DESARROLLO DE UN DISPOSITIVO DIGITAL PARA LA MEDICIÓN DE VARIABLES AMBIENTALES UTILIZANDO UN ARREGLO DE COMPUERTAS PROGRAMABLE EN CAMPO

RESUMEN
Este artículo trata sobre el desarrollo e implementación de un dispositivo digital para la medición de variables ambientales, específicamente temperatura en el rango +2°C a +150°C utilizando el sensor de temperatura LM35, cuya salida análoga es digitalizada por medio del conversor análogo-digital ADC0804 y posteriormente procesada por un Arreglo de Compuertas Programable en Campo (FPGA) XC3S200 de un módulo Spartan-3 para ser finalmente visualizado en su respectivo display de siete segmentos de cuatro caracteres.

PALABRAS CLAVES: Electrónica digital, VHDL, dispositivos de lógica programable, diseño digital, FPGA, CPLD.

ABSTRACT
This paper is about the implementation of a digital thermometer in the +2℃ to +150℃ range using the LM35 temperature sensor whose analogue voltage output is digitized by means of the ADC0804 analogue-to-digital converter, processed by a Field-Programmable Gate Array (FPGA) XC3S200 on a Spartan-3 board and finally displayed on its four-character seven-segment display.

KEYWORDS: Digital electronics, VHDL, programmable logic device, digital design, FPGA, CPLD.

JUAN CARLOS OLARTE CORTÉS
Tecnólogo Eléctrico, Economista y Magíster en Administración Económica y Financiera
jcolartec@utp.edu.co

GUILLERMO ROBERTO SOLARTE MARTINEZ
Ingeniero de Sistemas
Profesor Auxiliar
Universidad Tecnológica de Pereira.
roberto@utp.edu.co

JULIO CESAR JARAMILLO R.
Ingeniero Electricista
Universidad Tecnológica de Pereira
Ms.C en Telecomunicaciones
South Bank University, Londres
jaramijc@utp.edu.co

1. INTRODUCCIÓN

En la actualidad se está presenciando una gran revolución en el área del diseño digital que traerá grandes consecuencias en la forma como se diseñan, desarrollan y comercializan dispositivos electrónicos de consumo, de comunicaciones, militares, equipos médicos e industriales entre otros.

Tal revolución fue iniciada por los dispositivos lógicos programables (Programmable Logic Devices o PLDs) que permiten agilizar, mejorar y simplificar el proceso de diseño de hardware.

Los PLDs pueden ser agrupados en categorías de acuerdo a su escala de integración en Arreglos Lógicos Genéricos (GALs), Lógica de Arreglos Programables (PALs), pasando por los Dispositivos de Lógica Programable Simples (SPLDs) y Dispositivos de Lógica Programable Complejos (CPLDs) hasta llegar a los Arreglos de Compuertas Programables en Campo (FPGAs) que tienen el más alto desempeño y la más alta escala de integración, lo que permite la implementación de diseños de alta complejidad. Es por esta última razón que en el diseño del termómetro digital aquí implementado, se eligió la FPGA XC3S200 que viene incorporada en el módulo de desarrollo Spartan-3 que además incluye displays de siete segmentos, pulsadores, interruptores, comunicación serial y puertos digitales de entrada y de salida, facilitando la realización de prototipos.

Se mostrarán en este artículo las estructuras básicas del leguaje VHDL que servirán para la descripción de la sección digital del termómetro en la FPGA. También se mostrará el conversor análogo-digital ADC0804 que es la interface entre la señal análoga suministrada por el sensor de temperatura LM35 y el segmento digital de este proyecto.

Finalmente se hará una comparación entre dos posibles formas de implementación del termómetro digital llamadas *arquitectura1* y *arquitectura2*. La primera de ellas concibiendo el diseño como una sola unidad utilizando descripción comportamental, mientras que en la segunda se utilizará una filosofía estructural pero conservando un estilo comportamental.

2. DESCRIPCIÓN DEL PROYECTO

En el proyecto del termómetro digital se distinguen los siguientes bloques: sensor de temperatura, conversión análoga-digital, FPGA y visualización, incorporados estos dos últimos bloques en el mismo módulo Spartan-3. La figura 1 ilustra el diagrama de bloques mencionado.

Fecha de Recepción: 09 Febrero de 2007
Fecha de Aceptación: 17 Mayo de 2007

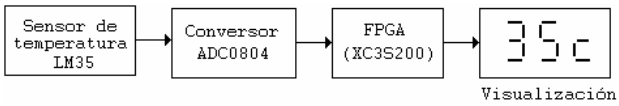

Figura 1. Diagrama de bloques del termómetro digital.

A continuación se describe en detalle cada uno de los bloques constructivos.

2.1 Sensor de temperatura LM35

El LM35 es un sensor de temperatura de propósito general de bajo costo que tiene una salida de voltaje linealmente proporcional a la temperatura circundante en grados Celsius (10mV/°C). Este sensor no requiere calibración externa y puede ser utilizado en el modo de medición básico para temperaturas entre +2°C a +150°C y en modo de medición de rango completo para temperaturas entre -55°C a +150°C. La figura 2 muestra como fue conectado el LM35 en este proyecto para operar en el modo básico de medición de temperaturas.

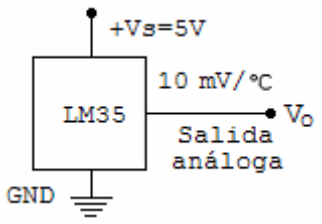

Figura 2. Conexión del sensor LM35 para operar en el rango +2°C a +150°C.

Por lo tanto en el rango de temperatura indicado la salida de voltaje puede describirse mediante la siguiente expresión:

$$Vo = 0.01T \qquad [1]$$

Donde Vo está dado en Voltios y T en grados Celsius.

La señal de salida del sensor de temperatura no requiere ser acondicionada por las razones que se explican a continuación, por lo tanto este sensor puede ser conectado directamente a la entrada de voltaje análogo Vin del ADC0804.

2.2 Conversor análogo-digital ADC0804

El conversor análogo-digital ADC0804 de aproximaciones sucesivas se caracteriza por tener una resolución de 8 bits que permite digitalizar la señal de voltaje suministrada por el sensor de temperatura con un error relativamente bajo para luego ser entregada a la FPGA para su procesamiento.

De acuerdo a lo anterior el código binario B resultante de la conversión se determina de la siguiente forma:

$$B = \frac{256 * Vin}{Vref} \qquad [2]$$

Donde:

B : Código binario resultante de la conversión.
Vin : Voltaje de entrada al conversor.
Vref : Voltaje de referencia del conversor.

La facilidad de ajustar el voltaje de referencia Vref del conversor ADC084 permite simplificar el diseño del termómetro, ya que este se puede fijar a un valor tal que el código binario (salidas DB0 a DB7) sea a su vez la temperatura en grados Celsius. Observe que el valor del voltaje Vref en cuestión es 2.56V ya que si se reemplaza Vo (es decir Vin del conversor ADC) de la ecuación [1] en la ecuación [2] se obtiene que el código binario B es igual a la temperatura T.

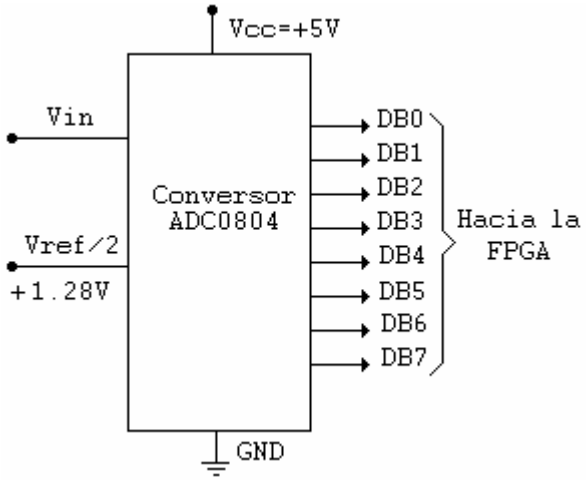

Figura 3. Conversor análogo-digital.

2.3 Lógica programable del proyecto

Como muestra la figura 4, la etapa de lógica programable fue implementada, tanto para la *arquitectura1* como para la *arquitectura2*, en una FPGA XC3S200, la cual hace parte de un módulo Spartan-3. Este dispositivo recibe la información de temperatura proveniente del ADC0804 a través de 8 líneas de entrada llamadas *temperatura* y tiene además una entrada para la señal de reloj *clk* y otra entrada asíncrona *reset* para reiniciar el diseño en cualquier momento.

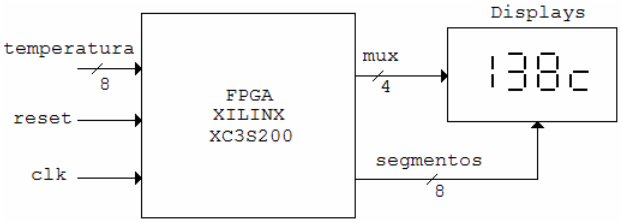

Figura 4. Entidad termómetro.

Debido a que los cuatro caracteres del display del módulo Spartan-3 comparten las líneas de entrada (7 segmentos y el punto) se requieren dos puertos de salida. El primero de ellos es *segmentos* (8 bits) que se utiliza para enviar el dígito del valor de temperatura respectivo y el segundo el *mux* (4 bits), el cual permite seleccionar cual de los cuatro caracteres del display está activo en un momento dado. Por lo tanto, la entidad *termómetro* del diseño en VHDL es:

```
entity termometro is
port    (temperatura: in std_logic_vector(7 downto 0);
         reset: in std_logic;
         clk : in std_logic;
         segmentos: out std_logic_vector(7 downto 0);
         mux: out std_logic_vector(3 downto 0));
end termometro;
```

2.3.1 Primera arquitectura del termómetro

La primera implementación del diseño se hizo con una descripción comportamental, la cual consiste en describir el funcionamiento del termómetro por medio de instrucciones secuenciales dentro de un único proceso llamado *principal* dentro de la *arquitectura1* como se muestra a continuación.

```
architecture arquitectura1 of termometro is
        signal cont: integer range 0 to 400_000;
function selector(digito: integer) return std_logic_vector is
        variable seg: std_logic_vector(7 downto 0);
begin
        case digito is
                when 0 => seg := "10000001";
                when 1 => seg := "11001111";
                when 2 => seg := "10010010";
                when 3 => seg := "10000110";
                when 4 => seg := "11001100";
                when 5 => seg := "10100100";
                when 6 => seg := "11100000";
                when 7 => seg := "10001111";
                when 8 => seg := "10000000";
                when 9 => seg := "10000100";
                when others => null;
        end case;
        return(seg);
 end selector;

begin
principal:process(clk,reset)
        variable aux: integer range 0 to 255;
        variable dig1,dig2,dig3:integer range 0 to 9;
begin
        if reset='1' then
                cont <= 0;
                mux <= "0000";
                segmentos <= "10110000";
        elsif   rising_edge(clk) then
                cont <= cont + 1;
                aux := conv_integer(temperatura);
                dig1:= 0;
                for i in 0 to 9 loop
                        if aux >= 100 then
                                aux := aux - 100;
                                dig1 := dig1 + 1;
                        end if;
                end loop;
                dig2 := 0;
                for j in 0 to 9  loop
                        if aux >=10 then
                                aux  := aux - 10;
                                dig2 := dig2 + 1;
                        end if;
                end loop;
                dig3 := aux;

                case cont is
                when 100_000 =>
                        mux <= "0111";
                        segmentos <= selector(dig1);
                when 200_000 =>
                        mux <= "1011";
                        segmentos <= selector(dig2);
                when 300_000 =>
                        mux <= "1101";
                        segmentos <= selector(dig3);
                when 400_000 =>
                        mux <= "1110";
                        segmentos <= "11110010";
                        cont <= 0;
                when others => null;
                end case;
        end if;
end process principal;
end arquitectura1;
```

Dentro de *arquitectura1* se destaca la función *selector* que funciona de forma similar a un decodificador BCD a 7 segmentos, recibiendo una variable de tipo entero (*digito*) que representa el dígito decimal a visualizar y devolviendo una señal del tipo *std_logic_vector* de 8 bits (7 segmentos y el punto) para encender o apagar los segmentos del display que representan el respectivo dígito.

También sobresale el proceso *principal* que permite que en cada flanco de subida de la señal de reloj *clk* se determinen los tres dígitos de temperatura (variables

dig1, *dig2* y *dig3* que representan las centenas, decenas y unidades respectivamente) por medio de divisiones enteras del valor de temperatura por 100 y 10. Los dígitos hallados luego son mostrados en la posición respectiva cada 2 ms (100 mil pulsos de reloj de 50 MHz) utilizando las líneas de selección *mux*. Una particularidad del proceso es la forma como se realizan las divisiones por 100 y por 10 por medio de restas sucesivas de 100 y 10 para el dig1 y el dig2 respectivamente utilizando un ciclo *for-loop*.

Es importante anotar que el display incluido en el módulo Spartan-3 opera con niveles activos bajos tanto para los segmentos como para las líneas de selección de carácter. Además se dio el siguiente significado a cada una de las señales de los puertos *segmentos* y *mux*.

segmentos

p	a	b	c	d	e	f	g
7	6	5	4	3	2	1	0

mux

centenas	decenas	unidades	°C
3	2	1	0

Figura 5. Significado de las señales de los puertos *segmentos* y *mux*.

2.3.2 Segunda arquitectura del termómetro

En este segundo enfoque de diseño del termómetro digital se dividió el problema en cuatro dispositivos más pequeños interconectados entre sí, los cuales están descritos en los procesos concurrentes *temp*, *secuencial*, *decodificador* y *selector*.

El proceso *temp* describe un dispositivo combinacional que se encarga de recibir la temperatura proveniente del conversor análogo-digital y determinar los tres dígitos decimales a partir de dicho valor. Los tres dígitos hallados (variables *dig1*, *dig2* y *dig3*) dentro del proceso son pasados al resto del circuito por medio de las señales *dig1_s*, *dig2_s* y *dig3_s*.

A continuación se encuentra el proceso *secuencial* que permite reiniciar el termómetro y activar cada 2 ms uno de los cuatro caracteres del display del módulo. Como su nombre lo indica es un proceso secuencial controlado por la señal de reloj *clk* de 50 MHz tomada del oscilador de la misma frecuencia incorporado en el módulo Spartan-3.

```
architecture arquitectura2 of termometro is
        signal cont: integer range 0 to 400_000;
        signal dig1_s,dig2_s,dig3_s:integer range 0 to 9;
        signal dig_mux: integer range 0 to 10;
begin
temp:process(temperatura)
        variable aux:integer range 0 to 255;
        variable dig1,dig2,dig3:integer range 0 to 9;
begin
        aux := conv_integer(temperatura);
        dig1:= 0;
        for i in 0 to 9 loop
                if aux >= 100 then
                        aux := aux - 100;
                        dig1 := dig1 + 1;
                end if;
        end loop;
        dig2 := 0;
        for j in 0 to 9  loop
                if aux >=10 then
                        aux  := aux - 10;
                        dig2 := dig2 + 1;
                end if;
        end loop;
        dig3 := aux;
        dig1_s <= dig1;
        dig2_s <= dig2;
        dig3_s <= dig3;
end process temp;

secuencial: process (clk,reset)
begin
        if reset='1' then
                cont <= 0;
                mux <= "1111";
        elsif       rising_edge(clk) then
                cont <= cont + 1;
        case cont is
                when 100_000 => mux <= "0111";
                when 200_000 => mux <= "1011";
                when 300_000 => mux <= "1101";
                when  400_000  =>  mux  <=  "1110";
                                cont <= 0;
                when others => mux <= mux;
                end case;
        end if;
end process secuencial;
decodificador:process(dig_mux)
begin
        case dig_mux is
                when 0 => segmentos <= "10000001";
                when 1 => segmentos <= "11001111";
                when 2 => segmentos <= "10010010";
                when 3 => segmentos <= "10000110";
                when 4 => segmentos <= "11001100";
                when 5 => segmentos <= "10100100";
                when 6 => segmentos <= "10100000";
                when 7 => segmentos <= "10001111";
                when 8 => segmentos <= "10000000";
                when 9 => segmentos <= "10000100";
                when 10=> segmentos <= "11110010";
                when others=>segmentos<= "10000001";
        end case;
end process decodificador;
```

```
selector: process(dig1_s, dig2_s, dig3_s, mux)
begin
        case mux is
                when "0111" => dig_mux <= dig1_s;
                when "1011" => dig_mux <= dig2_s;
                when "1101" => dig_mux <= dig3_s;
                when "1110" => dig_mux <= 10;
                when others => dig_mux <= 0;
        end case;
end process selector;
end arquitectura2;
```

La visualización de los dígitos de temperatura se hace a través del proceso *decodificador* que toma un dígito de temperatura (señal *dig_mux*) y lo decodifica para encender o apagar sus respectivos segmentos por medio de la señal del mismo nombre. El dígito de entrada al *decodificador* proviene del proceso *selector* que multiplexa las tres señales *dig1_s*, *dig2_s* y *dig3_s* (provenientes del proceso *temperatura*) y la señal constante correspondiente al carater °C.

Figura 6. Módulo Spartan-3

3. CONCLUSIONES Y RECOMENDACIONES

La correcta especificación del rango de valores que pueden tomar las señales y variables enteras de un diseño en VHDL, reduce considerablemente la cantidad de recursos (flip-flops, LUTs, etc.) necesarios para su implementación en una FPGA o cualquier otro dispositivo digital programable.

Con base en los reportes generados al sintetizar la *arquitectura1* y la *arquitectura2* del diseño del termómetro y comparar dichos resultados, se puede concluir que el estilo de descripción (estructural, comportamental, flujo de datos o mixto) utilizado en VHDL juega un papel muy importante en el diseño de cualquier dispositivo lógico, ya que determina la cantidad de recursos utilizados y por ende el desempeño y la máxima velocidad de operación del dispositivo descrito. Sin embargo es difícil saber cual de los estilos es mejor que los demás debido a que esto dependerá de la naturaleza misma del diseño.

Se recomienda resolver cualquier diseño en VHDL utilizando un enfoque descendente (top-down design), es decir, empezando con la descripción global del dispositivo (entidad) y luego dividirlo en dispositivos más pequeños que luego son interconectados para realizar la tarea deseada. Lo anterior es con el fin de poder detectar posibles fallas más fácilmente y hacer el proceso de desarrollo más rápido diseñando código reutilizable.

4. BIBLIOGRAFÍA

[1] FLOYD, Thomas L. Fundamentos de Electrónica Digital, 811 páginas, Editorial Limusa, Méjico, 2001.

[2] FLOYD, Thomas L. Digital Fundamentals with VHDL, 960 páginas, Prentice Hall, ISBN 0130995274, 2002

[3] TERÉS Lluís, VHDL Lenguaje Estàndar de Diseño Electrónico, 495 Páginas, Mc. Graw Hill, España, 1998.

[4] WAKERLY, John F. Digital Design: Principles and Practices, Cuarta edición, 895 páginas, Pearson Education, Estados Unidos, 2006

[5] ZWOLINSKI, Mark, Digital System Design with VHDL, 361 Páginas, Prentice Hall, Reino Unido, 2004.