

DISEÑO DIGITAL 2

MEMORIA DEL DISEÑO:

MEDTH: MEDidor de Temperatura y Humedad

Autor: Nombre y Apellidos

Curso 2020-2021.

Control de versiones

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Fecha | Autor | Cambios realizados |
| 0.0 | 10/02/2017 | DTE | Inicial |
| 1.0 | 13/03/2017 | DTE | Nuevos tics de 1ms y 125 ms en el TIMER. Ambos se utilizan en DISPLAYS. Nueva descripción de la FIFO de lectura en la interfaz de control del periférico I2C (apartado 3.2) |
| 1.1 | 31/03/2017 | DTE | Cambio de la la asignación de pines a las columnas del teclado en la tabla 5.1 |
| 2.0 | 08/01/2018 | DTE | Modificación especificaciones para el funcionamiento del reloj y del display. Apartado 1.3. Modificaciones en la descripción de la interfaz y la funcionalidad de algunos bloques del diseño jerárquico. Apartados 2.2 y 2.6. Modificación de la tabla del apartado 3.2. Modificación de la estructura de los testbench del Apartado 4.4. Se añade la referencia [3] |
| 2.1 | 21/01/2018 | DTE | Modificación de las especificaciones del controlador de teclado. Inclusión de comienzo siempre en edición de horas. |
| 2.2 | 28/02/2018 | DTE | Modificación de la ESP16. Modificación de la tabla de asignación de pines del apartado 5.1. Corrección de la dirección de mux\_disp en el apartado 1.2.3 |
| 2.3 | 11/02/2019 | DTE | Nueva especificación del controlador de teclado (apartado 2.5, pag. 14) |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Tabla de contenido

[1 Especificación del diseño. 4](#_Toc474756418)

[1.1 Introducción 4](#_Toc474756419)

[1.2 Interfaces 4](#_Toc474756420)

[1.2.1 Interfaz con el sensor de temperatura y humedad 4](#_Toc474756421)

[1.2.2 Interfaz con el teclado hexadecimal 4](#_Toc474756422)

[1.2.3 Interfaz con la barra de displays de 7 segmentos 5](#_Toc474756423)

[1.3 Especificaciones funcionales 5](#_Toc474756424)

[1.4 Especificaciones no funcionales 8](#_Toc474756425)

[2 Diseño jerárquico 9](#_Toc474756426)

[2.1 Bloque TEMPORIZADOR 9](#_Toc474756427)

[2.2 Bloque RELOJ 10](#_Toc474756428)

[2.3 Bloque PERIFERICO I2C 11](#_Toc474756429)

[2.4 Bloque PROCESADOR DE MEDIDAS 13](#_Toc474756430)

[2.5 Bloque CONTROLADOR DE TECLADO 14](#_Toc474756431)

[2.6 Bloque DISPLAYS 15](#_Toc474756432)

[3 Diseño detallado 16](#_Toc474756433)

[3.1 Estructura del proyecto 16](#_Toc474756434)

[3.2 Jerarquía del diseño y diseño detallado 17](#_Toc474756435)

[4 Pruebas de verificación funcional de MEDTH 19](#_Toc474756436)

[4.1 Test del reloj 19](#_Toc474756437)

[4.2 Test del periférico I2C 19](#_Toc474756438)

[4.3 Test del controlador de teclado 19](#_Toc474756439)

[4.4 Test del diseño completo. 20](#_Toc474756440)

[5 Diseño físico 20](#_Toc474756441)

[5.1 Asignación de pines 20](#_Toc474756442)

[5.2 Restricciones de la síntesis 21](#_Toc474756443)

[5.3 Recursos utilizados 21](#_Toc474756444)

[5.4 Frecuencia máxima de reloj 21](#_Toc474756445)

[6 Prototipado 22](#_Toc474756446)

[7 Bibliografía 22](#_Toc474756447)

[8 APÉNDICE 1. Interfaz FPGA-sensor. Compatibilidad lógica y MR 23](#_Toc474756448)

[9 APÉNDICE 2. Secuencia de operaciones en la interfaz I2C 24](#_Toc474756449)

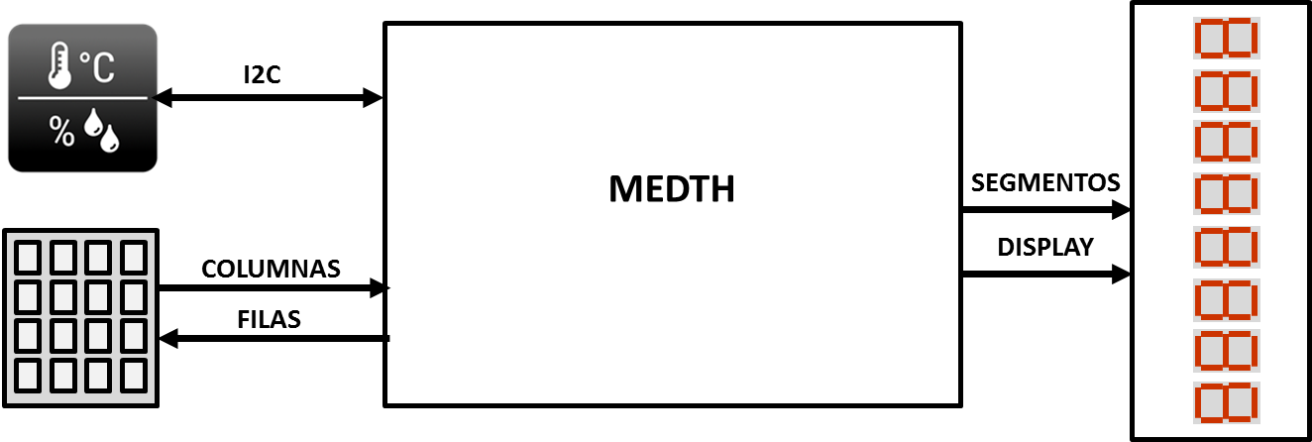
# Especificación del diseño.

## Introducción

El MEDidor de Temperatura y Humedad relativa (en adelante MEDTH) es un sistema que realiza medidas de temperatura y humedad periódicamente y visualiza los resultados en una barra de 8 displays de 7 segmentos. Posee un reloj interno que genera horas, minutos y segundos, el cual también puede mostrarse en los displays. Tanto el reloj como la presentación en los displays pueden configurarse utilizando para ello un teclado hexadecimal.

## Interfaces

El sistema se interconecta al sensor de temperatura y humedad, a un teclado hexadecimal y a una barra de displays de 7 segmentos a través de sendas interfaces que se describen a continuación.



*Fig. 1. Interconexión del MEDTH al sensor de temperatura y humedad, teclado hexadecimal y barra de displays de 7 segmentos.*

### Interfaz con el sensor de temperatura y humedad

El sistema se comunica con el sensor de temperatura y humedad por medio de una interfaz I2C [1]. MEDTH funciona en todo momento como master del bus. Las señales de esta interfaz son las siguientes :

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| SDA | bidireccional (drenador abierto) | Señal de datos de la interfaz I2C |
| SCL | salida (drenador abierto) | Señal de reloj de la interfaz I2C |

### Interfaz con el teclado hexadecimal

El sistema controla un teclado hexadecimal como el de la figura 2. La interfaz consiste en 4 filas y 4 columnas :

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| filas[3..0] | salida | Filas del teclado hexadecimal |
| columnas[3..0] | entrada | Columnas del teclado hexadecimal |
|  | |  |

*Fig. 2. Teclado hexadecimal con su interfaz de filas y columnas.*

El sistema activa inicialmente todas las filas (a nivel bajo) y, cuando se detecta una pulsación, activa secuencialmente cada fila (dejando el resto a nivel alto) y obtiene en las columnas la información que le permite determinar la tecla pulsada.

### Interfaz con la barra de displays de 7 segmentos

El sistema realiza la presentación de los datos utilizando una barra de displays de 7 segmentos del tipo cátodo común. La interfaz es la siguiente :

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| seg[7..0] | salida | seg[0] : segmento g  seg [1] : segmento f  seg [2] : segmento e  seg [3] : segmento d  seg [4] : segmento c  seg [5] : segmento b  seg [6] : segmento a  seg [7] : segmento punto |
| mux\_disp[7..0] | salida | mux\_disp [0] : cátodo del display 0 (LSD)  mux\_disp [1] : cátodo del display 1  mux\_disp [2] : cátodo del display 2  mux\_disp [3] : cátodo del display 3  mux\_disp [4] : cátodo del display 4  mux\_disp [5] : cátodo del display 5  mux\_disp [6] : cátodo del display 6  mux\_disp [7] : cátodo del display 7 (MSD) |

La interfaz permite iluminar solo un display a la vez. El display se selecciona activando (a nivel bajo) el cátodo correspondiente. El display activo se ilumina de acuerdo al código de 7 segmentos y punto decimal introducido (nivel alto).

## Especificaciones funcionales

|  |  |
| --- | --- |
| Ref | Especificación |
| ESP00 | El sistema realiza medidas de temperatura y humedad relativa cada 0.5 segundos. |
| ESP01 | El rango de representación de la temperatura estará entre 125 ºC y  -40 ºC, con una resolución de 1 ºC. |
| ESP02 | El rango de representación de la humedad relativa estará entre el 0% y el 100%, con una resolución de un 1%. |
| ESP03 | El sistema representará la temperatura en BCD, utilizando 6 displays : En cada uno de ellos se representa lo siguiente :   * display 0 (LSD) : letra “C” * display 1 : en blanco (no debe lucir ningún segmento) * display 2 : dígito menos significativo de la temperatura * display 3 : segundo dígito de la temperatura * display 4 : dígito más significativo de la temperatura * display 5 signo “-“ si la temperatura es negativa o en blanco si es positiva   Los ceros no significativos de las decenas y centenas no se representarán. El signo menos, en su caso, se desplazará a la posición de las centenas (si su valor es 0) o de las decenas (si el valor de ambas, decenas y centenas, es 0). |
| ESP04 | El sistema representará la humedad relativa utilizando 5 displays. En cada uno de ellos se representará lo siguiente :   * display 0 (LSD) : letra “h” * display 1 : en blanco (no debe lucir ningún segmento) * display 2 : dígito menos significativo de la humedad relativa * display 3 : segundo dígito de la humedad relativa * display 4 : dígito más significativo de la humedad relativa   Los ceros no significativos de las decenas y centenas no se representarán. |
| ESP05 | El sistema dispondrá de un reloj programable cuya salida también podrá visualizarse en los displays. |
| ESP06 | El reloj contará horas, minutos y segundos. |
| ESP07 | El reloj podrá funcionar en dos modos diferentes :   * modo 12 hs : desde las 00hs, 00min, 00 seg hasta las 11 hs, 59 min, 59 seg * modo 24 hs : desde las 00hs, 00min, 00 seg hasta las 23 hs, 59 min, 59 seg |
| ESP08 | Inicialmente, el reloj comenzará a funcionar en modo 12 h, comenzando a contar desde las 00 hs, 00 min, 00 seg (AM) |
| ESP09 | El sistema representará el reloj utilizando los 8 displays :   * display 0 (LSD) : segundos (LSD) * display 1 : segundos (MSD) * display 2 : minutos (LSD) * display 3 : minutos (MSD) * display 4 : horas (LSD) * display 5 : horas (MSD) * display 6: en blanco (no debe lucir ningún segmento) * display 7: en blanco en el modo 24 hs. En el modo 12 hs identificará AM (con la letra “A”) o PM (con la letra “P”)   Los ceros no significativos de las horas no se representarán. |
| ESP10 | El reloj se programará utilizando el teclado hexadecimal. Para la programación resulta relevante tanto la tecla pulsada como la duración de la pulsación, que podrá ser larga (si dura 2 segundos o más) o corta (inferior a los 2 segundos). |
| ESP11 | El reloj tendrá dos modos de trabajo : normal y programación. Tras el reset estará en el modo normal.  En el modo normal el reloj presentará la hora con una resolución de segundos. En el modo de programación se permitirá el cambio de formato 12h o 24h, así como la edición de los minutos y de las horas, no así el campo correspondiente a los segundos que tomará de manera automática el valor cero una vez acabada la edición. |
| ESP12 | La entrada al modo programación se realizará mediante la pulsación larga de la tecla ‘A’. Siempre se entra en el modo de edición de horas. |
| ESP13 | La salida del modo de programación se llevará a cabo mediante la pulsación breve de la tecla ‘A’, o bien tras siete segundos de inactividad en el teclado. |
| ESP15 | En el modo de programación la tecla ‘B’ permitirá cambiar el campo en edición, alternando entre minutos y horas a cada pulsación. Durante el modo de programación, el campo activo parpadeará con una frecuencia de 4 Hz. |
| ESP16 | La edición del campo que se encuentre seleccionado se podrá efectuar de varias formas. Se podrá incrementar en la unidad el valor de un campo mediante la pulsación breve de la tecla ‘C’. Sin embargo, si se lleva a cabo una pulsación larga de la tecla ‘C’, el valor del campo se incrementará cuatro veces por segundo mientras esta tecla se mantenga activada. Igualmente podremos introducir un valor directamente con el teclado, de la siguiente manera: al pulsar una tecla correspondiente a un dígito decimal, el valor de dicha tecla actualiza las unidades del campo activo, al tiempo que las decenas se ponen a 0; si la siguiente pulsación corresponde a otro dígitro decimal, el valor de la tecla pulsada actualizará las unidades y el dígito de las unidades (la anterior tecla numerica pulsada) pasará a las decenas, si el valor compuesto para el campo es válido (si no lo es, el valor del campo permancece invariable ignorando la segunda pulsación numérica). |
| ESP17 | Sea cual sea el modo del reloj se podrá cambiar el formato 12h y 24h mediante la pulsación breve de la tecla ‘D’. |
| ESP18 | La presentación en la batería de displays de 7 segmentos podrá funcionar en 4 modos diferentes:   * Solo reloj: solo se muestra el reloj como se especifica en ESP09. * Solo temperatura: solo se muestra la temperatura como se especifica en ESP03. * Solo humedad relativa: solo se muestra la humedad relativa como se especifica en ESP04. * Combinado: se muestran alternativamente el reloj, la temperatura y la humedad relativa |
| ESP19 | Inicialmente la presentación en los displays estará configurada en el modo “solo reloj”. |
| ESP20 | En el modo de presentación combinado cada elemento se visualizará durante 16 segundos y luego se pasa a visualizar el siguiente. La entrada de la temperatura y la humedad se producirá de manera secuencial, a razón de un dígito por segundo: primero, el LSD se visualizará en el display 7; un segundo después el LSD se visualizará en el display 6 mientras que el siguiente dígito lo hará en el display 7 y así sucesivamente hasta que se visualice el elemento completo con el LSD en el display 0. Durante los 8 segundos restantes la visualiación no cambiará. El reloj se visualizará durante 16 segundos como en el modo *solo reloj*. |
| ESP21 | El modo de presentación en los displays se programará utilizando el teclado hexadecimal. |
| ESP22 | Para cambiar el modo de programación de la presentación de displays, una pulsación corta de la tecla “E” permite pasar al modo “solo temperatura”. Posteriores pulsaciones cortas en la tecla “E” cambiarán a los modos “solo humedad relativa”, “combinado”, “solo reloj” y así sucesivamente. |
| ESP23 | La comunicación entre el sistema y el sensor de temperatura se realizará mediante una interfaz I2C [1] en modo *fast*, con un reloj I2C de 400 KHz. |

## Especificaciones no funcionales

|  |  |
| --- | --- |
| Ref | Especificación |
| ESP24 | El sistema se diseñará utilizando VHDL. Podrán utilizarse como elementos de librería los IPs de Altera que sean necesarios. |
| ESP25 | Se utilizará *ModelSim* como herramienta de simulación y *Quartus Prime* como herramienta para la realización del diseño físico, |
| ESP26 | El sistema se prototipará utilizando una tarjeta DECA-MAX10 del fabricante Arrow [2] |
| ESP27 | Para prototipar el sistema se conectará a la tarjeta DECA-MAX10 una tarjeta de expansión [3] con los 8 displays y un conector para el teclado hexadecimal. |
| ESP28 | Se utilizará como fuente de reloj uno de los osciladores de 50MHz que posee la tarjeta DECA-MAX10. |

# Diseño jerárquico

El diagrama de la Fig. 3 representa el primer nivel de la jerarquía del diseño[[1]](#footnote-1):

*Fig. 3. Diagrama de bloques del primer nivel de la jerarquía de MEDTH.*

En los siguientes subapartados se describe la interfaz y la función de cada uno de estos bloques.

## Bloque TEMPORIZADOR

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| tic\_025s | salida | Tic con un período de 250 ms. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |
| tic\_1s | salida | Tic con un período de 1 s. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |
| tic\_5ms | salida | Tic con un período de 5 ms. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |
| tic\_1ms | salida | Tic con un período de 5 ms. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |
| tic\_125ms | salida | Tic con un período de 125 ms. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |

Este bloque genera de manera centralizada las señales de tiempo que requieren los bloques RELOJ, PROCESADOR DE MEDIDA, CONTROLADOR DE TECLADO y DISPLAYS.

## Bloque RELOJ

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| tic\_025s | entrada | Tic de de 250 ms. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |
| tic\_1s | entrada | Tic de 1 s. Activo a nivel alto, tiene la anchura de un período del reloj del sistema. |
| tecla[3..0] | entrada | Código hexadecimal correspondiente a la pulsación de una tecla en el treclado hexadecimal. |
| tecla\_pulsada | entrada | Validación de tecla\_pulsada para pulsos cortos (duración inferior a 2 segundos). Activa a nivel alto, dura un período de reloj del sistema. |
| pulso\_largo | entrada | Validación de tecla\_pulsada para pulsos largos (2 segundos o más). Es excluyente con tecla\_pulsada. Se activa cuando se detecta una pulsación larga y se mantiene activa hasta que deja de pulsarse la tecla. |
| segundos[7:0] | salida | Dígitos BCD correspondientes a los segundos de la hora actual (entre 00 y 59). |
| minutos[7:0] | salida | Dígitos BCD correspondientes a los minutos de la hora actual (entre 00 y 59) |
| horas[7:0] | salida | Dígitos BCD correspondientes a las horas de la hora actual (entre 00 y 11 para el modo 12 h. y entre 00 y 23 para el modo 24 h.) |
| AM\_PM | salida | Indica, en el modo 12 h., si el período es AM (0) o PM (1). |
| modo | salida | Indica si el reloj está en modo 12 h (0) o 24 h (1). |
| info[1..0] | salida | Indica cuál es el campo activo en el modo programación : "00" ninguno, "01" minutos y "10" horas. |

El bloque funciona de la siguiente manera:

* Genera en sus salidas horas, minutos y segundos la hora actual expresada en BCD.
* Tiene dos modos de cuenta:
  + 12 h: cuenta desde 00:00:00 hasta 11:59:59 en dos períodos, AM y PM. La salida AM\_PM indica el período (0: AM; 1: PM)
  + 24 h: cuenta desde 00:00:00 hasta 23:59:59. La salida AM\_PM no tiene significado alguno en este modo

La salida modo informa sobre si el modo es 12 h (0) o 24 h (1). Inicialmente el modo es 12 h.

* Tiene dos modos de funcionamiento: normal y programación. En el modo normal presenta la cuenta mientras que en el modo de programación se puede cambiar el modo de cuenta (12h/24h) así como editar los valores de las horas y los minutos. Los segundos no pueden cambiarse y toman automáticamente el valor 0 tras cada edición.
* La **entrada** al modo programación se realizará mediante la pulsación **larga** de la **tecla ‘A’**. Siempre se entra en el modo de edición de horas. La **salida** del modo de programación se llevará a cabo mediante la pulsación **breve** de la **tecla ‘A’**, o bien tras siete segundos de inactividad en el teclado.
* En el modo de programación la **tecla ‘B’** permitirá **cambiar el campo** en edición, alternando entre minutos y horas a cada pulsación.
* La edición del campo que se encuentre seleccionado se podrá efectuar de varias formas. Se podrá **incrementar** en la unidadel **valor** de un **campo** mediante la pulsación breve de la **tecla ‘C’**. Sin embargo, si se lleva a cabo una pulsación **larga** de la **tecla ‘C’**, el valor del campo se incrementará **cuatro veces por segundo** mientras esta tecla se mantenga activada. Igualmente podremos **introducir un valor** directamente con el teclado.
* Sea cual sea el modo del reloj se podrá **cambiar el formato 12h y 24h** mediante la pulsación breve de la **tecla ‘D’**.
* La salida “**info**” indicará qué campo está siendo editado, o si el reloj se encuentra en modo de funcionamiento normal, presentando la hora.

## Bloque PERIFERICO I2C

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| we | entrada | Señal que indica la realización de una operación de escritura. |
| rd | entrada | Señal que indica la realización de una operación de lectura. |
| add[1:0] | entrada | Dirección de lectura o escritura. |
| dato\_in[7:0] | entrada | Entrada de datos para las operaciones de escritura. |
| SDA | bidireccional | Entrada/salida de datos para la interfaz I2C. Drenador abierto. |
| SCL | salida | Salida de reloj para la interfaz I2C. Drenador abierto. |
| dato\_out[7:0] | salida | Salida de datos para las operaciones de lectura. |

El bloque funciona de la siguiente manera:

* El circuito posee dos interfaces: interfaz de programación e interfaz I2C. La interfaz de programación permite escribir en sus registros de programación y leer de sus registros de datos y estado. La interfaz I2C permite escribir y leer en el sensor de temperatura y humedad.
* La interfaz de programación tiene 3 registros de escritura y 4 de lectura:

|  |  |  |  |
| --- | --- | --- | --- |
| Registro | add | rd/wr | Descripción |
| REG\_WR\_0 | 0 | wr | Registro de escritura de bytes en el bus I2C |
| REG\_WR\_1 | 1 | wr | Registro de número de bytes en operaciones de lectura |
| REG\_WR\_3 | 3 | wr | Registro de comandos |
| REG\_RD\_0 | 0 | rd | Registro de lectura de bytes en el bus I2C |
| REG\_RD\_1 | 1 | rd | Registro de número de bytes pendientes de lectura |
| REG\_RD\_2 | 2 | rd | Registro que almacena el número de bytes escritos |
| REG\_RD\_3 | 3 | rd | Registro de estado. |

* Los registros de la interfaz de programación tienen la siguiente funcionalidad:
* REG\_WR\_0: en realidad es un buffer que permite almacenar hasta 256 datos. Almacena los datos que se escribirán en el bus I2C durante la próxima transferencia.
* REG\_WR\_1: permite programar el número de bytes que han de ser leídos durante una transferencia en modo lectura.
* REG\_WR\_3: permite iniciar las transferencias de datos en el bus I2C así como el borrado de los buffers de lectura o escritura:

|  |  |
| --- | --- |
| Bit | Descripción |
| 0 | 1: inicia una transacción (no es necesario ponerlo a 0) |
| 1 | 1: todos los bytes transferidos en el bus I2C se almacenan en el buffer de lectura  0: necesario si el bit 2 se pone a 1 |
| 2 | 1: solo los bytes leídos a través del bus I2C se almacenan en el buffer de lectura  0: necesario si el bit 1 se pone a 1 |
| 3 | 1: borra el buffer de lectura (no es necesario ponerlo a 0). |
| 4 | 1: borra el buffer de escritura |

* REG\_RD\_0: en realidad es un buffer que permite almacenar hasta 256 datos. Almacena los datos que se leen por el bus I2C en la transferencia actual.
* REG\_RD\_1: almacena el número de bytes pendientes de leer del buffer de lectura
* REG\_RD\_2: almacena el número de bytes pendientes de leer del buffer de escritura
* REG\_RD\_3: registro de estado:

|  |  |  |
| --- | --- | --- |
| Bit | Descripción | Valoreset |
| 0 | 1: el periférico listo para iniciar una transacción  0: hay una transacción en curso | 1 |
| 1 | 1: la última transferencia se ha realizado correctamente  0: ha habido un error en la última transferencia | 0 |
| 2 | 1: el modo de lectura de todos los bits de una transferencia está activo  0: modo de lectura de solo los bytes leídos en una transferencia está activo | 1 |
| 3 | 1: la transferencia actual es una operación de lectura  0: la transferencia actual es una operación de escritura | 0 |
| 4 | Flag que indica (1) que el buffer de escritura está vacío | 1 |
| 5 | Flag que indica (1) que el buffer de escritura está lleno | 0 |
| 6 | Flag que indica (1) que el buffer de lectura está vacío | 1 |
| 7 | Flag que indica (1) que el buffer de lectura está lleno | 0 |

* Para ordenar una transacción de escritura en el bus I2C:
* Se espera a que el periférico esté listo (un 1 en el bit 0 de REG\_RD\_3)
* Se escribe en el buffer de escritura el primer byte con la dirección I2C y el tipo de operación (LSB a 0: escritura)
* Se escriben en el buffer de escritura uno o más bytes, en orden de transmisión, hasta un máximo de 255
* Se ordena la ejecución de la transacción (bit 1 de REG\_WR\_3 a 1)
* Para ordenar una transacción de lectura en el bus I2C:
* Se espera a que el periférico esté listo (un 1 en el bit 0 de REG\_RD\_3)
* Se escribe en el buffer de escritura el primer byte con la dirección I2C y el tipo de operación (LSB a 1: lectura)
* Se programa el número de bytes que se desea leer (REG\_WR\_1)
* Se ordena la ejecución de la transacción (bit 1 de REG\_WR\_3 a 1)

## Bloque PROCESADOR DE MEDIDAS

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| tic\_025s | entrada | Tic de 250 ms, activo a nivel alto. |
| dato\_out[7:0] | entrada | Entrada de datos para las operaciones de lectura del bloque PERIFERICO I2C. |
| we | entrada | Señaliza una opereción de escritura en el bloque PERIFERICO I2C. |
| rd | entrada | Señaliza una operación de lectura del PERIFERICO I2C. |
| add[1:0] | salida | Dirección de lectura o escritura para el PERIFERICO I2C. |
| dato\_in[7:0] | salida | Salida de datos para las operaciones de escritura en el PERIFERICO I2C |
| temperatura[11:0] | salida | Salida de temperatura (BCD 3 dígitos) |
| signo\_temp | salida | Signo de la temperatura : + (0) o – (1) |
| humedad[11:0] | salida | Salida de humedad relativa (BCD 3 dígitos) |

El bloque funciona de la siguiente manera:

* El circuito se sincroniza con los tics de 0.25 segundos que genera el timer.
* El circuito espera la llegada de un tic y realiza el disparo de una captura de temperatura y humedad, escribiendo en el puntero del sensor (dirección I2C x40) la dirección del registro de temperatura (x00).
* En el siguiente tic se lee la temperatura y la humedad en una lectura I2C de 4 bytes. El valor leido de ambas magnitudes se convierte a BCD y se entrega por las salidas temperatura (valor absoluto de la temperatura), signo\_temp (signo, positivo o negativo, de la temperatura) y humedad (humedad relativa).
* La resolución de la temperatura es de 1 ºC y la de la humedad relativa de un 1%
* En el siguiente tic se realiza una nueva captura de temperatura y humedad relativa, y así sucesivamente.

## Bloque CONTROLADOR DE TECLADO

|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| filas[3..0] | salida | Filas del teclado hexadecimal |
| columnas[3..0] | entrada | Columnas del teclado hexadecimal |
| tic\_5ms | entrada | Tic de 5 ms, activo a nivel alto |
| tecla[3:0] | salida | Código de la tecla pulsada |
| tecla\_pulsada | salida | Tic que valida la tecla pulsada con pulsación corta |
| pulso\_largo | salida | Detección de pulsación larga. |
|  | |  |

El bloque funciona de la siguiente manera:

* Inicialmente activa la fila 0 (fila[1:0]=1110)
* Cada vez que recibe un tic de 5 ms, activa una nueva fila desactivando la anterior, siguiendo la secuencia: fila0, fila1, fila2, fila3, fila0…
* Cuando detecta la activación, a nivel bajo, de alguna de las columnas, congela la fila activa.
  + Si la columna se desactiva antes de 2 segundos, entonces la pulsación es corta. El circuito activa tecla\_pulsada durante un ciclo de reloj y genera en tecla[3:0] el código de la tecla pulsada, comenzando de nuevo el barrido de las filas.
  + En caso contrario, entonces la pulsación es larga. El circuito activa pulsación\_larga hasta que se desactive la columna. También genera el tecla[3:0] el código de la tecla pulsada.
* El circuito utiliza como única referencia de tiempo el tic de 5 ms. Este tic es, por tanto, el que marca la precisión con la que se diferencian las pulsaciones largas o cortas.
* El circuito utiliza también el tic de 5 ms para registrar el valor de las columnas del teclado. Este procedimiento hace que sea inmune a los rebotes que pudiera tener el pulsador.

## Bloque DISPLAYS

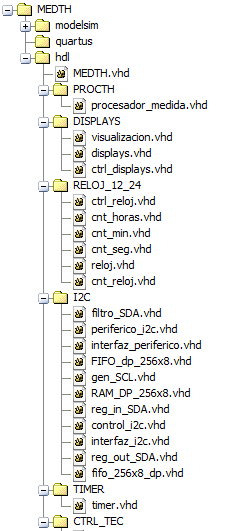
|  |  |  |
| --- | --- | --- |
| Señal | Dirección | Descripción |
| tic\_1ms | entrada | Tic de 1 ms, activo a nivel alto. |
| tic\_125ms | entrada | Tic de 125 ms, activo a nivel alto. |
| tic\_1s | entrada | Tic de 1 s, activo a nivel alto. |
| tecla\_pulsada | entrada | Señalización de tecla pulsada para las pulsaciones cortas. |
| tecla[3:0] | entrada | Código de la tecla pulsada. |
| pulso\_largo | entrada | Señalización de pulsación larga. |
| temperatura[11:0] | entrada | Temperatura medida (módulo), en BCD de 3 dígitos. |
| signo\_temp | entrada | Bit de signo de la temperatura. |
| humedad[11:0] | entrada | Humedad relativa medida, en BCD de 3 dígitos. |
| modo | entrada | Modos de trabajo del reloj (AM o PM). |
| info[1 :0] | entrada | Campo activo en modo programación del reloj. |
| horas[7:0] | entrada | Horas del reloj, BCD de 2 dígitos. |
| minutos[7:0] | entrada | Minutos del reloj, BCD de 2 dígitos. |
| segundos[7:0] | entrada | Segundos del reloj, BCD de 2 dígitos. |
| disp\_mux[7:0] | salida | Selector de display, activo a nivel bajo |
| disp\_seg[7:0] | salida | Código de 7 segmentos del display y punto decimal, activo a nivel alto. |

El bloque funciona de la siguiente manera:

* Tiene dos modos de funcionamiento: *normal* y *programación*. En el modo *normal* presenta los datos en el último modo de visualización programado. En el modo *programación* permite la configuración de un nuevo modo de visualización.
* Dispone de 4 modos de visualización:
  + *Solo reloj* (modo = 00): se visualiza el reloj.
  + *Solo temperatura* (modo = 01): se visualiza la temperatura.
  + *Solo* *humedad relativa* (modo = 10): se visualiza la humedad relativa.
  + *Combinado* (modo = 11): se visualizan el reloj, la temperatura y la humedad relativa secuencialmente.
* Inicialmente el circuito está en el modo de funcionamiento *normal* y en el modo de visualización *solo reloj*.
* Una pulsación corta en la tecla “E” permite cambiar el modo de visualización, de solo reloj a solo temperatura. Ulteriores pulsaciones cortas en la tecla “E” permiten cambiar el modo a *solo humedad relativa*, *combinado*, *solo reloj*, y así sucesivamente.
* En el modo combinado se alterna la visualización del reloj, temperatura y humedad relativa cada 16 segundos. Cada vez que se visualizan la temperatura o la humedad, los caracteres entran por la izquierda de la barra de displays a razón de un carácter por segundo. Una vez transcurridos los 8 primeros segundos, el elemento completo se visualiza en los displays durante los 8 segundos restantes. El reloj se visualiza durante los 16 segundos como en el modo *solo reloj*.
* Cuando se está visualizando el reloj, si éste se está programando, el campo activo parpadea con una frecuencia de 4 Hz.
* No muestra los dígitos no significativos de la temperatura, la humedad relativa, o las horas del reloj. Cuando visualiza la temperatura, si ésta es negativa, el signo aparece siempre al lado del dígito más significativo que se esté visualizando.
* La visualización en los displays utiliza la técnica de multiplexión temporal. Cada display se visualiza durante 1 ms, permaneciendo el resto apagados, de manera que la tasa de refresco de cada display es de 125 veces por segundo.

# Diseño detallado

## Estructura del proyecto

El proyecto está almacenado en la carpeta MEDTH, que a su vez contiene las carpetas hdl, modelsim y quartus. La carpeta *hdl* contiene los ficheros RTL y estructural del diseño, así como los modelos VHDL para simulación de los IPs de Altera que se han utilizado. La carpeta *modelsim* contiene el proyecto de simulación (MEDTH.mpf), y los ficheros donde se definen los diferentes test‑benches. Finalmente, *quartus* contiene el proyecto para el diseño físico y los ficheros relacionados con éste. La siguiente figura muestra la jerarquía de carpet as dentro de la carpeta MEDTH:

*Fig. 4. Estructura de carpetas del proyecto.*

## Jerarquía del diseño y diseño detallado

En la siguiente tabla se muestra la jerarquía del diseño y se realiza una descripción somera de los módulos que la componen. Los detalles del funcionamiento de cada módulo se han documentado en comentarios en la cabecera de los propios módulos.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Rojo : descripciones estructurales, Azul : descripciones RTL, Verde : descripciones mixtas (RTL con algún módulo instanciado), Negro : IPs de Altera | | | | | |
| ***Nivel jerárquico*** | | | | | ***Descripción*** |
| 1 (top) | 2 | 3 | 4 | 5 |
| MEDTH | | | | | Medidor de temperatura y humedad (nivel superior de la jerarquía) |
|  | timer | | | | Temporizador. Genera tics de 5 ms, 250 ms y 1 s. |
|  | ctrl\_tec(TBD) | TBD |  |  | Controlador de teclado. Codifica la tecla pulsada y detecta pulsación larga o corta. |
|  |  |  |  |  | Añada las filas que necesite |
|  |  |  |  |  |  |
|  | reloj | | | | Reloj programable 12h/24h. Nivel superior de la jerarquía del reloj. |
|  | ctrl\_reloj | | | Control del reloj. Interpreta los comandos del teclado y genera las señales de control apropiadas para el módulo contador de reloj. |
| cnt\_reloj | | | Contador de reloj. Instancia contadores para los segundos, minutos y horas. |
|  | cnt\_seg | | Contador de segundos. |
| cnt\_min | | Contador de minutos. |
| cnt\_horas | | Contador de horas. |
|  | periferico\_i2c | | | | Periférico I2C. Nivel superior de la jerarquía del circuito. Instancia una interfaz de programación y un circuito de comunicaciones compatible I2C. |
|  | interfaz\_control | | | Interfaz paralelo de control/datos. Permite la programación del tipo de operación con la interfaz I2C así como la lectura y la escritura de los datos. Posee un modelo de programación basado en registros. Instancia una FIFO de lectura y otra de escritura. |
| |-----------  ……..|  ……..|----------- | fifo\_rd:fifo\_dp\_256x8 | | FIFO de lectura. Almacena los bytes leídos a través de la interfaz paralelo de control/datos para que puedan ser, a su vez, leídos a través de la interfaz I2C. FIFO de doble puerto de 256 x 8 bits de la librería de IPs de Altera. |
|  | |  |
| fifo\_wr:fifo\_256x8\_dp | | FIFO de escritura. Almacena los bytes leídos a través de la interfaz I2C para que puedan ser, a su vez, leídos a través de la interfaz paralelo de control/datos. Instancia una RAM. |
| |---------------- | ram\_dp\_256x8 | RAM de doble puerto de 256 x 8 bits de la librería de IPs de Altera. |
| interfaz\_i2c | | | Circuito de comunicaciones compatible I2C. Instancia un controlador, un circuito para generar la señal SCL y tres circuitos para generar, recibir y filtrar la señal SDA. |
|  | control\_i2c | | Controlador. Iniciado por el circuito de interfaz\_control. |
| gen\_scl | | Genera la señal de reloj SCL de la interfaz I2C. |
| reg\_out\_sda | | Lee en paralelo de interfaz\_control los bytes para transmitir y los envía en serie por SDA.. |
| filtro\_sda | | Filtra los datos que entran por SDA en las operaciones de lectura. |
| reg\_in\_sda | | Lee los bits SDA filtrados y los entrega, en paralelo (bytes), a interfaz\_control. |
|  | displays | | | | Controlador de displays. Interpreta los comandos del teclado e identifica el modo de visualización. Multiplexor de datos. Visualiza los datos en la barra de displays en función del modo de visualización. Utiliza multiplexión temporal de los datos. |
|  | procesador de medida | | | | Procesador de temperatura y humedad. Ordena al periférico I2C que realice lecturas de temperatura y humedad cada 0.5 seg. Codifica los datos obtenidos en BCD de 3 dígitos. |

# Pruebas de verificación funcional de MEDTH

El plan de pruebas de MEDTH consiste en un conjunto de tests dirigidos. Se realizan tests específicos para los bloques funcionales más complejos, tests en los que se combinan varios bloques funcionales cuando resulta necesario y un test final para el diseño completo. Algunas de las pruebas requieren escalado temporal para reducir su duración.

En los siguientes subapartados se describen los tests realizados.

## Test del reloj

Es un test específico para el módulo del reloj. Consiste en un conjunto de pruebas exhaustivo en las que se comprueba el funcionamiento de todas las opciones de configuración y en todos los modos de trabajo.

|  |  |  |
| --- | --- | --- |
| Ubicación de los ficheros del test | En /MEDTH/modelsim/TEST | |
| Simulación escalada | Sí | |
| Ficheros | test\_reloj | testbench |
| pack\_test\_reloj | paquete con procedimientos para el test |
| test\_estimulos\_reloj | código de generación de secuencias de test |
| test\_monitor\_reloj | código de verificación automática |
| Descripción del test | Se verifica el modo normal de funcionamiento para todas las posibles configuraciones y el modo de programación para todos los comandos. | |

## Test del periférico I2C

Es un test específico para el periférico I2C. Consiste en un conjunto de pruebas exhaustivo que comprueba el funcionamiento de todos los modos de programación, así como el cumplimiento de las especificaciones de compatibilidad con el bus I2C.

|  |  |  |
| --- | --- | --- |
| Ubicación de los ficheros del test | En /MEDTH/modelsim/TEST | |
| Simulación escalada | No | |
| Ficheros | test\_periferico\_i2c | testbench |
| Descripción del test | Se verifica toda la funcionalidad del periférico I2C | |

## Test del controlador de teclado

TBD

## Test del diseño completo.

Es un test que combina todos los bloques funcionales del diseño para verificar su funcionamiento conjunto.

|  |  |  |
| --- | --- | --- |
| Ubicación de los ficheros del test | En /MEDTH/modelsim/TEST | |
| Simulación escalada | SI | |
| Ficheros | test\_top.vhd | testbench |
| pack\_test\_reloj.vhd | contiene procedimientos para facilitar el test del reloj |
| pack\_test\_teclado.vhd | contiene procedimientos para la emulación del teclado |
| Descripción del test | Es un test básico en el cual se realizan pruebas simples de configuración del reloj y se verifican los 4 modos de funcionamiento de la visualización en los displays. | |

# Diseño físico

En este apartado se documentan los detalles básicos relacionados con el diseño físico del circuito: la asignación de pines, las restricciones de la síntesis y los informes que proporciona Quartus Prime sobre los recursos de la FPGA utilizados y la frecuencia máxima de reloj obtenida.

## Asignación de pines

En la siguiente tabla se detalla la asignación de los pines de la interfaz de MEDTH a los pines de la FPGA especificando para cada caso el tipo de pin, el número de pin de la FPGA que se corresponde con el pin del diseño, el banco al que corresponde, el estándar de entrada/salida que utiliza, el fan-out (para los pines de salida) y el slew-rate.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Pin de la interfaz del diseño | direccion | Pin FPGA | I/O bank | I/O standard | Current strength  (mA) | Slew rate | Pull-up interno |
| clk | Input | M8 | 2 | 2.5-V | N.A. | N.A. | No |
| rst\_n | Input | H21 | 6 | 1.5-V Scmitt Trigger | N.A. | N.A. | No |
| SCL |  |  |  |  |  |  | No |
| SDA |  |  |  |  |  |  | No |
| fila[0] | Output | Y18 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| fila[1] | Output | AA17 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| fila[2] | Output | AA19 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| fila[3] | Output | AB20 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| columna[0] | Input | W18 | 4 | 3.3-V Scmitt Trigger | N.A. | N.A. | Sí |
| columna[1] | Input | Y19 | 4 | 3.3-V Scmitt Trigger | N.A. | N.A. | Sí |
| columna[2] | Input | AA20 | 4 | 3.3-V Scmitt Trigger | N.A. | N.A. | Sí |
| columna[3] | Input | AB21 | 4 | 3.3-V Scmitt Trigger | N.A. | N.A. | Sí |
| seg [0] | Output | W16 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [1] | Output | AB11 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [2] | Output | W15 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [3] | Output | AB10 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [4] | Output | AA15 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [5] | Output | W12 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [6] | Output | AA13 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| seg [7] | Output | AB12 | 4 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[0] | Output | Y5 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[1] | Output | W6 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[2] | Output | W8 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[3] | Output | AB8 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[4] | Output | R11 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[5] | Output | AB6 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[6] | Output | AA6 | 3 | 3.3-V LVTTL | 8 | 2 | No |
| mux\_disp[7] | Output | V10 | 3 | 3.3-V LVTTL | 8 | 2 | No |

## Restricciones de la síntesis

TBD

## Recursos utilizados

TBD

## Frecuencia máxima de reloj

TBD

# Prototipado

En este apartado se enumeran las pruebas realizadas con el prototipo y el resultado de cada una de ellas. Todas las pruebas se han realizado en las siguientes condiciones:

* Tarjeta DECA-MAX10 con la tarjeta de expansión conectada.
* Tarjeta alimentada con el alimentador de 5 V DC
* FPGA de la tarjeta configurada con el diseño MEDTH (medth.pof)

|  |  |  |
| --- | --- | --- |
| Ref. | Descripción | Resultados |
| PR00 | Reset inicial | El circuito muestra el reloj contando en modo 12h, AM, avanzando desde las 00:00:00. |
| TBD | TBD | TBD |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Bibliografía

[1] UM10204. I2C bus specification and user manual [online] <http://www.nxp.com/documents/user_manual/UM10204.pdf>

[2] Tarjeta DECA-MAX10 (página web del fabricante). [online] <https://www.arrow.com/en/products/deca/arrow-development-tools>

[3] Tarjeta XDECA. Manual de usuario. [moodle DD2-documentacion técnica]

# APÉNDICE 1. Interfaz FPGA-sensor. Compatibilidad lógica y MR

TBD

# APÉNDICE 2. Secuencia de operaciones en la interfaz I2C

TBD

1. Todos los bloques tienen un reloj común, clk, y una entrada de reset asíncrono, rst\_n, activa a nivel bajo. Estas señales no se incluyen en las interfaces por simplicidad. [↑](#footnote-ref-1)