere hacer girar el motor a una velocidad menor que la máxima en sentido directo, se debería alternar la activación de los transistores S1 y S4 con la activación de los transistores S2 y S3, activando, en este caso, durante más tiempo los transistores S1 y S4.

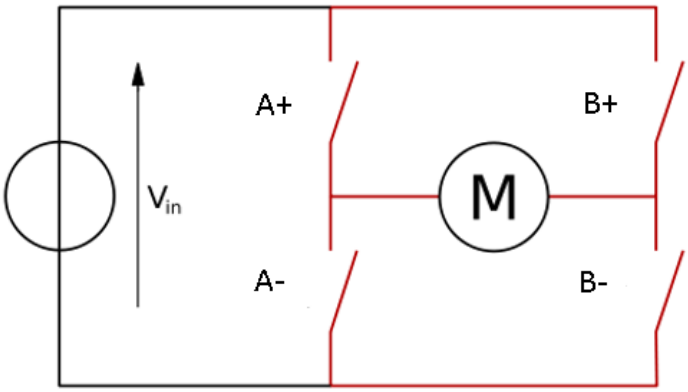

Figura 2.3 Esquema Puente en H

2.2.2 Modulación por ancho de pulso (PWM)

Como se ha comentado anteriormente, el control de la apertura y cierre de los transistores del puente en H se realiza mediante la modulación PWM.

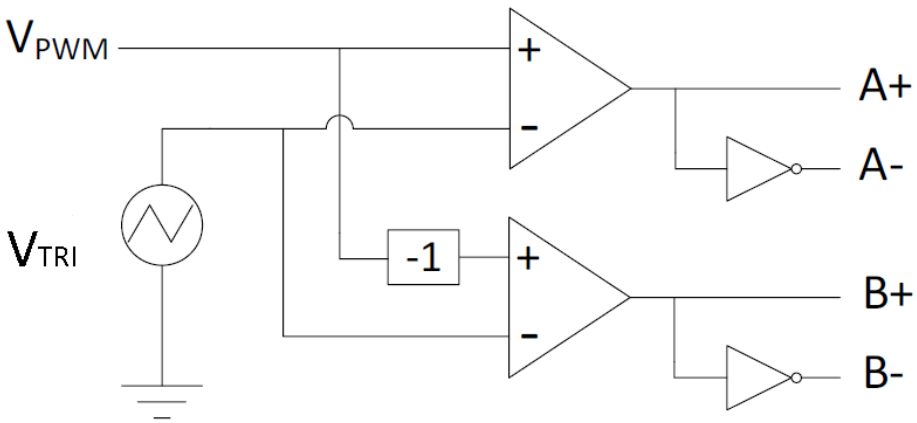

Figura 2.4 Esquema modulación PWM unipolar

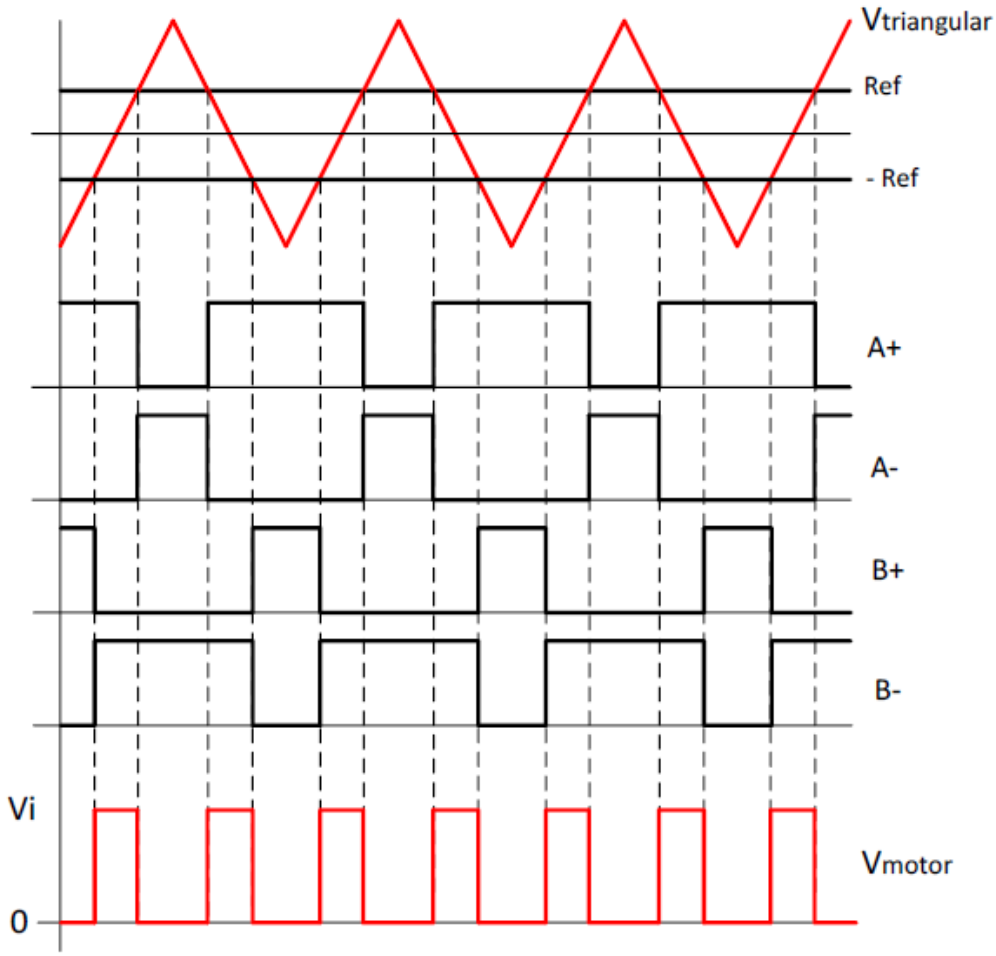

Figura 2.5 Señales PWM

En las figuras 2.4 y 2.5 se puede observar como a partir de una referencia y una señal triangular se pueden obtener las señales de activación de los transistores del puente en H. El ciclo de trabajo debe ser muy rápido para que el motor no perciba estos cambios y actúe simplemente como si le llegara un voltaje constante cuyo valor fuese la media del voltaje en el ciclo. Por último, se ha de tener especial cuidado con que no se activen a la vez los transistores A+ y A- ni tampoco los transistores B+ y B-, ya que esto provocaría un cortocircuito. Por lo tanto, la conmutación entre los diferentes transistores se ha de realizar teniendo en cuenta esta limitación y se ha de dejar un tiempo muerto en el que no estén activados ninguno de los dos.

2.3 Field Programmable Gate Array (FPGA)

Una FPGA es un circuito integrado que contiene bloques de lógica programable. Estos bloques de lógica pueden ser interconectados y programados según las necesidades del

programador después del proceso de manufactura, tantas veces como sea necesario. Por tanto, una misma FPGA puede ser utilizada para múltiples aplicaciones.

2.3.1 Programación

Para configurar las celdas de la FPGA con funciones específicas (función lógica AND, OR, multiplexor, etc…), el programador debe describir el hardware que tendrá dicha FPGA.

Para realizar la programación, el programador cuenta con entornos de desarrollo especializados en su diseño e implementación en la FPGA. En este trabajo se ha utilizado el programa Quartus II.

El diseño se puede programar mediante esquemáticos o mediante un lenguaje de alto nivel específico, conocidos como HDL, siendo los más utilizados:

- VHDL
- Verilog
- ABEL

Se ha optado en este caso por realizar la programación mediante esquemáticos.

2.3.2 Aplicaciones

Las aplicaciones más comunes de las FPGA incluyen:

- Procesamiento digital de señales
- Sistemas aeroespaciales y de defensa
- Prototipos de ASICs
- Sistemas de imágenes para medicina
- Sistemas de visión para computadoras
- Reconocimiento de voz
- Bioinformática
- Emulación de hardware de computadora

Como se puede observar, el rango de aplicaciones de las FPGA es muy amplio, debido a la versatilidad y flexibilidad de estos dispositivos, cuyo uso es cada vez mayor, sobretodo en aplicaciones que requieren un alto nivel de paralelismo.

2.4 Placa de potencia

Para el presente proyecto se ha utilizado la placa de potencia de la figura 2.6. En esta placa se encuentran los transistores del puente en H, utilizados para variar la tensión en el motor. Las señales de activación de estos transistores provendrán de la FPGA.

También se puede observar en esta placa la presencia de los elementos fundamentales de este trabajo:

- Tarjeta DE0-Nano
- Botonera acoplada a la placa
- Pantalla de cristal líquido (LCD)

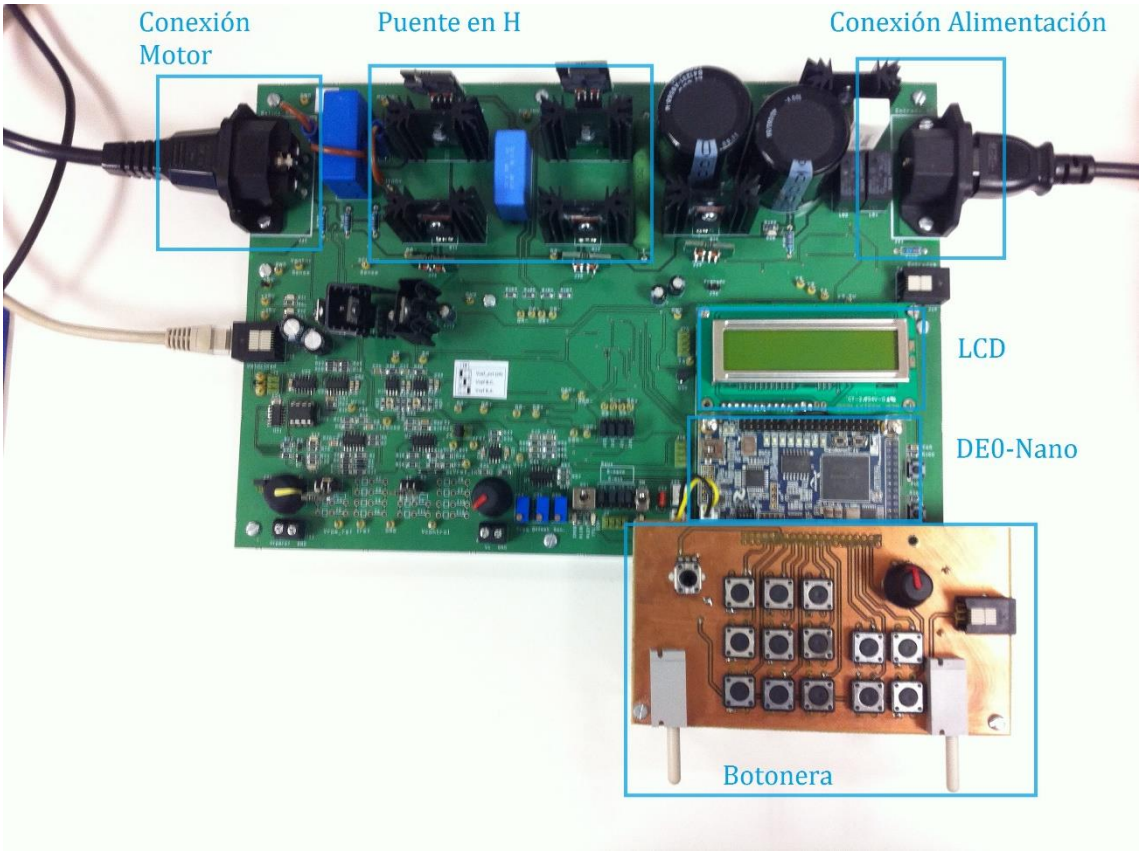

Figura 2.4 Placa de potencia

2.4.1 Tarjeta DE0-Nano

La tarjeta DE0-Nano es una plataforma para la fácil utilización y programación de la FPGA. Se puede observar en la figura 2.7 el aspecto de la tarjeta. Esta tarjeta incluye una FPGA Altera Cyclone IV EP4CE22F17C6N, la cual dispone de un máximo de 153 pines de entrada/salida, suficientes para la aplicación que se va a desarrollar.

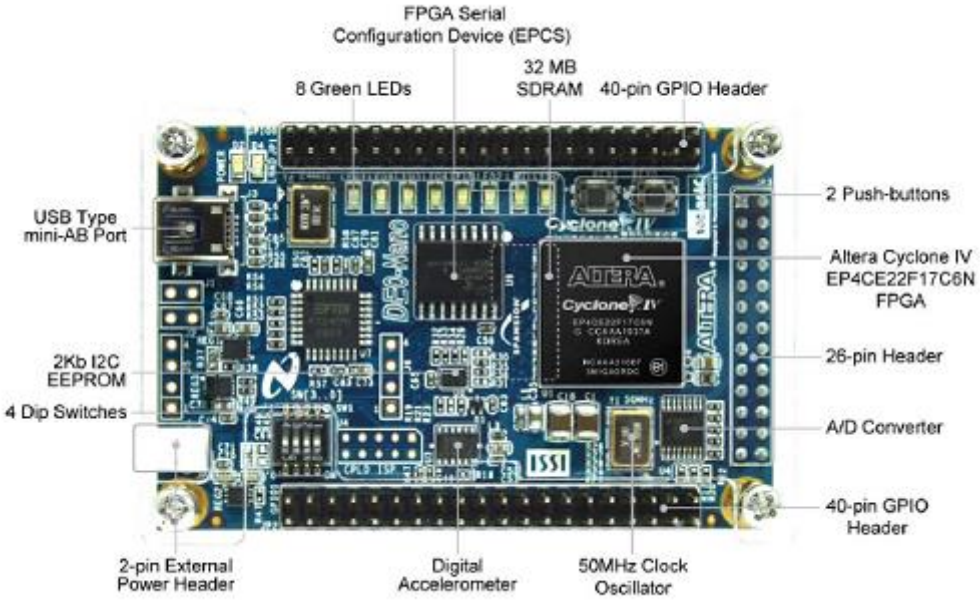

Figura 2.5 Tarjeta DE0-Nano

También incluye los siguientes elementos:

- **USB-Blaster:** Utilizado para la programación de la tarjeta.
- **2 módulos de expansión de 40 pines:** 72 pines entrada/salida, 1 pin 5V, 2 pines 3.3V y 4 pines conectados a tierra.
- **Memoria:** La tarjeta dispone de 32MB de SDRAM y 2Kb de EEPROM.
- **Reloj de 50MHz**
- **Conversor A/D**
- **Otras entradas/salidas:** 8 leds verdes y 2 botones.

2.4.2 Botonera

Como se puede apreciar en la figura 2.6, se ha incorporado a la placa de potencia una botonera, de la cual se van a utilizar para este trabajo 6 pulsadores. La botonera cuenta con una alimentación a 3.3V. Cada pulsador tiene como salida 3.3V (1 lógico) en el caso de no estar pulsado y 0V (0 lógico) si está pulsado. Cada pulsador cuenta con un condensador anti-rebote para evitar el pulsado accidental de los botones.

La integración de la botonera se realiza mediante uno de los módulos de expansión que posee la tarjeta DE0-Nano.

2.4.3 Pantalla de cristal líquido (LCD)

Por último, se puede observar en la figura 2.6 la presencia de una pantalla de cristal líquido o LCD (liquid crystal display). Es el visualizador HD44780U, de la marca HITACHI, mediante el cual se pueden mostrar mensajes formados por distintos caracteres. Este display tiene un formato de 2x16 (2 filas y 16 columnas).

Los LCD vienen gobernados por un microcontrolador, que es el que se encarga de gestionar el display, mientras que el programador únicamente tiene que conocer las instrucciones que debe enviar al controlador (lenguaje de alto nivel) para hacerlo funcionar.

En cuanto a la memoria, posee una memoria ROM en la que están almacenados todos los caracteres que se pueden utilizar. Además, el fabricante reserva una zona de memoria RAM por si el programador necesita algún carácter especial que no esté en la memoria ROM.

3 Descripción del Diseño del Software

Como se ha comentado anteriormente, para realizar el diseño se ha utilizado el programa Quartus II, y se va a programar mediante esquemáticos.

En la figura 3.1 se observa el TOP del programa, en el que se ven dos bloques. El bloque superior (control_motor_completo) ha sido facilitado por el departamento de electrónica, ya que su implementación supera el alcance del trabajo. Por tanto el bloque inferior (TFG_TOP) es el que se ha diseñado durante este Trabajo Fin de Grado.

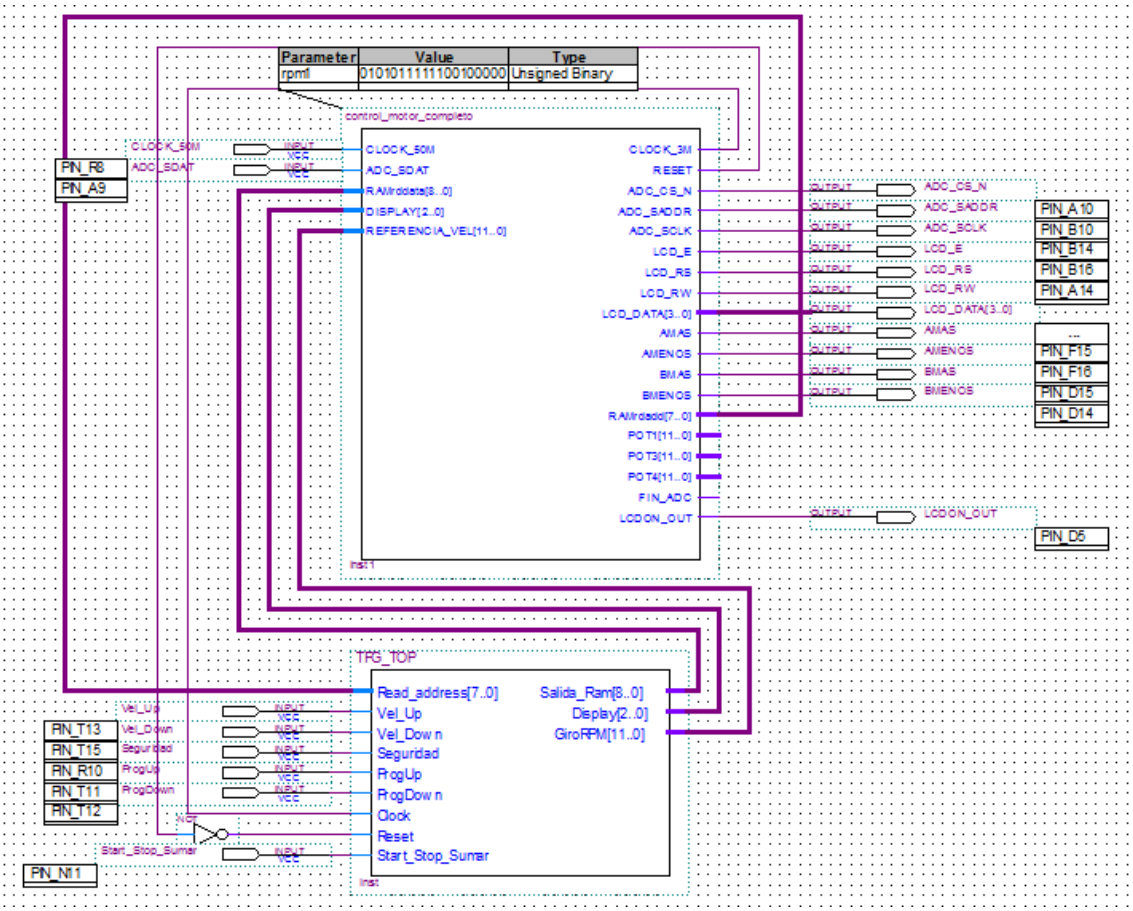

Figura 3. 1 Esquemático TOP

3.1 Bloque control_motor_completo

Como ya se ha comentado, este bloque, que aparece ampliado en la figura 3.2, ha sido facilitado por el departamento de electrónica debido a que su implementación supera el alcance del trabajo. Las funciones de este bloque son las siguientes:

- **Generar un reloj de 3MHz** (CLOCK_3M) a partir del reloj de 50MHz que le llega a la FPGA mediante un PLL.

- **Generar las señales de activación de los transistores del puente en H** (AMAS, AMENOS, BMAS Y BMENOS) para variar el voltaje del motor a partir de una referencia de velocidad (en 12 bits) recibida del bloque que se ha diseñado en este trabajo.
- Realizar la **lectura** de una **memoria RAM** utilizada en el bloque de este trabajo en la que se encuentran los datos que se quieren mostrar en el LCD, mediante las señales RAMrdadd[7..0] y RAMrddara[8..0]. Posteriormente **generar y enviar** las **señales necesarias al procesador del LCD** para mostrar estos datos (señales LCD_E, LCD_RS, LCD_RW Y LCD_DATA[3..0]).

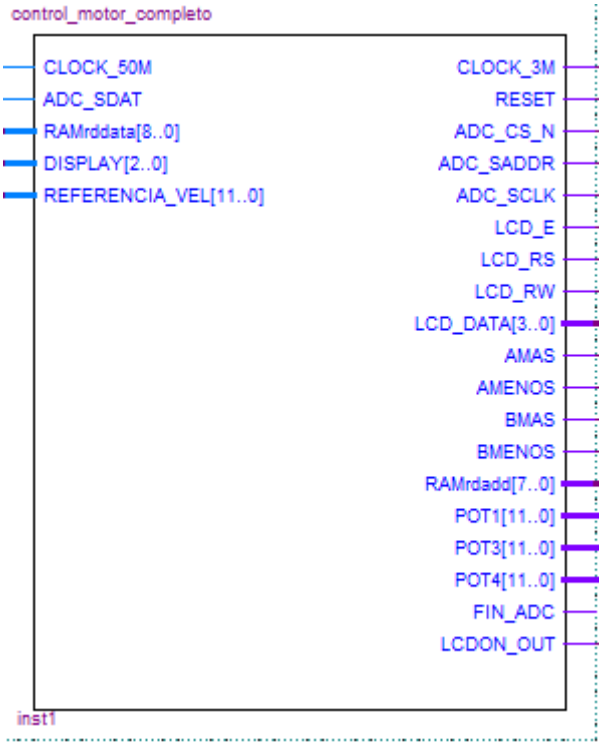

Figura 3. 2 Esquemático Bloque Control_motor_completo

3.2 Bloque TFG_TOP

El bloque realizado durante el presente trabajo es el mostrado en la figura 3.3.

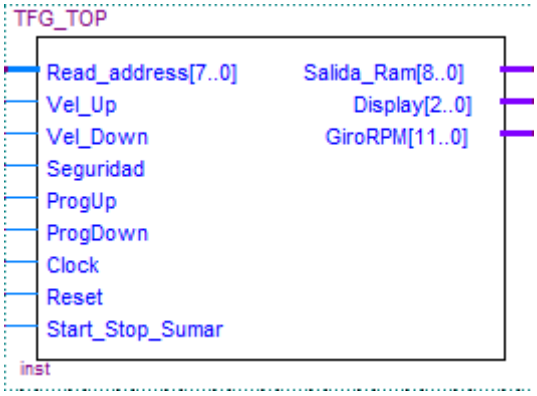

Figura 3. 3 Bloque TFG_TOP

Las señales de entrada de este bloque son las siguientes:

- Read_address[7..0]: señal enviada desde el bloque control_motor_completo para hacer la lectura de la memoria RAM. Esta señal especifica la dirección de memoria que quiere leer.
- Vel_Up: señal que proviene de un pulsador utilizada para subir la velocidad del motor. Este pulsador sólo podría utilizarse en el caso de que el sistema se encontrase en el programa que se ha llamado LIBRE (en la sección programas se describirán los diferentes programas).
- Vel_Down: su comportamiento sería el mismo que Vel_Up, pero en este caso sirve para bajar la velocidad del motor.
- Seguridad: esta señal simula el comportamiento de un sistema del que suelen disponer las cintas modernas. Para que la cinta funcione debe estar conectada una llave de seguridad, que viene con un cordón que se debe llevar atado a la muñeca. En el caso de caerse de la cinta, la llave se desconectaría y la cinta se pararía. En este caso el botón sin pulsar equivale a la llave conectada y el botón pulsado equivale a la llave desconectada.
- ProgUp: señal que proviene de un pulsador que se utiliza para pasar al programa siguiente.
- ProgDown: mismo funcionamiento que ProgUp pero es utilizado para retroceder al anterior.
- Clock: señal de reloj de 3MHz.
- Reset: señal para resetear el programa, activa a nivel bajo.
- Start_Stop_Sumar: señal que proviene de un pulsador que se utiliza para empezar o finalizar un programa llamado SUMAR (en la sección programas se describirán los diferentes programas).

Las señales de salida de este bloque son entradas del bloque control_motor_completo (figura 3.1) y son las siguientes**:**

- Salida_Ram[8..0]: mediante esta señal se envía el dato contenido en la memoria que se ha especificado a través de la señal Read_Address[7..0].
- Display[2..0]: señal utilizada para elegir el número de pantalla que se quiera visualizar en el LCD. Se ha programado de modo que el número de pantalla coincide con el número de programa, así el dato que le pasamos es el número del programa en el que nos encontramos. Puesto que en el bloque control_motor_completo esta señal es de 3 bits, se ha tenido que añadir un bit de valor 0 en la posición más alta, ya que nuestra señal de programa únicamente tiene 2 bits (figura 3.4).
- GiroRPM[11..0]: señal de 12 bits que indica la velocidad del motor y que le pasamos al bloque proporcionado por el departamento para que dicho bloque genere las señales para el puente en H.

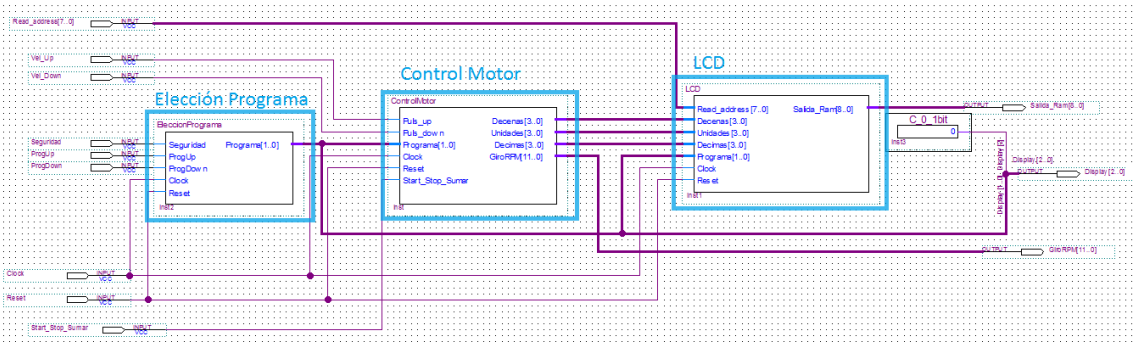

Figura 3. 4 Esquema Interior TFG_TOP

En la figura 3.4 se puede observar la vista interior del bloque final del trabajo que consta de tres bloques. El bloque de la izquierda es el bloque utilizado para la elección del programa (llamado Elección Programa), el bloque central es utilizado para control del motor (llamado Control Motor), y, por último, el bloque de la derecha es el bloque utilizado para escribir en la memoria RAM que posteriormente se enviará al LCD (llamado LCD). Se va a proceder a la explicación de cada bloque durante los siguientes puntos.

3.2.1 Bloque Elección Programa

En la figura 3.5 se pueden observar las entradas y salidas que tiene este bloque. El funcionamiento de este bloque es simple, si se activa la señal ProgUp, se pasa al programa siguiente, si se activa la señal ProgDown, se pasa al programa anterior.

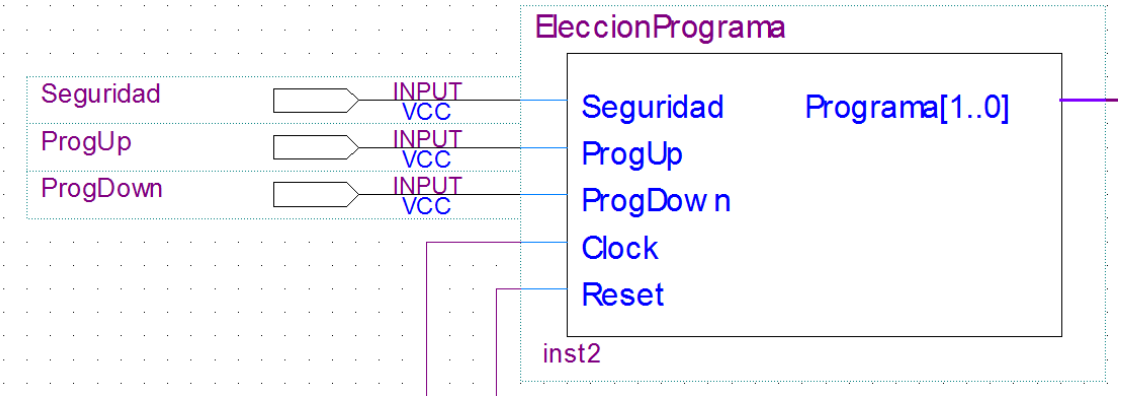

Figura 3. 5 Bloque Elección Programa

En este trabajo únicamente se dispone de 3 programas, por lo tanto la salida Programa[1..0] tendría los siguientes valores:

- 0: No hay programa
- 1: Programa LIBRE
- 2: Programa SUMAR

Solamente harían falta 2 bits, como se aprecia en el nombre de la salida.

En la figura 3.6 se pueden ver los tres tipos de bloque necesarios para la implementación del bloque ElecciónPrograma (Bouncer Retardo, Bouncer Pulso y la Máquina de Estados que controla el bloque).

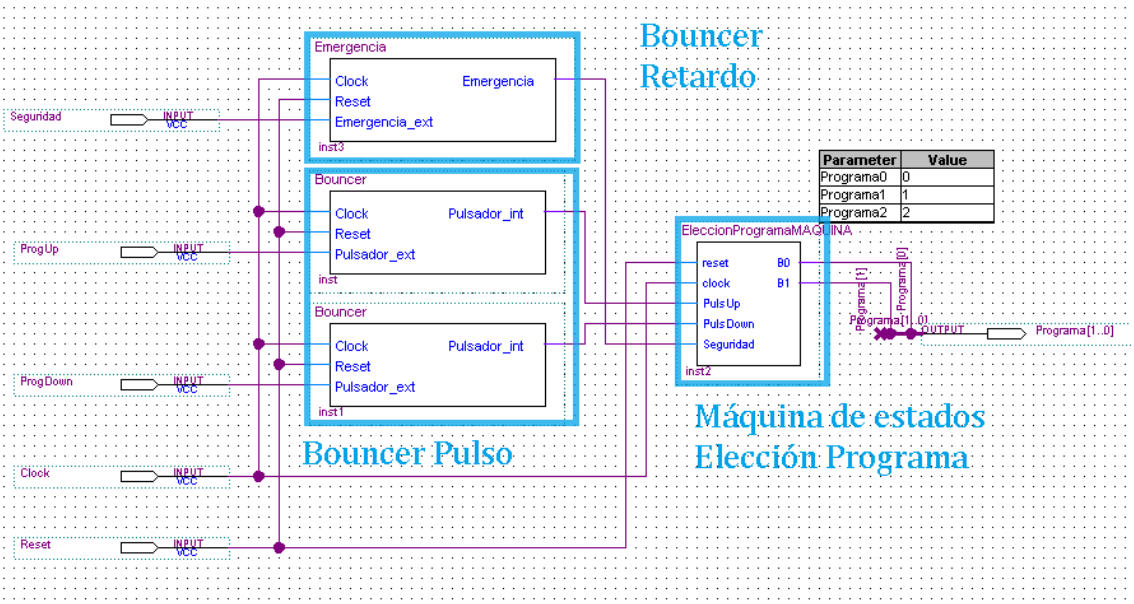

Figura 3. 6 Esquema Elección Programa

3.2.1.1 Máquina de estados Elección Programa

Cada uno de los tres estados de la Máquina de estados Elección Programa corresponde a cada uno de los programas y sus entradas y salidas son las siguientes:

- Entradas: Son las señales provenientes de los pulsadores después de aplicarles el software anti-rebote llamado Bouncer (explicado en el siguiente punto).

- Salida: Consiste en dos bits de salida, cuya concatenación forma el número de programa en el que se encuentra el usuario.

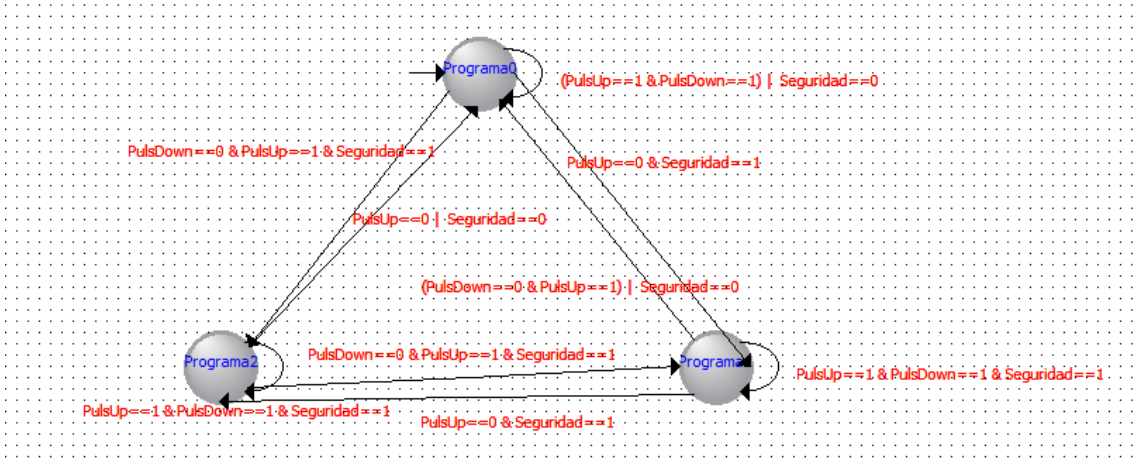

Figura 3. 7 Máquina de Estados Elección Programa

El funcionamiento de la máquina de estados (figura 3.7) es el siguiente: El estado inicial es el llamado Programa 0 (estado en el que no hay programa). Para salir de este estado debe estar enchufada la llave de seguridad (Seguridad=1). Dependiendo del valor de las señales PulsUp y PulsDown se produce la transición hacia los estados Programa (Programa LIBRE) y Programa2 (Programa SUMAR). En concreto, si se activa la señal PulsUp se pasa al programa siguiente, y si se activa la señal PulsDown se pasa al programa anterior. Como se ha comentado existen los siguientes programas:

- 0: No hay programa (sistema encendido, pero la cinta sin movimiento).
- 1: Programa LIBRE
- 2: Programa SUMAR

Es posible pasar del programa 2 al programa 0 únicamente activando la señal PulsUp. También es posible pasar del programa 0 al programa 2 activando la señal PulsDown (figura 3.7).

3.2.1.2 Bouncer

Pese a los condensadores anti-rebote que poseen los pulsadores, las señales que provienen de los mismos (llamadas por el diseñador pulsador externo) no son señales limpias, la transición pulsado/no pulsado presenta múltiples rebotes. Para eliminar estos rebotes se utiliza un software anti-rebote llamado Bouncer.

Todos los pulsadores son activos a nivel bajo. La pulsación de uno de los pulsadores puede provocar que la tensión en el resto caiga hasta un nivel entendido como un 0, aunque sea sólo durante un instante. Por la tanto se ha de aplicar el software Bouncer a todos los pulsadores, para que siempre se espere 20 ms después de recibir el primer 0, así no confundir una pulsación real con una caída de tensión provocada por otro pulsador.

A la señal interna del pulsador creada por el diseñador se le llamará pulsador interno.

Muchas veces la duración del pulso tampoco es la deseada, así que el bouncer también se utiliza para modificar la duración la duración de los pulsos.

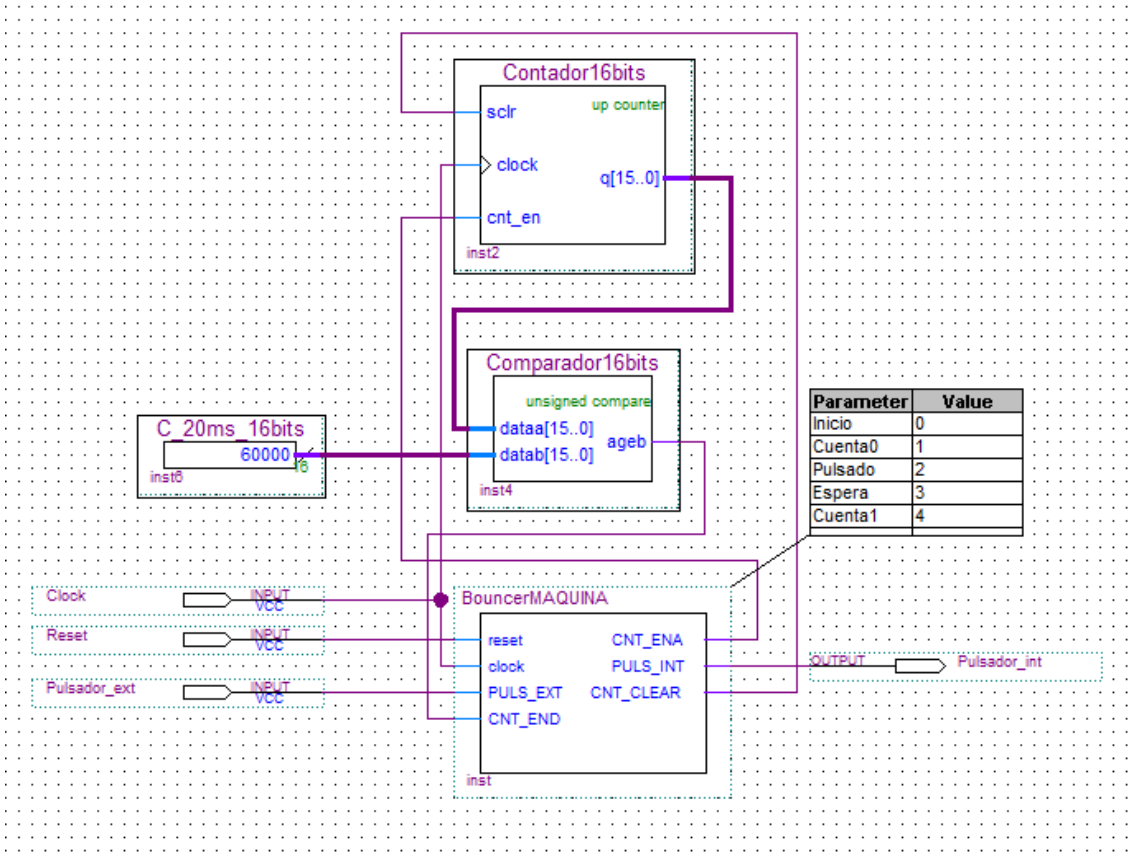

Figura 3. 8 Esquemático Bouncer

En la figura 3.8 se muestra el esquemático típico de los bouncers utilizados. En todos los casos, una vez se recibe un nivel bajo por parte del pulsador, se espera 20 ms para comprobar que ha sido realmente una pulsación del pulsador y no una caída provocada por otro pulsador. Si al pasar los 20 ms sigue a nivel bajo, se sabe que realmente ha sido pulsado y se activa el pulsador interno la duración que el diseñador decida.

3.2.1.2.1 Bouncer Pulso

En el caso de los pulsadores de subir y bajar programa (PulsUp y PulsDown), se necesita que los pulsos sean de un ciclo, ya que si el pulso interno se mantuviera hasta que se suelta el pulsador, se aumentaría o disminuiría de programa más veces de las deseadas. Por tanto, la función de la máquina de estados del **Bouncer Pulso** (Figura 3.6) es generar un pulso de un ciclo una vez se ha comprobado que ha sido una pulsación real del pulsador.

3.2.1.2.2 Bouncer Retardo

En el caso de la señal de seguridad, se desea que la señal interna se mantenga mientras siga pulsado el botón, ya que si la llave de seguridad no está metida, debe dar constantemente un 0 lógico (botón pulsado).

Por tanto, una vez han pasado los 20 ms para comprobar que es una pulsación real del pulsador, la señal interna pasa a 0, quedando a la espera de que se deje de pulsar el botón, momento en el que volvería a esperar 20 ms para ver si se ha dejado de pulsar realmente. Una vez pasado este tiempo, si el botón sigue estando sin pulsarse, la señal interna volvería a 1. Por este motivo se ha elegido el nombre **Bouncer Retardo** (Figura 3.6), porque la señal interna es una señal perfecta que va retrasada con respecto a la señal externa 20 ms.

3.2.2 Bloque Control Motor

El Bloque Control Motor (figura 3.9) controla la referencia de velocidad que se le pasa al bloque que controla la modulación PWM (control_motor_completo). Además, genera también los dígitos de las decenas, unidades y décimas en BCD correspondientes a la velocidad (en km/h). Posteriormente estos dígitos se pasarán al control del LCD para mostrarlos en el display. El bloque ControlMotor tiene las siguientes señales de entrada y de salida:

Entradas:

- PulsUp
- PulsDown
- Programa: esta entrada es la salida del bloque de Elección de Programa explicado en el punto anterior.
- Start_Stop_Sumar

Salidas:

- GiroRPM[11..0]: referencia del motor.
- Decenas[3..0]: valor de las decenas de la velocidad en 4 bits (velocidad en km/h).
- Unidades[3..0]: valor de las unidades de la velocidad en 4 bits.
- Décimas[3..0]: valor de las décimas de la velocidad en 4 bits.

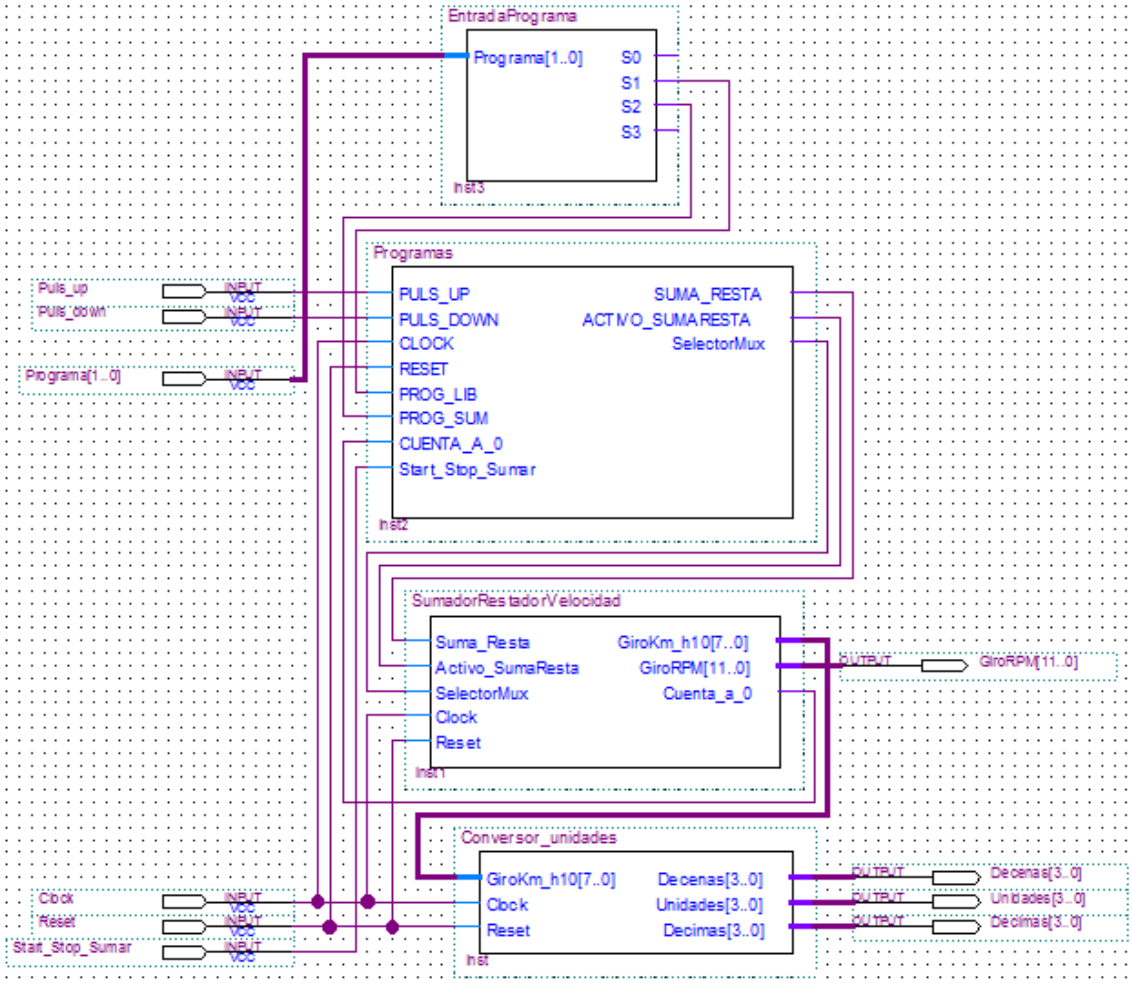

Figura 3. 9Vista Interna del Bloque de Control Motor

Como se puede observar en la figura 3.9, el funcionamiento de este bloque se consigue gracias a la interconexión de varios bloques, ya que hay que controlar la conmutación entre los programas, la programación de los mismos, la suma/resta de la referencia de velocidad y la separación de cada dígito de la referencia de velocidad en kilómetros por hora para el posterior envío al LCD.

3.2.2.1 Entrada Programa

Este bloque (figura 3.9) es utilizado para seleccionar el programa de la cinta a ejecutar. Si el usuario selecciona el programa 1, se activara el bit 1, si selecciona el programa 2, se activará el bit 2.

Para implementar este bloque se ha utilizado un decodificador de 2 bits (figura 3.10), el cual dispone de 4 salidas. En el presente caso, sólo se utilizarán 2 (S1 para el programa LIBRE y S2 para el programa SUMAR). El programa LIBRE será el programa 1 y el programa SUMAR será el programa 2. El programa 0 corresponde al estado en el que no hay programa y no se le asigna ninguna salida.

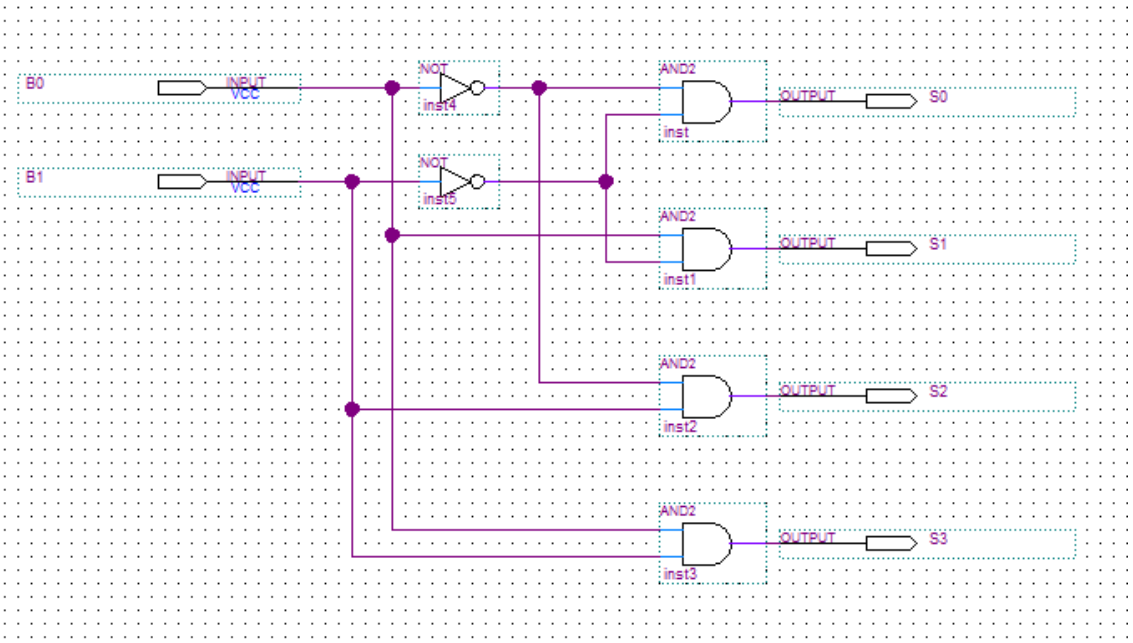

Figura 3. 10 Decodificador de 2 bits

En la figura 3.10 se puede observar la lógica del decodificador de dos bits. Posteriormente, se utilizarán estas señales de programa individuales (S1 y S2) para realizar el paso entre los programas de la cinta, las cuales son muy necesarias porque hasta que el sistema no se encuentre preparado para cambiar de programa, no cambiará.

3.2.2.2 Sumador/Restador de Velocidad

En la figura 3.11 se pueden ver las entradas y salidas de las que dispone el bloque Sumador/Restador de velocidad.

Las entradas son las siguientes:

- Suma_Resta: señal que se conecta a un sumador/restador (figura 3.12*)*. En el caso de que valga 0, el sumador/restador resta el valor de sus dos entradas, y en el caso de que valga 1, el sumador/restador suma el valor de sus dos entradas.
- Activo_SumaResta: señal que activa la memorización del valor de salida del sumador/restador.
- SelectorMux: señal que se conecta a un multiplexor (figura 3.12*)* para elegir el valor que se desea sumar/restar a la velocidad. En el caso de que valga 0, se suma/resta 0.1 km/h. En el caso de que valga 1, se suma/resta 1km/h.

El bloque dispone de las siguientes salidas:

- GiroKm_h10: valor de la velocidad de la cinta en km/h multiplicado por 10. Este valor se multiplica por 10 para poder conseguir hasta las décimas de la velocidad sin tener que trabajar con decimales.
- GiroRPM: valor de la velocidad del motor en revoluciones por minuto.
- Cuenta_a_0: señal que se activa cuando el motor está parado.

Figura 3. 11 Bloque Sumador/Restador de Velocidad

Durante el diseño de este bloque se ha programado la lógica necesaria para, a partir de las entradas mencionadas, obtener, por una parte el valor de la velocidad del motor en revoluciones por minuto para enviarla al bloque control_motor_completo (figura 3.1), y, por otra, el valor de la velocidad de la cinta en kilómetros por hora multiplicado por 10, para enviarla al bloque que separa cada dígito del valor (figura 3.9).

También se genera la señal que indica que el motor está parado, muy útil para para el diseño de los distintos programas de ejercicio, ya que en todos los programas se comienza con el motor detenido.

Durante el trabajo se utiliza la velocidad en kilómetros por hora multiplicado por 10, así, si se quiere aumentar la velocidad en 0.1 km/h, se debe aumentar en una unidad el valor de la velocidad de trabajo. Si se quiere aumentar en 1 km/h, se debe aumentar esta velocidad en 10 unidades.

El motor tiene una velocidad máxima de giro de 4000 rpm, y se ha supuesto que esta velocidad máxima de giro se corresponde con una velocidad máxima de la cinta de 25 km/h.

Se puede observar que para pasar de la velocidad en km/h multiplicada por 10 a la velocidad de giro del motor, únicamente hay que multiplicar por 16. Esta multiplicación en binario es muy simple, ya que solamente hay que concatenar 4 ceros al final del valor de la velocidad en km/h multiplicada por 10 para obtener la velocidad en rpm.

Posteriormente se limitó la velocidad del motor a 3400 rpm, por razones mecánicas. Frente a esta nueva situación se decidió mantener la relación entre la velocidad del motor y la de la cinta y, por lo tanto, la velocidad máxima de la cinta pasó a ser de 21.2 km/h.

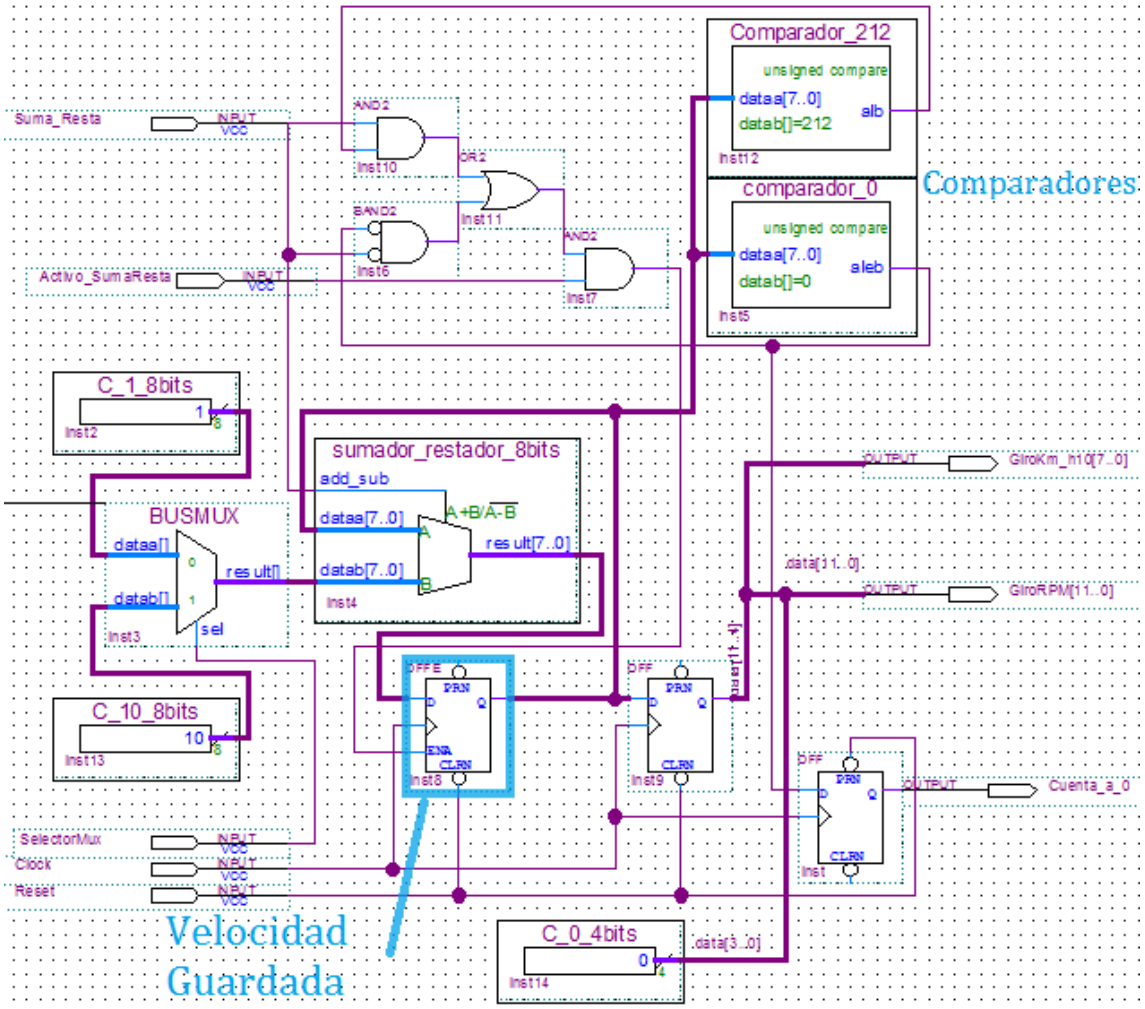

Figura 3. 12 Esquemático del Bloque Sumador/Restador de Velocidad

En la figura 3.12 se muestra el esquema del bloque Sumador/Restador de velocidad.

Para realizar las modificaciones de la velocidad se ha de activar la puerta ENABLE del biestable que está a la salida del sumador/restador, que aparece recuadrado en la figura 3.12. En este biestable se almacena la velocidad actual de la cinta. A la entrada del biestable se tiene el resultado de la suma/resta (depende de lo que valga la señal SUMA/RESTA) entre la velocidad actual y el dato a sumar/restar (depende de lo que valga la señal SelectorMux). Para que ese resultado se guarde en la velocidad, se ha de activar la puerta ENABLE de dicho biestable (se guarda el valor de la entrada).

Se observa la presencia de dos comparadores (comparador_212 y comparador_0). Estos comparadores se utilizan como limitadores. Para realizar esta función se utilizan las puertas AND, OR y BAND presentes en la figura.

El comparador_212 actúa cuando la velocidad es la máxima (21.2 km/h) y la señal Suma_Resta está puesta a 1 (sumar). En este caso, aunque se active la señal Activo_SumaResta, esta señal no llegaría a activar la puerta en ENABLE del biestable que guarda la velocidad del motor, lo que evitaría la subida de la velocidad por encima del valor máximo.

El comparador_0 actúa cuando la velocidad es 0 km/h y la señal Suma_Resta está puesta a 0 (restar). En este caso, aunque se active la señal Activo_SumaResta, esta señal tampoco llegaría a activar la puerta ENABLE del biestable, lo que evitaría que la velocidad bajase por debajo de cero.

3.2.2.3 Programas

La función de este bloque es, a partir de las señales de entrada que se van a comentar, y según el programa de ejercicio actual, controlar las señales de suma y de resta para la referencia del motor.

En la figura 3.13 se pueden observar las señales de entrada y las señales de salida que tiene este bloque.

Entradas:

- PulsUp
- PulsDown
- Prog_Lib: señal que vale 1 si el usuario ha elegido el programa 1 (programa LIBRE).
- Prog_Sum: señal que vale 1 si el usuario ha elegido el programa 2 (programa SUMAR).
- Cuenta_a_0: Señal que vale 1 cuando el motor está parado.
- Start_Stop_Sumar: señal que controla el inicio y fin del programa SUMAR.

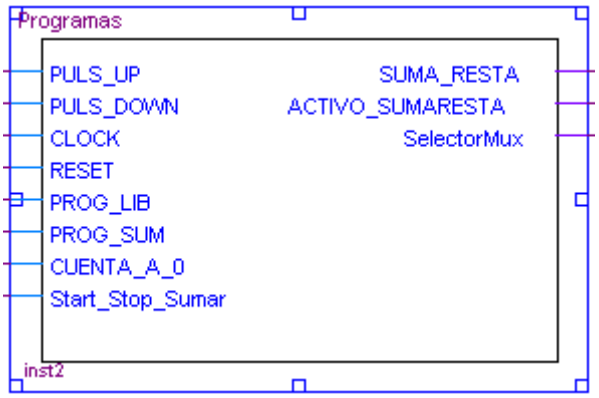

Figura 3. 13 Bloque Programas

Salidas:

- SUMA_RESTA: se utiliza en el bloque de suma/resta de velocidad (figura 3.11), entendiendo un 0 como resta, y un 1 como suma.
- ACTIVO_SUMARESTA: se utiliza también el en bloque suma/resta de velocidad. Mientras esta señal esté a 0, el bloque no sumará ni restará, y una vez esté a 1, sumará o restará en función del valor de la seña SUMA_RESTA.
- SelectorMux: cuando valga 0, se sumará de 0.1 km/h en 0.1 km/h y cuando valga 1, se sumará de 1 en 1.

En la figura 3.14 se ve el esquema interno del bloque Programas.

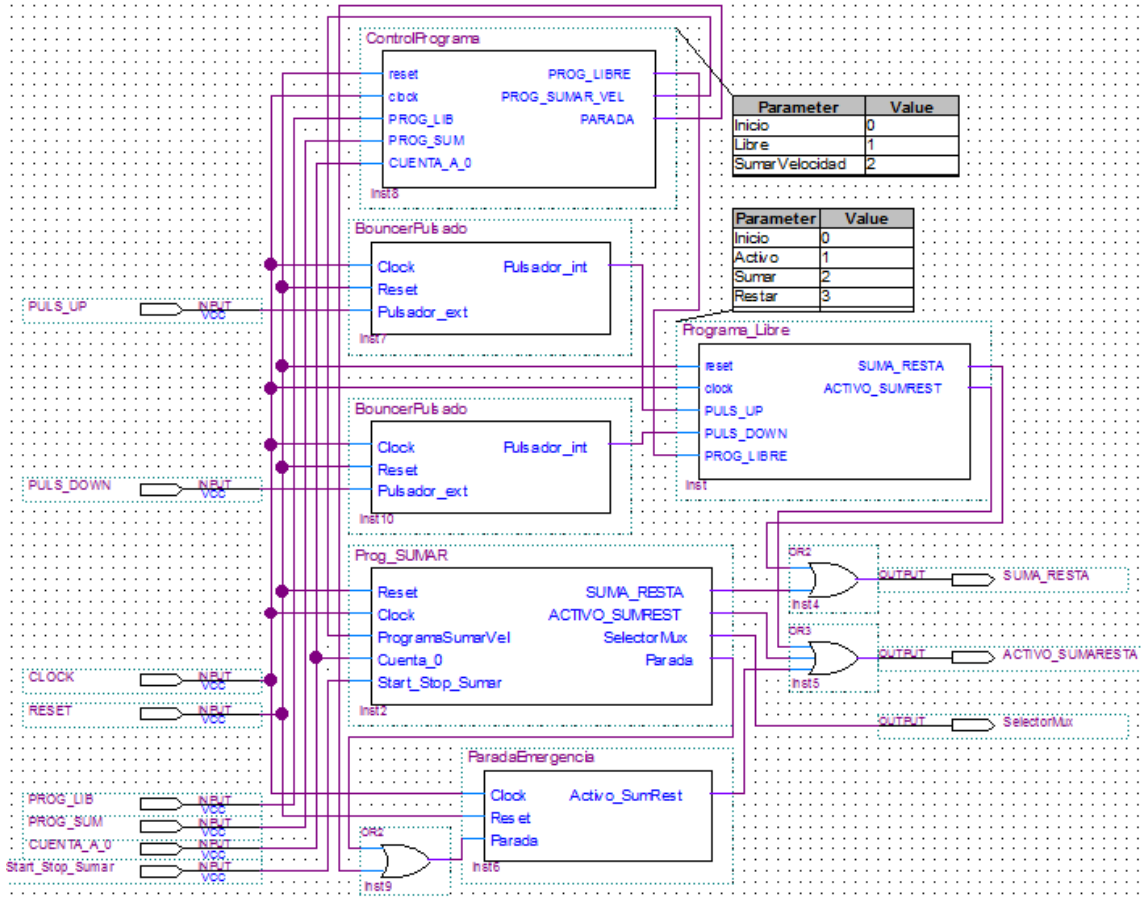

Figura 3. 14 Vista Interna de Bloque Programas

El diseño se ha implementado de modo que un cambio de programa implique, primero, que el motor se detenga, y que, posteriormente, se inicie el programa deseado por el usuario.

La máquina de estados “Control Programa” (figura 3.15) se encarga de realizar esta función. Una vez le ha llegado el bit de activación del programa deseado por el usuario (PROG_LIB o PROG_SUMAR_VEL), pasa a un estado en el que se activa la Parada de Emergencia (estado Inicio), y en el momento en el que el motor se ha parado (momento en el que la señal Cuenta_a_0 se activa), ya se pasa al estado en el que se activa el bit de activación que va directo al bloque del programa deseado.

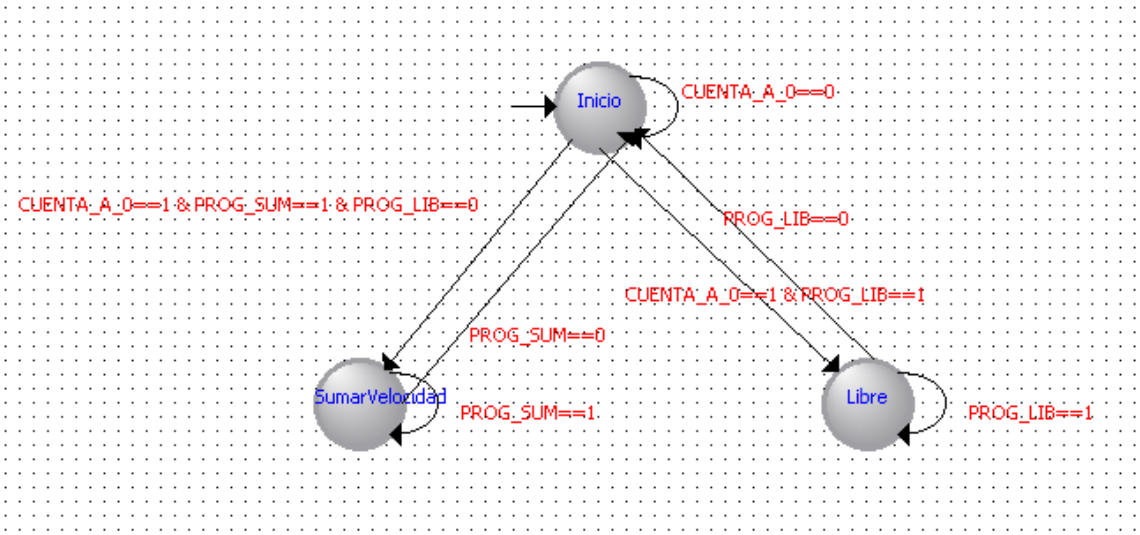

Figura 3. 15 Máquina de Estados Control Programa

En la etapa inicial de cada programa (programa LIBRE o programa SUMAR) se mantiene a 0 la señal SUMA_RESTA, con esto se consigue asegurar que en el momento en el que se realiza la parada de emergencia esta señal va a estar a 0 (preparada para restar).

Se comenzará explicando la Parada de emergencia, para después explicar los propios programas (Programa LIBRE y Programa SUMAR).

3.2.2.3.1 Parada de Emergencia

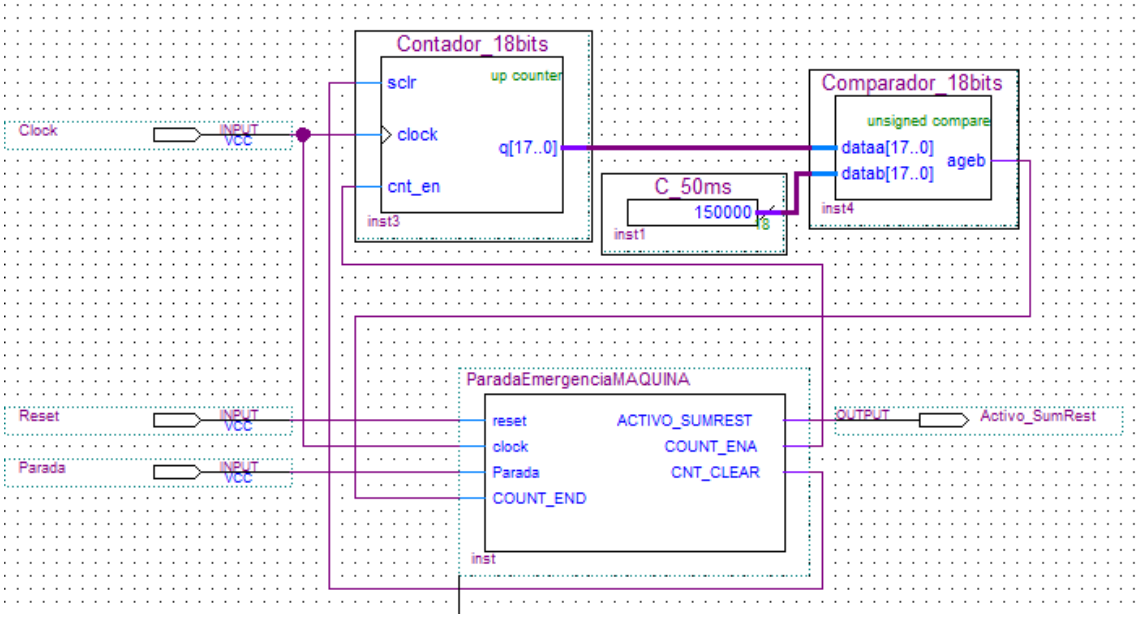

Figura 3. 16 Esquemático Parada de Emergencia

La figura 3.16 muestra el esquema interno del bloque Parada de Emergencia. Mientras esté activa la señal de parada, cada 50 ms se resta 0.1 km/h, lo que equivale a restar 2 km/h por cada segundo.

3.2.2.3.2 Programa LIBRE

En el momento en el que se entra en el programa LIBRE, el usuario puede subir o bajar la velocidad del motor a su placer mediante los pulsadores PulsUp y PulsDown.

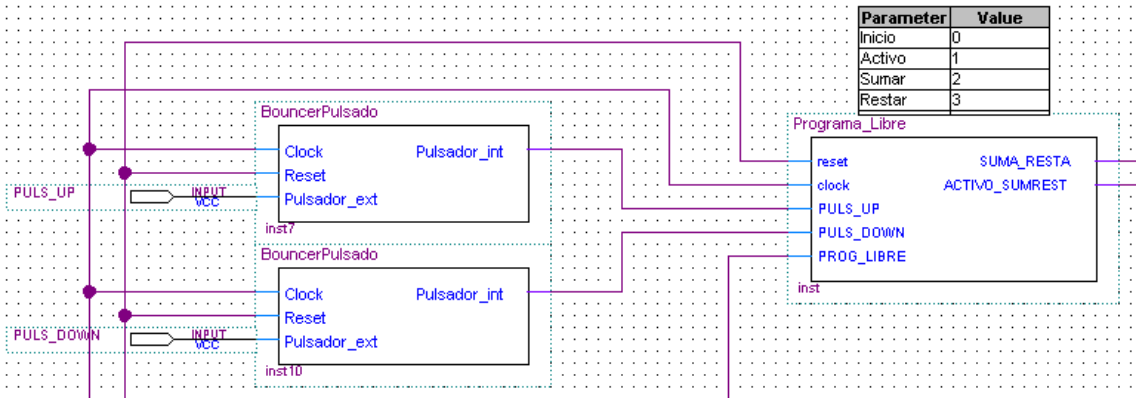

Figura 3. 17 Esquemáticos Programa LIBRE

En la figura 3.17 se puede observar que, además de la señal de activación del programa (PROG_LIBRE), al bloque del Programa LIBRE también le llegan las dos señales para subir y bajar velocidad que provienen de los Bouncers (PULS_UP y PULS_DOWN).

La máquina de estados que controla el programa se puede ver en la figura 3.18.

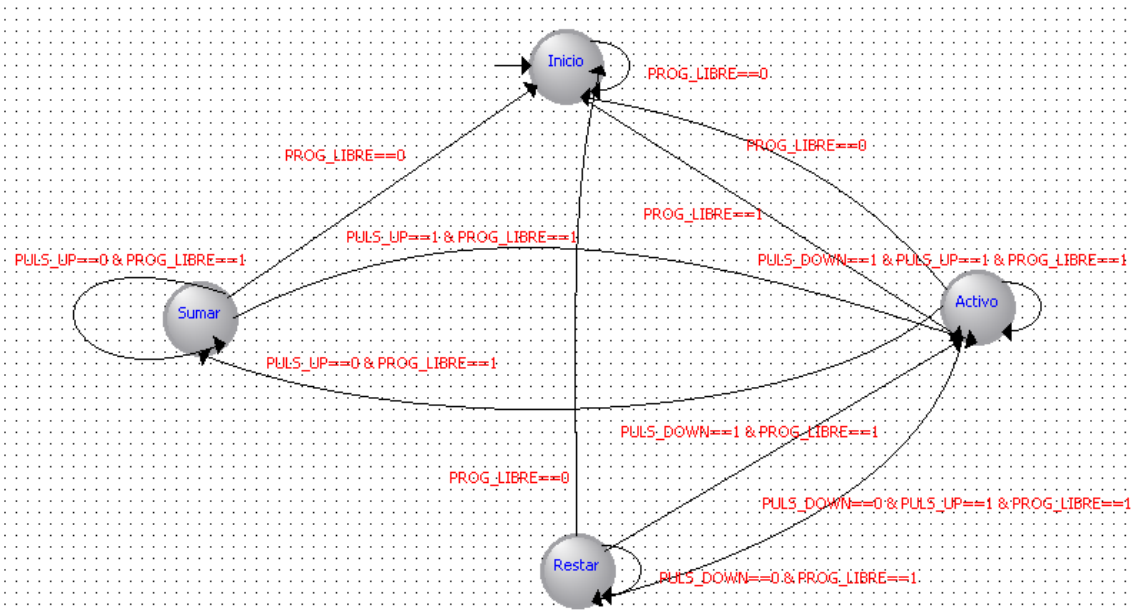

Figura 3. 18 Máquina de Estados Programa LIBRE

Para que el programa esté en funcionamiento, tiene que estar activa la señal de activación del programa en todo momento (señal PROG_LIBRE).

Para subir o bajar velocidad, simplemente se ha que pulsar el botón de subir o bajar respectivamente. En el momento que llega la orden de subir velocidad, el bloque habilita tanto la señal SUMA_RESTA (preparado para sumar) como la señal ACTIVO_SUMARESTA (activación de la operación). La operación sumar corresponde al estado “Sumar” de la máquina de estados presente en la figura 3.18. En el caso de la

resta, la señal SUMA_RESTA se dejaría desactivada. La operación restar corresponde al estado “Restar” de la máquina de estados presente en la figura 3.18.

Frente a la necesidad de, por ejemplo, subir la velocidad progresivamente si el botón para subirla se mantiene pulsado, se ha diseñado el bloque Bouncer Pulsado.

Bouncer Pulsado

Se va a proceder a la explicación del funcionamiento del programa cuando se presiona el pulsador que se utiliza para subir la velocidad del motor.

El bloque Bouncer Pulsado es el encargado de detectar el tipo de activación del pulsador. En el caso de una pulsación normal, este bouncer actúa igual que el Bouncer Pulso explicado anteriormente. Cuando el pulsador se activa, el bloque Bouncer Pulsado se espera 20 ms para comprobar que es una pulsación real y, si después de estos 20 ms la señal sigue estando a nivel bajo (pulsador pulsado), la señal interna se activa de modo que únicamente se recibe un pulso.

A partir de este momento el funcionamiento del Bouncer Pulsado es diferente al Bouncer Pulso. Si se mantiene presionado el pulsador después de los primeros 20 ms, en el momento en el que pasa 1 segundo, se comienza a generar un pulso interno cada 100 ms, siempre y cuando el pulsador se mantenga presionado. Este funcionamiento equivale a sumar 0.1 km/h cada 100 ms, lo que aumentaría un total de 1 km/h cada segundo.

En las figuras 3.19 y 3.20 se pueden ver las conexiones de los contadores (Contador 1 s y Contador 20ms) con la máquina de estados (esquemático del bloque Bouncer Pulsado) y la máquina de estados que gobierna el bloque respectivamente.

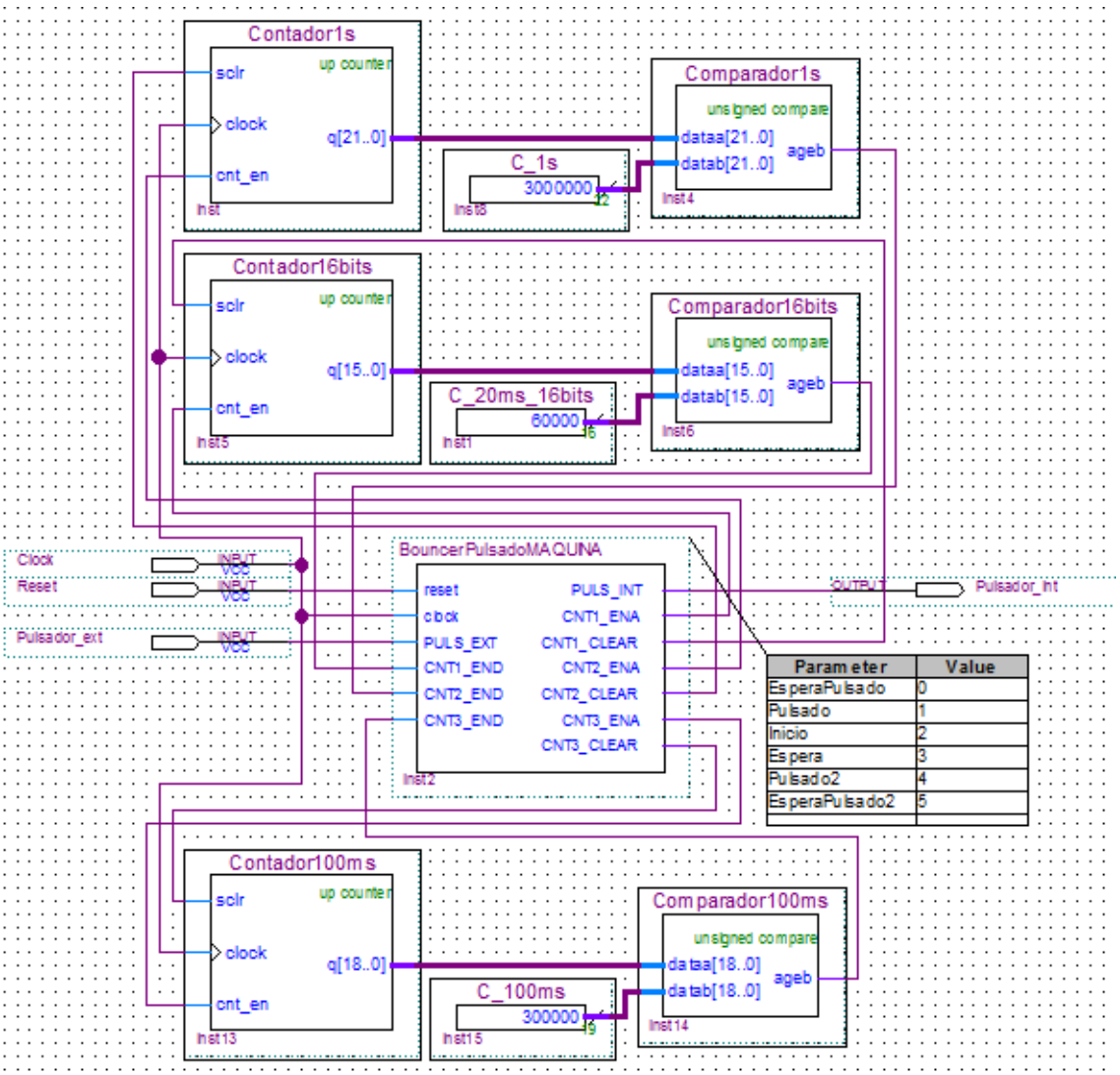

Figura 3. 19 Esquemático Bloque Bouncer Pulsado

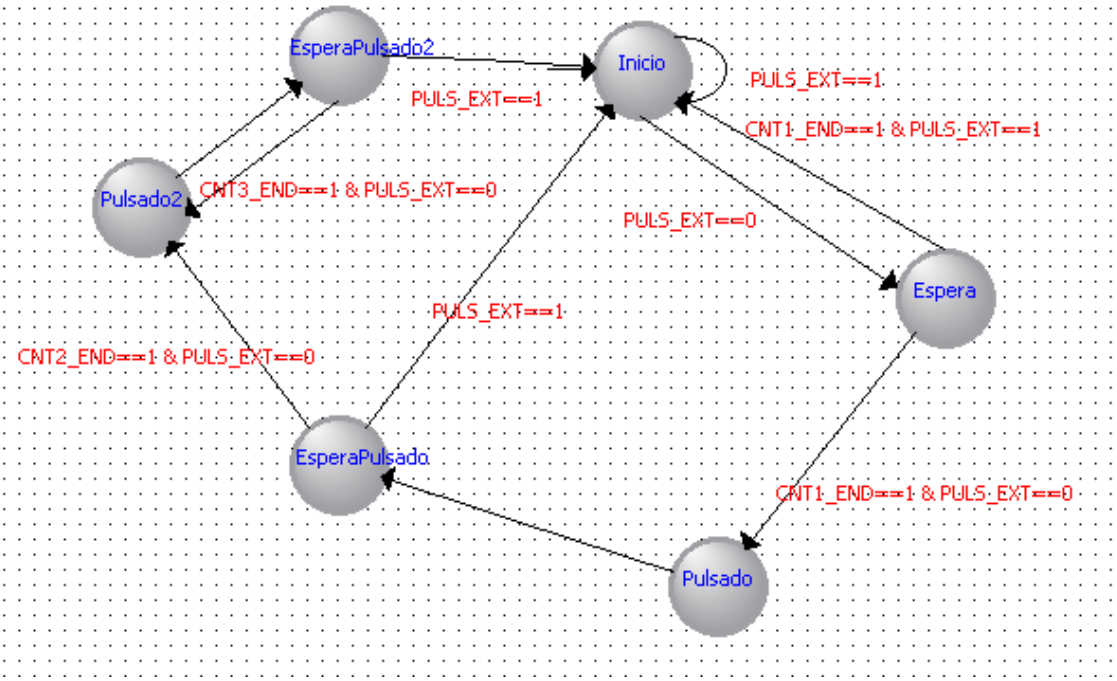

Figura 3. 20 Máquina de Estados Bloque Bouncer Pulsado

3.2.2.3.3 Programa SUMAR

La idea de realizar este programa se ha surgido de la ahora famosa Prueba de Esfuerzo que se realizan tantos deportistas. Esta prueba consiste en un ejercicio de duración variable, el cual comienza a una velocidad muy suave (6 km/h) para calentar. A partir de ese momento, cada minuto se aumenta la velocidad 1 km/h hasta que el deportista llega a su límite, momento en el que avisa al médico que le practica la prueba para que pare la cinta.

Mediante esta prueba se descartan posibles anomalías cardíacas y se obtienen muchos datos útiles para estos deportistas.

En la figura 3.21 se puede observar el formato del bloque.

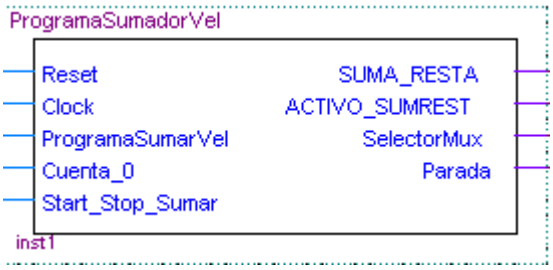

Figura 3. 21 Bloque del Programa SUMAR

El bloque dispone de las siguientes entradas:

- ProgramaSumarVel: señal que se activa cuando el programa SUMAR está seleccionado y listo para usarse (ya se ha realizado la parada de emergencia si se viene de otro programa).

- Start_Stop_Sumar: pulsador mediante el cual, una vez el sistema se encuentra en el programa SUMAR, se puede iniciar o parar la Prueba de Esfuerzo. La señal proviene de un Bouncer Pulso.
- Cuenta_0: señal que indica que el motor está parado. En este bloque hace falta esta señal puesto que el programa SUMAR se puede poner en funcionamiento y parar sin cambiar de programa. Por tanto, si se le da al botón Start_Stop_Sumar cuando está en funcionamiento, para volverse a iniciar se ha de asegurar primero que la cinta está parada.

Mientras que las salidas de las que dispone son las siguientes:

- SUMA_RESTA: utilizada posteriormente en el bloque sumador/restador para elegir si se desea sumar o restar velocidad.
- ACTIVO_SUMARESTA: utilizada posteriormente en el bloque sumador/restador para activar la suma o la resta.
- SelectorMux: activando esta señal en el momento de la suma de velocidad, se suma de 1 km/h en 1 km/h, en vez de 0.1 km/h en 0.1 km/h.
- Parada: cuando se está en el programa SUMAR esta señal se puede activar en dos ocasiones, cuando se le da al botón Start_Stop_Sumar mientras este programa está en funcionamiento o cuando se termina el programa. Esta señal va directa al bloque de Parada de Emergencia.

En la figura 3.22 se puede observar el esquema interno del programa SUMAR.

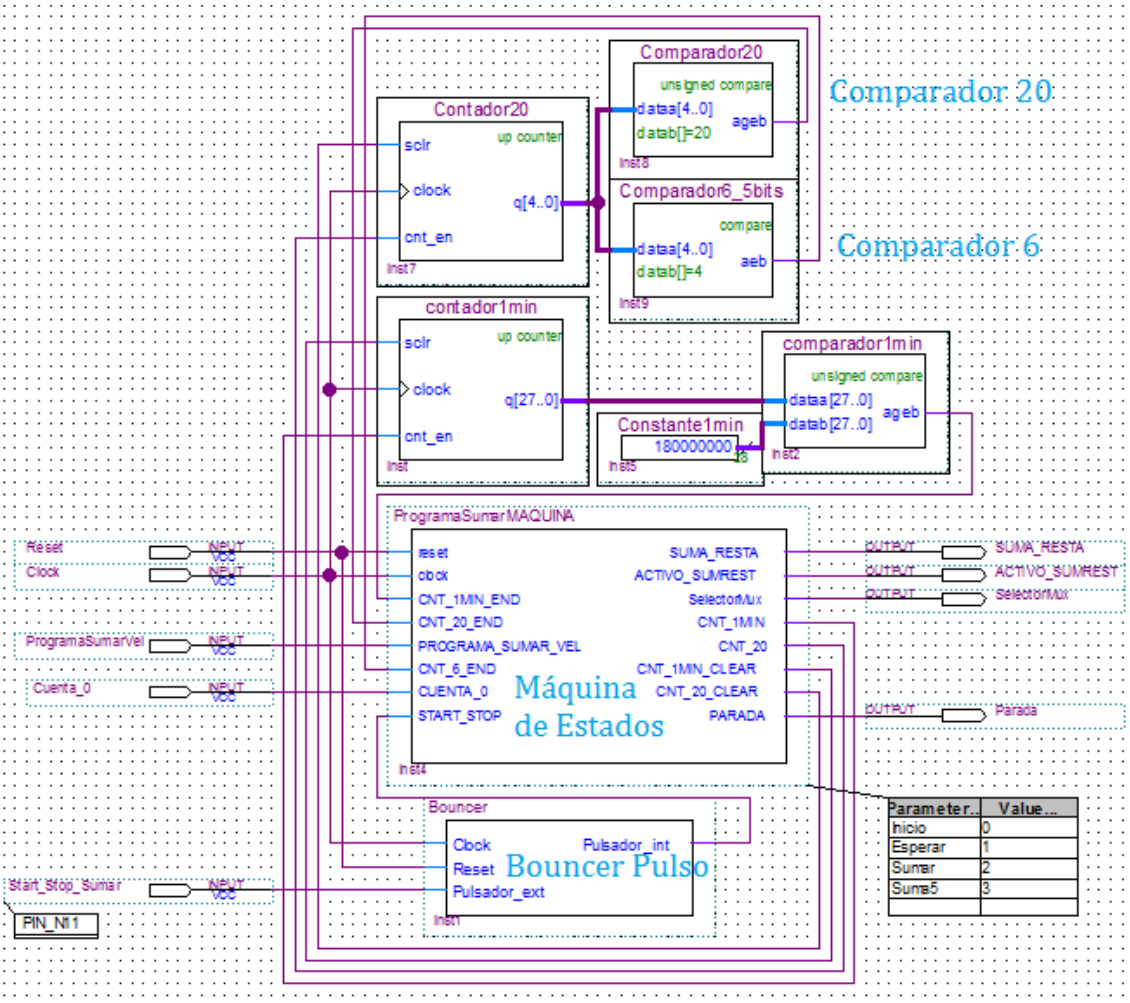

Figura 3. 22 Esquema Interno Bloque SUMAR

El funcionamiento del programa SUMAR es el siguiente:

Una vez ha llegado el bit de activación del programa, el sistema se queda esperando a que el motor esté parado y que se pulse el botón Start_Stop_Sumar. Cuando se cumplen estas dos condiciones el programa comienza. Durante todas las sumas, el multiplexor está programado para realizar sumas de 1 km/h.

Se comienza sumando 6 km/h, y, cada minuto que pasa se suma 1 km/h más. Se ha supuesto que la velocidad máxima de esta prueba es de 20 km/h, lo que equivale a realizar 1km en 3 minutos. Si se alcanza la velocidad máxima, cuando se lleve 1 minuto en esta velocidad, el programa se detendrá automáticamente realizando la parada de emergencia (2 km/h por cada segundo).

Si se pulsa el botón Start_Stop_Sumar mientras está en funcionamiento la cinta, el motor se detiene automáticamente mediante la parada de emergencia. En el momento en el que el motor se para, si se pulsa de nuevo el motor, el programa comienza de nuevo desde el principio.

En la figura 3.23 se puede ver la máquina de estados que gobierna el programa SUMAR.

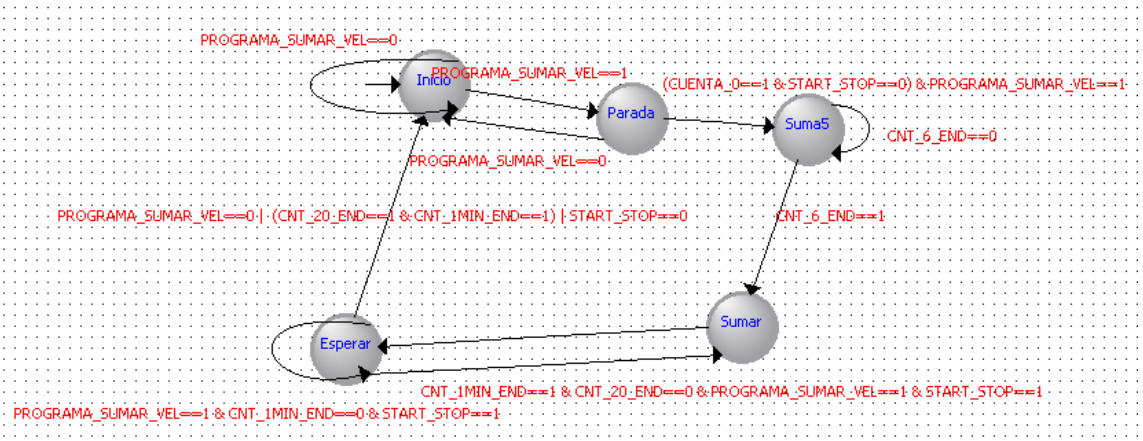

Figura 3. 23 Máquina de Estados Programa SUMAR

3.2.2.4 Conversor Unidades

En la figura 3.24 se puede observar el aspecto del bloque Conversor de Unidades

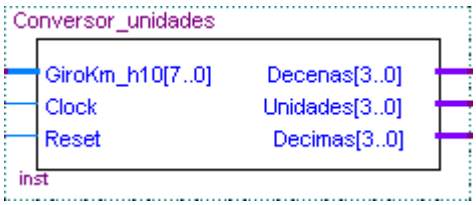

Figura 3. 24 Bloque Conversor de Unidades

La función de este bloque es obtener cada dígito del número en decimal de la velocidad de la cinta en kilómetros por hora por separado, a partir de la velocidad de la cinta en binario de 8 bits.

En la figura 3.25 se muestra el esquema interno del bloque.

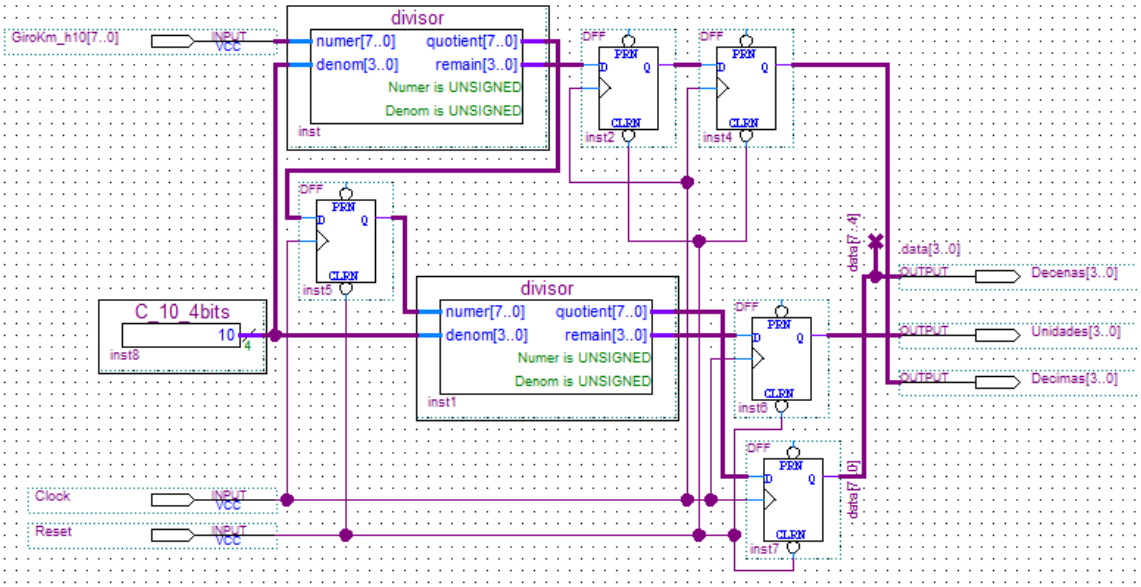

Figura 3. 25 Esquemático Bloque Conversor Unidades

Para conseguir separar cada dígito del número decimal, se parte del valor de la velocidad en km/h multiplicado por 10 en binario, y se va dividiendo sucesivamente entre 10.

Así el resultado de la primera división es el valor de las décimas de km/h. El resultado de la primera de división se divide de nuevo entre 10, para obtener, por un lado, las unidades de km/h (resto de la operación), y, por otro, las decenas de km/h (resultado de la operación).

Los biestables se utilizan para que las señales salgan del bloque en el mismo instante (tarden los mismos ciclos en salir).

Una vez se tienen los dígitos del número decimal de la velocidad de la cinta por separado, se envían al bloque que controla la memoria RAM utilizada para el LCD.

3.2.3 Bloque Control LCD

La función principal del bloque de Control del LCD es la escritura en una memoria RAM tanto de los valores de velocidad de la cinta, como del programa de ejercicio actual. Posteriormente, mediante el bloque control_motor_completo, se lee esta memoria y se envían las señales necesarias al procesador del LCD para mostrar los datos.

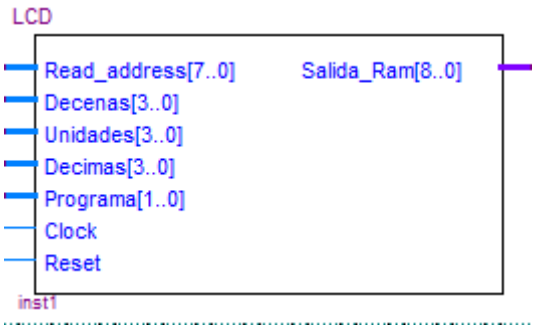

Figura 3. 26 Bloque LCD

En la figura 3.26 se muestran las entradas y salidas del bloque LCD.

Las entradas son las siguientes:

- Read_address[7..0]: entrada utilizada por el bloque control_motor_completo para elegir la dirección de memoria de la que quiere leer el dato (figura 3.1). Esta señal es una salida del bloque control_motor_completo.
- Decenas[3..0]: valor del dígito de las decenas de velocidad (velocidad en km/h).
- Unidades[3..0]: valor del dígito de las unidades de velocidad (velocidad en km/h.
- Decimas[3..0]: valor del dígito de las décimas de velocidad (velocidad en km/h)
- Programa[1..0]: señal que indica en el número de programa en el que se encuentra el sistema:
    - 0: No hay programa
    - 1: Programa LIBRE

- 2: Programa SUMAR

En cuanto a las salidas, se tiene la siguiente:

- Salida_Ram[8..0]: valor del dato almacenado en la dirección de memoria indicada mediante la señal Read_address[7..0].

Como ya se ha comentado, el bloque control_motor_completo realiza la lectura de la memoria RAM. Este bloque está preparado para hacer lecturas cíclicas sobre unas direcciones de memoria específicas. Para elegir estas direcciones se utiliza la señal de Display[2..0] (salida del bloque TFG_TOP y entrada del bloque control_motor_completo):

- Display[2..0]=0: se lee desde la dirección 0x05 hasta la dirección 0x26 (en hexadecimal).
- Display[2..0]=1: se lee desde la dirección 0x27 hasta la dirección 0x48.
- Display[2..0]=2: se lee desde la dirección 0x49 hasta la 0x6A.

Las primeras 4 direcciones de memoria se utilizan para la inicialización del LCD.

Cada lectura cíclica lee un total de 34 direcciones de memoria, en las que:

- Las 16 primeras direcciones contienen los valores de los 16 dígitos de la primera línea
- La dirección 17 es un salto de línea
- Las 16 siguientes contienen los valores de los 16 dígitos de la segunda línea
- La dirección 34 contiene la orden “return home”, acción mediante la cual se vuelve a la primera dirección.

Por tanto, mediante la señal Display[2..0] se puede elegir la pantalla que se quiere mostrar.

En el presente proyecto se va a optar por utilizar un total de 3 pantallas, la primera para indicar que no hay programa, y las dos siguientes para los dos programas disponibles. Como ya se explicó al inicio, la programación se ha realizado de modo que el número de programa coincida con el número de pantalla. Por lo tanto, se conecta la señal Programa[1..0] añadiéndole un 0 delante, a la señal Display[2..0] para mandarla al bloque control_motor_completo.

Se va a proceder a continuación a explicar qué información se incluye en cada pantalla, así como los esquemáticos utilizados para hacer la escritura de los valores de velocidad. Se explica en primer lugar la memoria RAM, para a continuación explicar el proceso de elección de la dirección de escritura y la elección y codificación del dato a escribir.

En la figura 3.27 se muestra el esquema interno del bloque LCD.

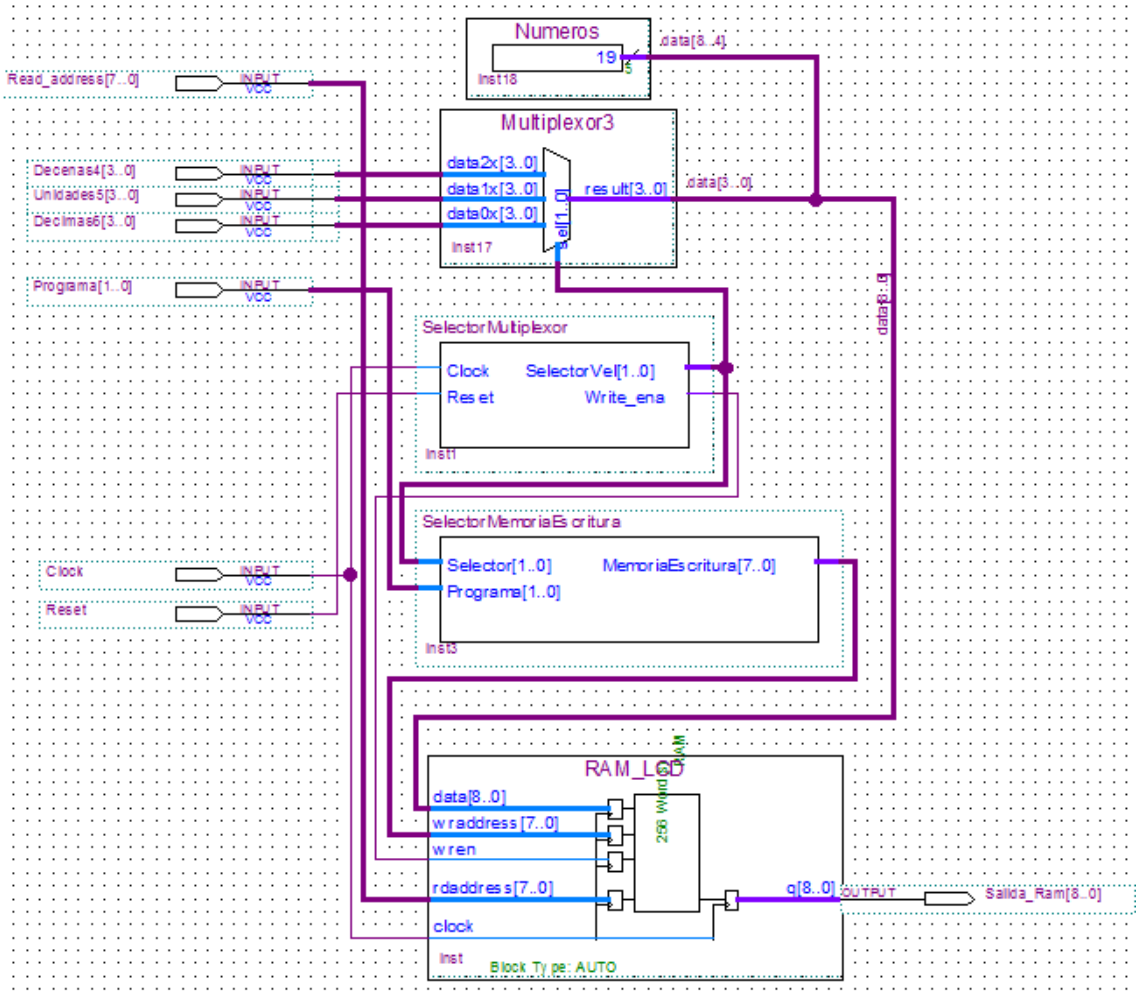

Figura 3. 27 Esquemáticos Bloque LCD

3.2.3.1 Memoria RAM

Como la señal Read_address[7..0] que proviene del bloque control_motor_completo tiene 8 bits, se ha elegido una memoria RAM con 256 direcciones, puesto que 2^8=256.

Por otra parte, se ha elegido una memoria RAM de doble puerto, uno de escritura y otro de lectura. Esto es necesario ya que, al mismo tiempo, el bloque LCD escribe los valores deseados en la RAM y el bloque control_motor_completo realiza la lectura.

En cuanto al tamaño de palabra, se ha escogido un tamaño de 9 bits, debido a que se tiene que guardar el dato o instrucción (8 bits), precedido de un bit que indica si es un dato (bit=1) o una instrucción (bit=0). En este trabajo las únicas instrucciones que se utilizan son para inicializar la tarjeta y “salto de línea” y “return home”.

En la figura 3.28 se muestra la codificación de cada uno de los diferentes caracteres.

Figura 3. 28 Codificación Caracteres LCD

En primer lugar se inicializa la memoria con los datos que debe mostrar cada pantalla.

A continuación se muestran los valores de cada dirección de memoria al inicializarla:

05: 101010000; -- P ####### PRIMERA LINEA PRIMERA PANTALLA

06: 101010010; -- R

07: 101001111; -- O

08: 101000111; -- G

09: 101010010; -- R

0A: 101000001; -- A

0B: 101001101; -- M

0C: 101000001; -- A

0D: 100111010; -- dos puntos

0E: 100100000; -- espacio

0F: 101001110; -- N

10: 101001111; -- O

11: 100100000; -- espacio

12: 101001000; -- H

13: 101000001; -- A

14: 101011001; -- Y

15: 011000000; -- salto de línea

16: 101010110; -- V ####### SEGUNDA LINEA PRIMERA PANTALLA

17: 101000101; -- E

18: 101001100; -- L

19: 101001111; -- O

1A: 101000011; -- C

1B: 100101110; -- punto

1C: 100100000; -- espacio

1D: 100110000; -- 0 (decenas velocidad)

1E: 100110000; -- 0 (unidades velocidad)

1F: 100101100; -- coma

20: 100110000; -- 0 (décimas velocidad)

21: 100100000; -- espacio

22: 101101011; -- K

23: 101101101; -- m

24: 100101111; -- barra

25: 101101000; -- h

26: 000000011; -- return home ####### FIN PRIMERA PANTALLA

27: 101010000; -- P ####### PRIMERA LINEA SEGUNDA PANTALLA

28: 101010010; -- R

29: 101001111; -- O

2A: 101000111; -- G

2B: 101010010; -- R

2C: 101000001; -- A

2D: 101001101; -- M

2E: 101000001; -- A

2F: 100111010; -- dos puntos

30: 100100000; -- espacio

31: 100100000; -- espacio

32: 101001100; -- L

33: 101001001; -- I

34: 101000010; -- B

35: 101010010; -- R

36: 101000101; -- E

37: 011000000; -- salto de línea

38: 101010110; -- V ####### SEGUNDA LINEA SEGUNDA PANTALLA

39: 101000101; -- E

3A: 101001100; -- L

3B: 101001111; -- O

3C: 101000011; -- C

3D: 100101110; -- punto

3E: 100100000; -- espacio

3F: 100110000; -- 0 (decenas velocidad)

40: 100110000; -- 0 (unidades velocidad)

41: 100101100; -- coma

42: 100110000; -- 0 (décimas velocidad)

43: 100100000; -- espacio

44: 101101011; -- K

45: 101101101; -- m

46: 100101111; -- barra

47: 101101000; -- h

48: 000000011; -- return home ####### FIN SEGUNDA PANTALLA

49: 101010000; -- P ####### PRIMERA LINEA TERCERA PANTALLA

4A: 101010010; -- R

4B: 101001111; -- O

4C: 101000111; -- G

4D: 101010010; -- R

4E: 101000001; -- A

4F: 101001101; -- M

50: 101000001; -- A

51: 100111010; -- dos puntos

52: 100100000; -- espacio

53: 100100000; -- espacio

54: 101010011; -- S

55: 101010101; -- U

56: 101001101; -- M

57: 101000001; -- A

58: 101010010; -- R

59: 011000000; -- salto de línea

5A: 101010110; -- V ####### SEGUNDA LINEA TERCERA PANTALLA

5B: 101000101; -- E

5C: 101001100; -- L

5D: 101001111; -- O

5E: 101000011; -- C

5F: 100101110; -- punto

60: 100100000; -- espacio

61: 100110000; -- 0 (decenas velocidad)

62: 100110000; -- 0 (unidades velocidad)

63: 100101100; -- coma

64: 100110000; -- 0 (décimas velocidad)

65: 100100000; -- espacio

66: 101101011; -- K

67: 101101101; -- m

68: 100101111; -- barra

69: 101101000; -- h

6A: 000000011; -- return home ####### FIN TERCERA PANTALLA

Como se puede observar, se ha inicializado la velocidad en cada pantalla a 0 km/h. Por lo tanto lo único que faltaría sería ir actualizando la velocidad actual del motor en la dirección de memoria adecuada, dependiendo del programa (y por tanto de la pantalla) en que se encuentre el sistema.

En las figuras 3.29, 3.30 y 3.31 se puede observar el aspecto del LCD tras la inicialización de cada una de las 3 pantallas.

Figura 3. 29 LCD Sin Programa

Figura 3. 30 LCD Programa LIBRE

Figura 3. 31 LCD Programa SUMAR

3.2.3.2 Elección Dirección de Escritura y Dato a Escribir

Se ha optado por hacer una escritura de la velocidad en la memoria RAM cada 10 ms, un tiempo suficientemente pequeño con respecto al cambio de velocidad. Se opta por escribir cada cierto tiempo para no saturar a la FPGA.

Por tanto, cada 10 ms, se ha de realizar una escritura de los 3 dígitos de velocidad (decenas, unidades y décimas). Para realizar esta esta escritura, se tiene que elegir el dato a escribir y, al mismo tiempo, la dirección de escritura.

Como puede observarse en la figura 3.27, para cambiar entre los diferentes dígitos se usa un multiplexor, el cual deja pasar las décimas si recibe un 0, las unidades si recibe un 1 y las decenas si recibe un 2.

En la figura 3.32 se muestra el esquema del bloque utilizado para generar la señal del selector y la señal de activación de la escritura. Ambas señales se generan en los mismos ciclos, pero la señal del selector no se activa durante los 10 ms de espera.

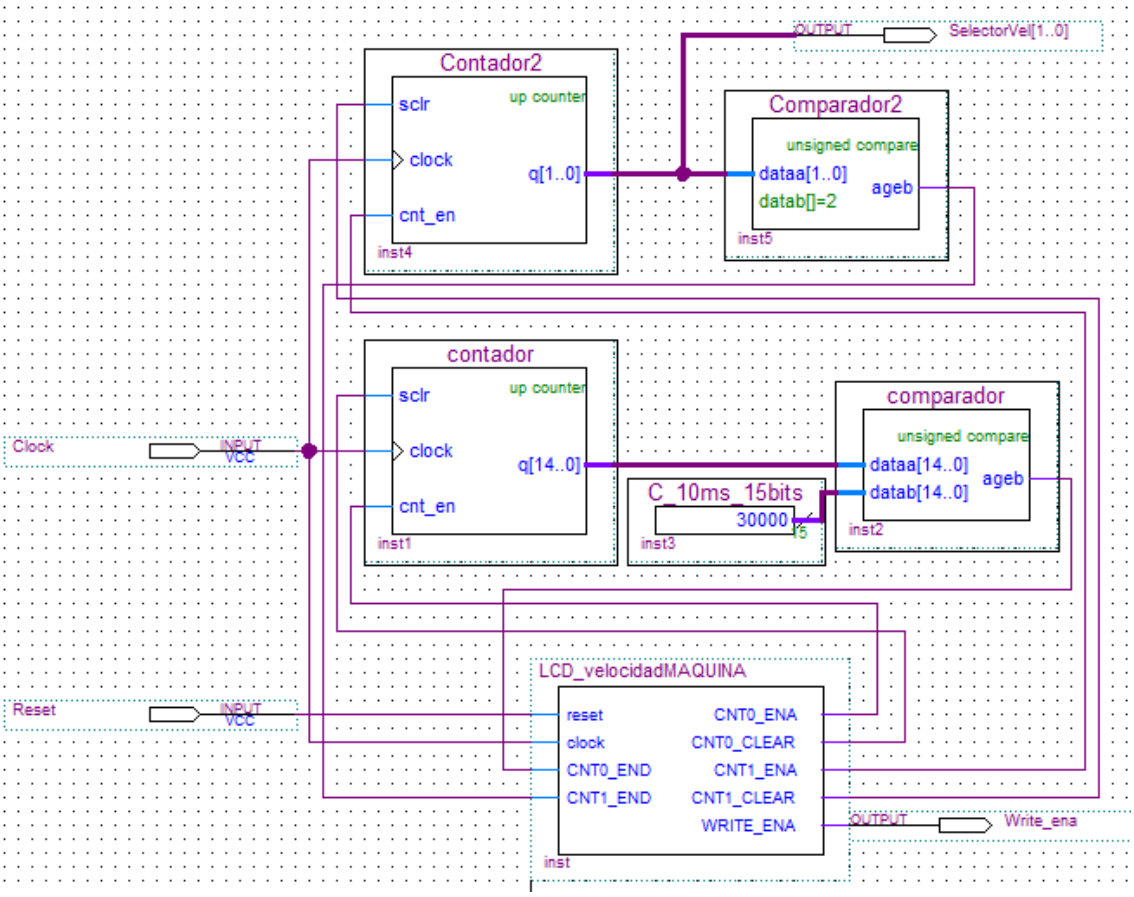

Figura 3. 32 Esquemático Bloque Selector Multiplexor

Una vez elegido el dígito que se quiere escribir, se ha de seleccionar la dirección de memoria en la cual se ha de escribir. Para esto se usa el bloque SelectorMemoriaEscritura mostrado en la figura 3.27.

En la figura 3.33 se muestra la vista interna de este bloque.

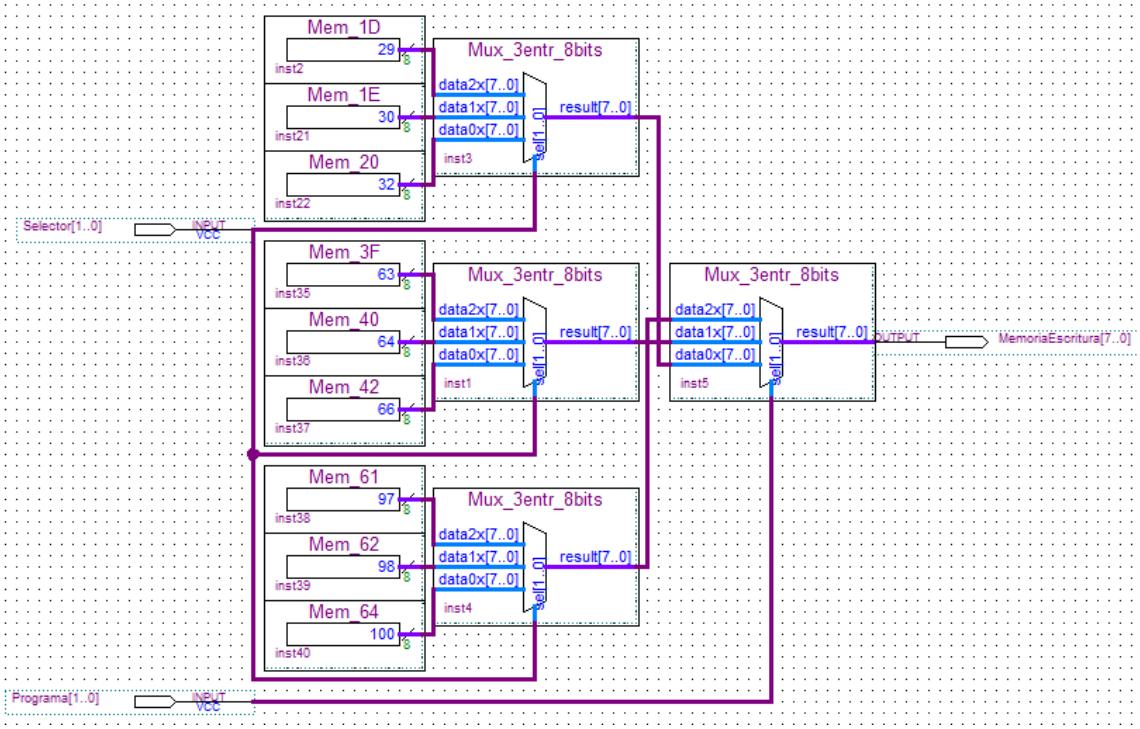

Figura 3. 33 Esquemático Bloque SelectorMemoriaEscritura

Se puede observar que para elegir la dirección de memoria de escritura se utiliza la misma señal para los primeros multiplexores que se utiliza para seleccionar el dígito, que es la señal que sale del bloque SelectorMultiplexor (Señal Selector[1..0]).

Por ejemplo, para el caso de las décimas, la señal Selector[1..0] vale 0. Esto quiere decir que se eligen para escribir las memorias codificadas en hexadecimal 20 (32 en decimal), 42 (66 en decimal) y 64 (100 en decimal). Por último, para elegir entre estas 3 memorias se utiliza el multiplexor mostrado en la parte derecha de la figura 3.33 y la señal Programa[1..0], que puede adoptar los siguientes valores:

- 0: No hay programa (primera pantalla)
- 1: Programa LIBRE (segunda pantalla)
- 2: Programa SUMAR (tercera pantalla)

Por lo tanto, si estuviera en el programa LIBRE, esta señal valdría 1, y la señal MemoriaEscritura, contendría la memoria 42 (66 en decimal).

3.2.3.3 Codificación del Dato a Escribir

Para la escritura del dato, no es suficiente con enviar el dato en 4 bits. Como se ha comentado anteriormente se ha de enviar un dato compuesto por 8 bits precedido de un 1 (que indica que es un dato).

En la figura 3.28 se muestra la codificación de cada número. Esta codificación se puede dividir en dos partes:

- 4 bits más significativos: constante de valor 0011 en binario.
- 4 bits de menor peso: Codificación en BCD del número.

Si delante de cada dato se ha de añadir un 1 (indicando que es un dato), el dato a enviar a la memoria RAM se compondrá de las siguientes partes:

- Constante de 5 bits de valor 10011 (19 en decimal)
- Valor en BCD de cada número

En la figura 3.27 se observa que al valor codificado en BCD que sale del multiplexor se le añade en los bits más significativos la constante 19 en decimal (10011 en binario).

De esta forma se ha conseguido una fácil transformación de la codificación en BDC a la codificación del LCD.

4 Trabajo en el Laboratorio

Una vez simulado todo con el software ModelSim se dispuso a hacer la prueba en la FPGA y sobre el motor real.

Para realizar las pruebas se dispone de la placa de potencia descrita en el apartado 2.4. Como se ha comentado, esta placa dispone de los transistores del puente en H necesarios para variar la tensión del motor, así como de la Tarjeta DE0-Nano, la cual incluye la FPGA utilizada durante el trabajo. La placa también incluye la pantalla de cristal líquido necesaria para la visualización descrita en el apartado 3.2.3. Por último, lleva conectada a la tarjeta DE0-Nano la botonera utilizada para el control del proceso.

En primer lugar se realizó la asignación de pines. Asignando un pulsador físico diferente a cada señal de los pulsadores. La asignación de los pulsadores se puede observar en la figura 4.1.

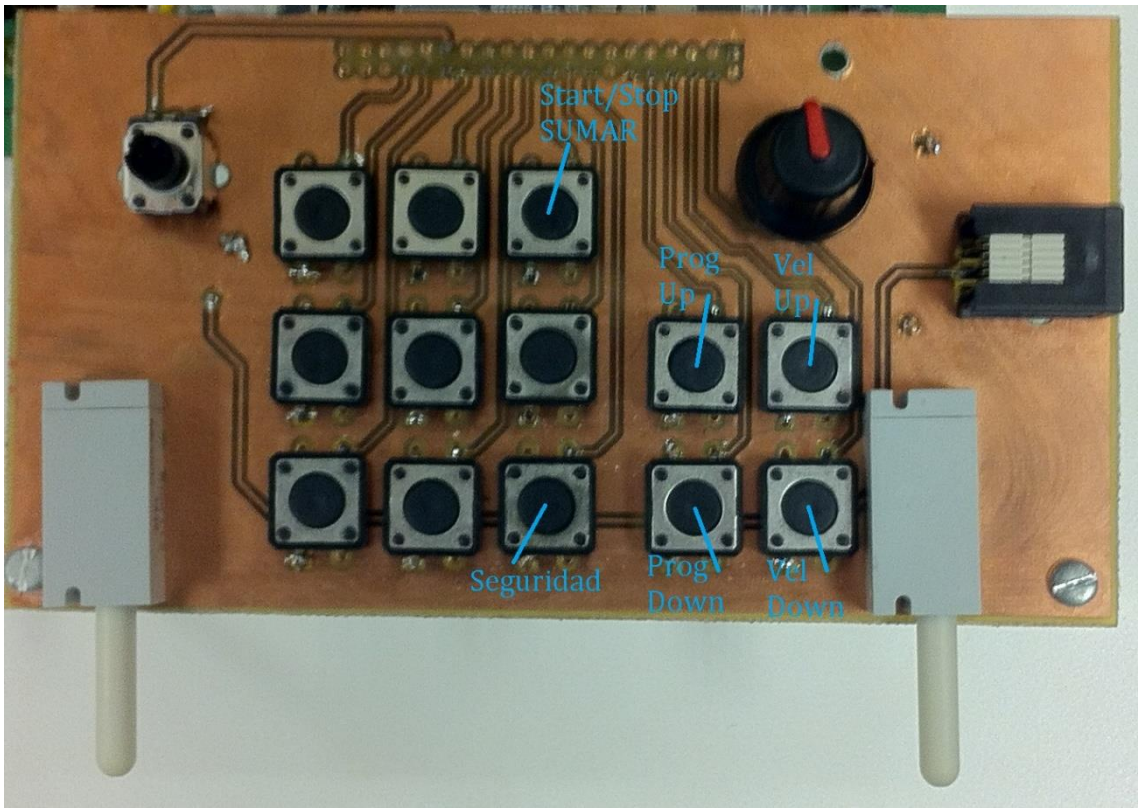

Figura 4. 1 Asignación Botones

En un principio se realizó la una prueba sin dejar pasar las señales de los transistores del puente en H. Se tomó esta medida por si había algún error en el diseño, ya que, como se comentó en el apartado 2.2, un error en estas señales puede provocar un cortocircuito en una de las líneas.

4.1 Problemas con los Pulsadores

En la primera prueba realizada hubo menos problemas de los esperados. El principal problema fue que cuando el sistema se encontraba en el Programa LIBRE, alguna de las veces en las que se pulsaba el botón de subir o el botón de bajar velocidad, saltaba al estado en el que no hay programa. Pronto se descubrió que esto era debido a que el pulsador de seguridad (botón que simula la llave de seguridad), no disponía del sistema Bouncer explicado en el punto 3.2.1.2. Al pulsar uno de los botones de subir o bajar velocidad, la tensión en el pulsador Seguridad disminuía tanto que el sistema lo entendía como que había sido pulsado (simulando la llave de seguridad desconectada).

Para solucionar este problema se diseñó el software Bouncer Retardo, el cual una vez recibe un 0 por parte del pulsador, espera 20 ms para comprobar que ha sido una pulsación real, si es así, cambia el valor del pulsador interno a 0. En ese momento el sistema queda a la espera de recibir un 1, para realizar la comprobación de nuevo. Si se ha dejado de pulsar realmente, después de los 20 ms cambia el valor del pulsador interno a 1.

En el momento en el que todos los errores estaban solucionados, se probó el sistema completo, conectando las señales del puente en H. Al principio se tuvo algunos problemas con los pulsadores, pero pronto se descubrió que era problema de la tarjeta, ya que al cambiar de puesto todo funcionó correctamente.

Por último, se vio la posibilidad de hacer una mejora. En un principio los pulsadores para subir y bajar la velocidad del motor cuando el sistema se encontraba en el programa LIBRE disponían del mismo Bouncer que los pulsadores para la elección del programa. Por tanto, cada vez que se pulsaba, por ejemplo, el pulsador para subir la velocidad, solo se aumentaba en 0.1 km/h.

En ese momento fue cuando se diseñó el Bouncer Pulsado (explicado en el punto 3.2.2.3.2), software mediante el cual si se mantiene pulsado, por ejemplo, el pulsador para subir la velocidad durante más de un segundo, esta velocidad va aumentando progresivamente (0.1 km/h cada 100 ms). El pulsador para disminuir la velocidad tiene el mismo comportamiento, pero disminuyendo la velocidad.

5 Manual de Usuario

En la figura 5.1 se muestra el panel de mandos de la cinta de correr. Se observa que se utilizan un total de 6 pulsadores.

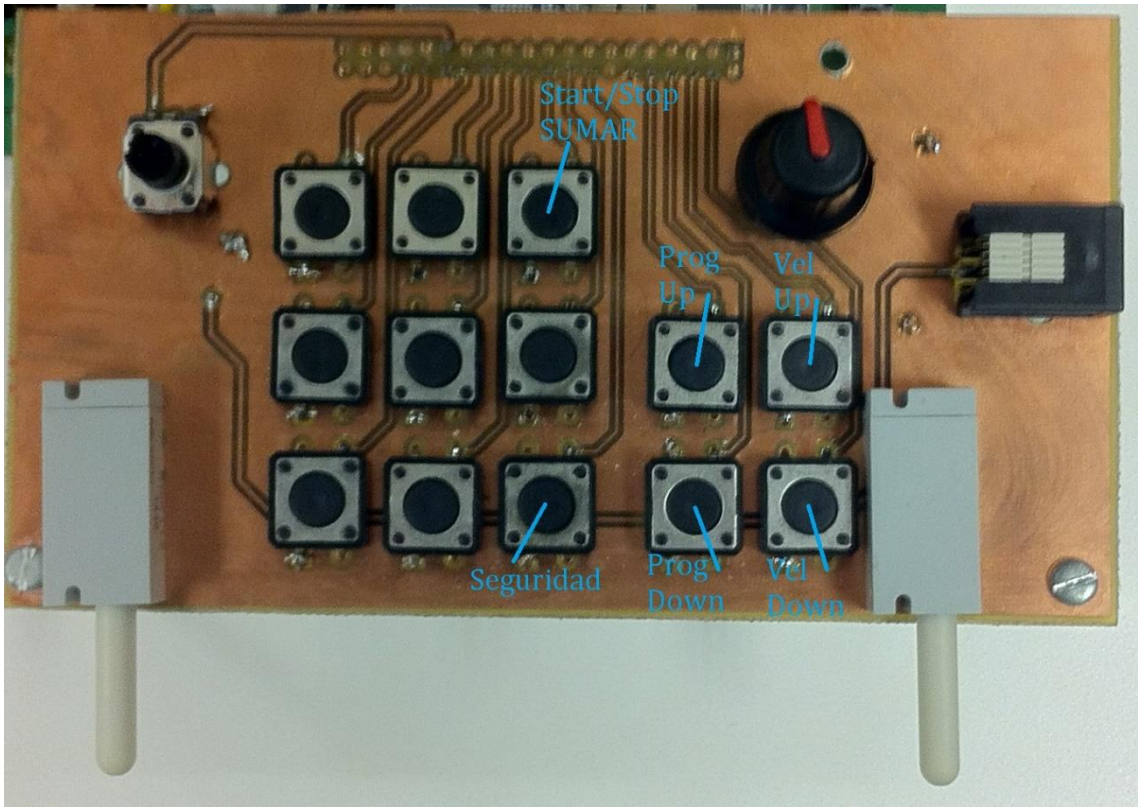

Figura 5. 1 Pulsadores Control Cinta

El control del motor de la cinta de correr comienza por defecto sin ningún programa de la cinta en funcionamiento. El pulsador de seguridad debe estar siempre sin pulsar. Este pulsador simula una llave de seguridad y el pulsador sin pulsar equivale a la llave enchufada. Siempre que este pulsador se pulse, el sistema volverá al estado en el que la cinta no está en ningún programa, y por lo tanto, la cinta se parará.

El sistema puede encontrarse en los siguientes estados:

- 0: No hay programa
- 1: Programa LIBRE
- 2: Prueba de Esfuerzo (Programa SUMAR)

En la figura 5.2 se muestra la imagen del LCD en el momento del inicio.

Figura 5. 2 LCD Inicio

Para moverse entre los distintos programas de la cinta se han de utilizar los pulsadores para aumentar o disminuir programa (ProgUp y ProgDown respectivamente). En el estado inicial, si se pulsa el botón para aumentar programa se pasa al programa LIBRE, y si se pulsa el botón para disminuir programa se pasa a la prueba de esfuerzo. En cualquier momento se puede realizar el cambio entre programas, si la cinta se encontrase en funcionamiento, se detendría, y en el momento en el que se parase, se daría comienzo el programa elegido.

5.1 Programa LIBRE

En la figura 5.3 se puede observar el estado del LCD cuando el sistema se encuentra en el programa LIBRE.

Figura 5. 3 Programa LIBRE Inicio

Este programa permite variar la velocidad de la cinta mediante los pulsadores diseñados para ello. Mediante el pulsador para subir velocidad (VeUp), se aumenta la velocidad de la cinta y mediante el pulsador para bajar velocidad (VelDown), se disminuye la velocidad. Si se mantiene presionado uno de los pulsadores, la velocidad aumentará o

disminuirá progresivamente, dependiendo del pulsador presionado. En el caso de presionar los dos, se ha dado preferencia al pulsador de subir velocidad.

En la figura 5.4 se muestra el LCD con el programa LIBRE en funcionamiento.

Figura 5. 4 Programa LIBRE en Funcionamiento

5.2 Prueba de Esfuerzo

En la figura 5.5 se puede observar el estado del LCD cuando el sistema se encuentra en la Prueba de Esfuerzo.

Figura 5. 5 Programa LIBRE Inicio

Para iniciar este programa se debe pulsar el botón Start/Stop Prueba. La Prueba consiste en una progresión que pone a prueba las facultades físicas del usuario. Se comienza a 6 km/h y se va aumentando 1 km/h cada minuto. La prueba termina si el usuario alcanza una velocidad de 20 km/h, velocidad que tendrá que aguantar durante 1 minuto y posteriormente la cinta se para. Se podrá detener la prueba en cualquier momento del ejercicio, ya sea por fatiga del usuario o por cualquier otro motivo, para ello basta con pulsar el botón Start/Stop de nuevo.

En la figura 5.6 se muestra el LCD cuando el usuario se encuentra realizando la prueba.

Figura 5. 6 Programa SUMAR en Funcionamiento

6 Conclusiones

El objetivo de este proyecto es el desarrollo de un sistema que permita controlar la puesta en marcha, apagado, selección de programas y velocidad del motor de una cinta de correr.

Se ha diseñado el control digital síncrono del motor DC presente en una cinta de correr utilizando una FPGA Altera Cyclone IV EP4CE22F17C6N.

La comunicación con el usuario se ha realizado a través de un total de 6 pulsadores y un visualizador de cristal líquido (LCD).

Se ha diseñado un pulsador que hace la función de llave de seguridad. Esta llave va cogida mediante una cuerda a una pulsera que el usuario debe llevar puesta. La cinta tiene un funcionamiento normal mientras la llave esté conectada (botón sin pulsar), pero, si por ejemplo el usuario sufre una caída, la llave se desconecta y la cinta se para.

Se emplean 2 botones más para cambiar entre los 3 estados posibles en los que el sistema puede estar:

1. Sin programa
2. Programa Libre
3. Prueba de Esfuerzo

El programa Libre permite al usuario aumentar o disminuir la velocidad de la cinta mediante otros 2 botones diseñados específicamente para ello.

La Prueba de Esfuerzo consiste en un ejercicio programado en el que la cinta comienza a una velocidad de 6 km/h, para posteriormente ir aumentando 1 km/h cada minuto. Se utiliza otro pulsador para iniciar y detener la prueba, ya que es posible que no todos los usuarios lleguen a alcanzar la velocidad máxima (20 km/h).

Mediante el visualizador se mantiene informado al usuario de la velocidad instantánea de la cinta, así como del programa de ejercicio actual.

El funcionamiento de este diseño se ha verificado tanto en modo de simulación como en un montaje real en el que se ha conectado la DE0-NANO a una botonera, un display y una etapa de potencia en H que controla el motor DC.

6.1 Perspectivas de Desarrollo

Gracias al tipo de programación utilizada, la adición de nuevos programas de la cinta es muy simple. Por tanto, además del programa Libre y la Prueba de Esfuerzo, se podrían añadir nuevos programas, como por ejemplo un entrenamiento por intervalos, alternando tiempo corriendo a la máxima velocidad con otros momentos de carrera suave.

Por otra parte, se podría diseñar la obtención de estadísticas del ejercicio. Se podría visualizar el tiempo total de ejercicio, así como la velocidad media de la cinta y kilómetros recorridos durante ese tiempo.

También se podría diseñar la sincronización con un medidor de frecuencia cardíaca, para así visualizar mediante el LCD el pulso del usuario.

7 Bibliografía

- Quartus II: http://dl.altera.com/?edition=web
- Manual de Usuario Tarjeta DE0-Nano: ftp://ftp.altera.com/up/pub/Altera_Material/12.1/Boards/DE0-Nano/DE0_Nano_User_Manual.pdf
- Información FPGA Altera Cyclone IV EP4CE22F17C6N: http://www.altera.com/literature/hb/cyclone-iv/cyiv-51001.pdf
- Información Visualizador LCD HITACHI HD44780U: https://www.sparkfun.com/datasheets/LCD/HD44780.pdf
- Diseño Digital: Principios y Prácticas. John F.Wakerly

Pliego de condiciones

Pliego de Condiciones

1 Descripción del Proyecto

El presente documento hace referencia al desarrollo de un sistema de control de un motor eléctrico para una cinta de correr sobre una FPGA ALTERA CYCLONE IV.

Las características deben estar sujetas a las especificaciones iniciales del promotor, siempre de acuerdo a lo establecido en el Pliego de Condiciones.

El proyecto queda definido en los siguientes documentos

1. Memoria
2. Pliego de Condiciones
3. Presupuesto

2 Condiciones Generales

2.1 Disposiciones Generales

La finalidad del presente apartado es regular la ejecución del proyecto, acotando las funciones que corresponden al promotor, al proyectista y al fabricante, así como las relaciones entre ellos.

En este apartado se recogen los aspectos legales del proyecto y se fijan las condiciones que regirán la ejecución y puesta en marcha del sistema objeto del proyecto. Se trata de un documento de carácter facultativo, económico y legal que regirá en el desarrollo de los distintos módulos que constituyen el presente proyecto.

2.2 Documentación del Contrato

Será condición indispensable el conocimiento y aceptación del presente Pliego de Condiciones por parte del fabricante para la correcta materialización del proyecto. Por lo tanto, este pliego deberá incluirse como un documento más a firmar por el fabricante al hacerse cargo de la ejecución. El contrato se encuentra formado por los siguientes documentos:

- Condiciones fijadas en el documento de contrato.
- Pliego de condiciones técnicas particulares.
- El presente pliego general de condiciones.
- Otra documentación del proyecto: Memoria y Presupuesto.

2.3 Condiciones Generales Facultativas

2.3.1 Funciones a Desarrollar por el Fabricante

Corresponde al fabricante:

1. Organizar los trabajos de montaje, así como elaborar los planos que se necesiten para la elaboración del sistema.
2. Velar por el cumplimiento de la normativa vigente en cuanto a la seguridad e higiene en el trabajo.
3. Ostentar la jefatura de todo el personal que intervenga en la fabricación y coordinar las intervenciones de los subcontratistas.
4. Asegurar la idoneidad de todos los materiales y elementos empleados para la implementación del sistema, rechazando aquellos que no cuenten con las garantías exigidas por la normativa vigente o por el presente pliego de condiciones.
5. Firmar con el promotor las actas de recepción provisional y definitiva.
6. Concertar los seguros por accidente de trabajo durante la fabricación.

Derechos y Obligaciones:

1. Verificar los documentos del proyecto así como conocer la normativa al respecto. El fabricante deberá indicar que la documentación entregada es suficiente para la comprensión del proyecto, o en caso contrario solicitar las aclaraciones pertinentes.
2. El fabricante debe ejecutar los trabajos necesarios para el correcto montaje y buen funcionamiento del sistema, aun cuando no se hallen expresamente determinados, siempre que lo disponga el ingeniero proyectista dentro de los límites del presupuesto.
3. Las modificaciones y aclaraciones de los documentos serán comunicadas por escrito al fabricante, debiendo éste devolver los originales, y signando al pie de todas las instrucciones, avisos u órdenes.
4. El fabricante podrá requerir del ingeniero proyectista tantas instrucciones o aclaraciones como sea necesario. Recibirá también solución a los problemas no previstos en el proyecto.
5. Las reclamaciones por parte del fabricante deberán presentarse ante el promotor si son de índole económica. En el caso de ser de orden técnico se deberán presentar ante el ingeniero proyectista. En ambos casos deberán presentarse por escrito. El proyectista podrá limitar su contestación al acuse de recibo del escrito, que en todo caso será obligatorio para este tipo de reclamaciones.

2.3.2 Funciones a Desarrollar por el Ingeniero Proyectista

Es el máximo responsable de la ejecución del proyecto. Decide sobre el comienzo, ritmo y calidad de los trabajos, y velará por el cumplimiento de los mismos.

Corresponde al ingeniero proyectista:

1. Redactar los complementos o modificaciones del proyecto que se precisen.
2. Asistir al montaje cuando se le requiera para asistir al fabricante ante las posibles contingencias que se produzcan e impartir las instrucciones necesarias.
3. Asesorar al promotor en el acto de recepción.
4. Exigir la modificación o agregación de nuevos elementos al sistema si así lo cree conveniente, siempre y cuando no constituya una variación excesiva sobre el proyecto inicial.
5. Coordinar la intervención de otros técnicos durante el montaje.

2.4 Condiciones Generales de la Ejecución

2.4.1 Comienzo y Ritmo de Ejecución de los Trabajos

El promotor marcará el comienzo y el ritmo, para que el trabajo quede ejecutado dentro del plazo exigido

2.4.2 Orden de los Trabajos

El fabricante será el encargado de determinar el orden de los trabajos, salvo en casos que por circunstancias técnicas el ingeniero proyectista estime conveniente su variación.

2.4.3 Ampliación del Proyecto por Causas Imprevistas o de Fuerza Mayor

Si hubiera que ampliar el proyecto, ya sea por causas imprevistas o razones de fuerza mayor, no se interrumpirán los trabajos, continuándose según las instrucciones dadas por el ingeniero en tanto se formula o tramita el proyecto reformado.

2.4.4 Prórroga por Causa de Fuerza Mayor

Si por causas de fuerza mayor no pudiesen iniciarse los trabajos, o se tuviesen que suspender o no se acabasen en los plazos prefijados, se otorgará una prórroga para su finalización, consultándolo antes al ingeniero proyectista.

2.4.5 Condiciones Generales de Ejecución de los Trabajos

Los trabajos se realizarán de acuerdo al proyecto, a las modificaciones del mismo que hayan sido aprobadas y a las órdenes e instrucciones que entregue por escrito el ingeniero proyectista

Cualquier variación sin el conocimiento o aprobación por parte del ingeniero proyectista, supondrá que el fabricante ejecutante del proyecto responderá de todas las consecuencias técnicas, económicas que ocasione dicha variación. No será justificante que la variación proviniera del promotor.

2.4.6 Trabajos Defectuosos

El fabricante deberá realizar los trabajos de acuerdo con lo establecido en el pliego de condiciones, y a falta de especificación, según las órdenes del ingeniero proyectista, empleando los materiales y mano de obra que cumplan con las condiciones exigidas.

Tanto el ingeniero proyectista como sus representantes podrán disponer que las partes defectuosas sean sustituidas, de acuerdo con lo contratado, siendo responsabilidad del fabricante la modificación del producto hasta cumplir con lo especificado.

2.4.7 Averías y Accidentes en la Fabricación

Serán de cuenta y riesgo del fabricante la aportación de toda la maquinaria y mano de obra necesaria para la materialización del proyecto.

Por tanto, es el fabricante y no el promotor el máximo responsable de cualquier avería o accidente personal. Se deberá disponer de todos los medios de seguridad necesarios, de acuerdo con la legislación vigente.

2.4.8 Propiedad del Producto

El producto queda en propiedad del promotor, no pudiendo ser copiado por los fabricantes que intervienen en su materialización sin consentimiento del propietario.

2.4.9 Trabajos Sin Preinscripciones

El fabricante acatará las instrucciones que el ingeniero proyectista dicte en el caso de los trabajos para los que no existan preinscripciones.

2.4.10 Recepción del Proyecto

Una vez concluido el montaje se realizará una inspección del mismo por parte del ingeniero proyectista. Si este considera necesaria una modificación, se dará un plazo para realizar

las modificaciones pertinentes. Si el resultado es admitido, se facilitará la documentación final con las especificaciones y contenidos dispuestos por la legislación vigente.

2.5 Condiciones Generales Económicas

2.5.1 Forma de pago y plazos

El fabricante percibirá el importe íntegro de todos los trabajos ejecutados siempre que estos se hayan realizado según el proyecto o a las modificaciones del ingeniero proyectista. La forma de pago y los plazos serán acordados entre el promotor y el fabricante.

2.5.2 Referencias

Se podrán exigir al fabricante las certificaciones bancarias a fin de cerciorarse de que el fabricante podrá llevar a cabo el cumplimiento del contrato. Deberá entregar estas referencias, si le son requeridas, antes de la firma del contrato.

2.5.3 Fianzas

Las fianzas impuestas al fabricante serán devueltas en un plazo máximo de 8 días una vez finalizado el proyecto, siempre que el fabricante acredite que no existen reclamaciones contra su persona.

2.6 Condiciones Legales

2.6.1 Rescisión de contrato

Se considerarán causas suficientes de rescisión de contrato las siguientes:

1. La modificación del proyecto en forma tal que represente alteraciones fundamentales del mismo a juicio del ingeniero proyectista
2. El incumplimiento por parte del fabricante de las condiciones estipuladas en el presente pliego de condiciones.
3. La falta de cumplimiento de las órdenes recibidas.
4. La insubordinación.

La interpretación de cuantas otras causas de rescisión que pudieran presentarse corresponderían al ingeniero proyectista, a cuyas instrucciones deberá someterse el fabricante.

3 Condiciones Particulares

3.1 Objeto

El objeto de este apartado es la exposición de las especificaciones técnicas de la FPGA, el visualizador y la botonera que se deben utilizar para el correcto funcionamiento del proyecto.

3.2 FPGA

La FPGA utilizada durante el diseño del programa es la FPGA Altera Cyclone IV EP4CE22F17C6N. Para garantizar el correcto funcionamiento del sistema, no se podrá utilizar ningún otro dispositivo. En caso de imposibilidad de conseguirlo por parte del fabricante, se deberá utilizar otra FPGA con características iguales o superiores.

Características FPGA Altera Cyclone IV EP4CE22F17C6N:

- 22.320 elementos lógicos
- 594Kb de memoria embebida
- 4 PLLs de uso general
- 153 entradas/salidas disponibles para el usuario

Durante el presente proyecto se ha utilizado la tarjeta DE0-Nano, la cual incorpora la FPGA antes mencionada. A su vez, esta tarjeta incorpora elementos necesarios para correcto funcionamiento del sistema y facilita la conexión de los periféricos, por lo tanto, se recomienda su utilización.

Características Tarjeta DE0-Nano:

- Dispositivos de Memoria
  - 32 MB SDRAM
  - 2Kb I2C EEPROM
- Periféricos de uso general
  - 8 LEDs verdes
  - 2 pulsadores
  - Acelerómetro de 3 ejes con una resolución de 13 bits
  - Conversor A/D NS ADC128S022 de 8 canales y 12 bits.
- Reloj de 50 MHz
- 2 Módulos de expansión de 40 pines cada uno.
- Fuente de alimentación
  - Puerto MiniUSB (5V)
  - 2 Pines DC 5V para los módulos de expansión

- 2 Pines externos (3.6-5.7V)

3.3 LCD

La pantalla de cristal líquido utilizada en el presente proyecto es la HD44780U, de la marca HITACHI. Las características del visualizador son las siguientes:

- 16 caracteres por 2 líneas
- Display de matriz de puntos de 5x8 o 5x10
- Alta variedad de funciones de instrucción
  - Encendido/apagado del Display
  - Desplazamiento izquierda/derecha del cursor
  - Regreso del cursor al inicio
  - …
- Reconoce ASCII estándar
- Soporta 132 caracteres alfanuméricos y 32 de control
- Bajo consumo de energía

Presupuesto

Presupuesto

1 Introducción

En el presente documento se muestra el presupuesto del desarrollo del trabajo. Dado el carácter inmaterial del mismo, solamente se muestran los costes de los recursos humanos necesarios y los costes de amortización de los equipos.

No se han realizado estudios completos mercado ni de viabilidad, por lo tanto no se incluyen en el presupuesto.

2 Coste de la Mano de Obra

El trabajo se ha realizado en el departamento de ingeniería electrónica de la Universidad Politécnica de Valencia. El autor no ha recibido remuneración debido al carácter académico del mismo, pero se deben contemplar las horas de trabajo dedicadas.

El tiempo dedicado a la realización del trabajo se estima en 300 horas, en las cuales se incluye el diseño del programa y la simulación del mismo, así como el trabajo en el laboratorio para su perfeccionamiento. También se incluye en las 300 horas la redacción de los documentos.

El coste de la mano de obra para un Ingeniero en Tecnologías Industriales se estima en 50 € la hora.

3 Cuadro de Precios Unitarios

Mano de obra			
Nº	**Unidad**	**Descripción**	**Precio**
1	Hora	Ingeniero en Tecnologías Industriales	50,00 €

4 Cuadro de Precios Parciales

4.1 Presupuesto Parcial de la Mano de Obra

Nº	Unidad	Descripción	Cantidad	Precio Unitario	Total
1	Hora	Ingeniero en Tecnologías Industriales	300	50,00 €	15.000,00 €
				Total Mano de Obra Directa	15.000,00 €
				Total Mano de Obra Indirecta 10%	1.500,00 €
				Total Presupuesto Parcial de Mano de Obra	**16.500,00 €**

4.2 Presupuesto Parcial de Amortización de Equipos

Concepto	Periodo de Amortización	Periodo Computado	Precio	Coste
Tarjeta DE0-Nano	10 años	3 meses	120,00 €	3,00 €
Placa de potencia con puente en H	10 años	3 meses	312,00 €	7,80 €
Motor DC	15 años	3 meses	370,00 €	6,17 €
Osciloscopio	15 años	3 meses	1.310,00 €	21,83 €
Fuente de alimentación	15 años	3 meses	220,00 €	3,67 €
Ordenador SONY VAIO VPCEB1J1E	5 años	4 meses	600,00 €	40,00 €
Software Quartus II	1 año	3 meses	2.000,00 €	500,00 €
Software ModelSim	1 año	3 meses	700,00 €	175,00 €
Total Presupuesto Parcial de Amortización de Equipos				**757,47 €**

5 Presupuesto Total

5.1 Presupuesto de Ejecución Material

Concepto	Total
Mano de Obra	16.500,00 €
Amortización de Equipos	757,47 €
Presupuesto de Ejecución Material	**17.257,47 €**

5.2 Presupuesto de Desarrollo

Concepto	Total
Total Presupuesto de Ejecución Material	17.257,47 €
Gastos Generales 15%	2.588,62 €
Beneficio Industrial 10%	1.725,75 €
Subtotal Presupuesto Desarrollo	**21.571,84 €**
I.V.A. (21%)	4.530,09 €
Total Presupuesto Desarrollo	**26.101,93 €**

El presupuesto para el diseño e implementación del control electrónico digital del motor eléctrico en una cinta de correr sobre una FPGA ALTERA CYCLONE IV es de **VEINTISEIS MIL CIENTO UNO CON NOVENTA Y TRES EUROS**.