66.17 - Sistemas Digitales
Trabajo Práctico 2 - Voltímetro digital con salida VGA

Andrew Parlane

14 de mayo de 2018

Índice

Índice de figuras

1. Introducción

El objetivo de este trabajo es a diseñar, simular y implementar en FPGA un circuito digital que mide una tensión analógica y mostrarle por una pantalla VGA. El conversor A/D debe usar una entrada LVDS como un comparador para implementar un Sigma-Delta ADC.

El circuito digital debe funcionar con una Altera DE2 placa de desarrollo que tiene un cyclone II 2C35 FPGA.

2. Herramientas

Las herramientas que usé para este trabajo son:

- Questa Sim v10.2c
- Quartus II v13.0sp1
- GNU Make v4.2.1
- FTDI Font conversion utility v0.4 http://www.ftdichip.com/Support/Utilities.htm
- SRecord v1.63 http://srecord.sourceforge.net
- VGA Simulator https://ericeastwood.com/lab/vga-simulator/

3. Implementación

Los componentes principales de mi diseño son:

- Contador Binario que cuenta desde cero hasta un parámetro MAX.
- Contador BCD que cuenta desde cero hasta 99...9 con un parámetro de CIFRAS.
- ADC Sigma-Delta que usa la entrada diferencial y una salida para obtener muestras de la señal.
- VGA que genera las señales de VGA.
- ADV7123 que genera las señales para uso con el ADV7123 VGA DAC.
- Un carácter ROM que contiene las mapas de pixeles por los caracteres ASCII desde ' ' hasta '~'.
- TP2 que junta todo el diseño.

También implementé unos paquetes con funcionalidad útil:

- Utils para funciones genéricos y útil.
- Type para tipos comunes.
- VGA Timings para los timings de VGA para diferentes resoluciones.
- Char ROM para un enum de los caracteres.

Los recursos de la placa DE2 que uso son:

- Tres pins GPIO para el conversor ADC.
- Un reloj de 50MHz.
- Un botón para la señal reset.
- El VGA DAC y el puerto VGA.

No voy a detallar los dos componentes de contadores aquí porque son casi el mismo del TP1. Tampoco voy a detallar los paquetes porque no son tan relevante al diseño.

3.1. ADC

Ese componente es Un ADC Sigma-Delta con un filtro digital low-pass. Hay un contador BCD de cinco cifras que cuenta solo cuando la entrada diferencial es un '1'. Hay un segundo contador que cuenta hasta 33001. Cuando llega a 33001 captura el valor del contador BCD, esto es la muestra. Usando un reloj de 50MHz obtenemos una muestra cada 33002 ticks = 1.5KHz.

3.1.1. Señales y Parámetros

Nombre	Tipo	Bits	Descripción
Entradas			
clk	std_ulogic	1	El reloj.
rst	std_ulogic	1	Reset asíncrono.
dInDiff	std_ulogic	1	La salida del comparador LVDS.
Salidas			
dOut	std_ulogic	1	dInDiff retardado un tick.
resultado	unsignedArray	3 * 4	La última muestra en BCD de 3 cifras.

3.1.2. Diagrama de Bloques

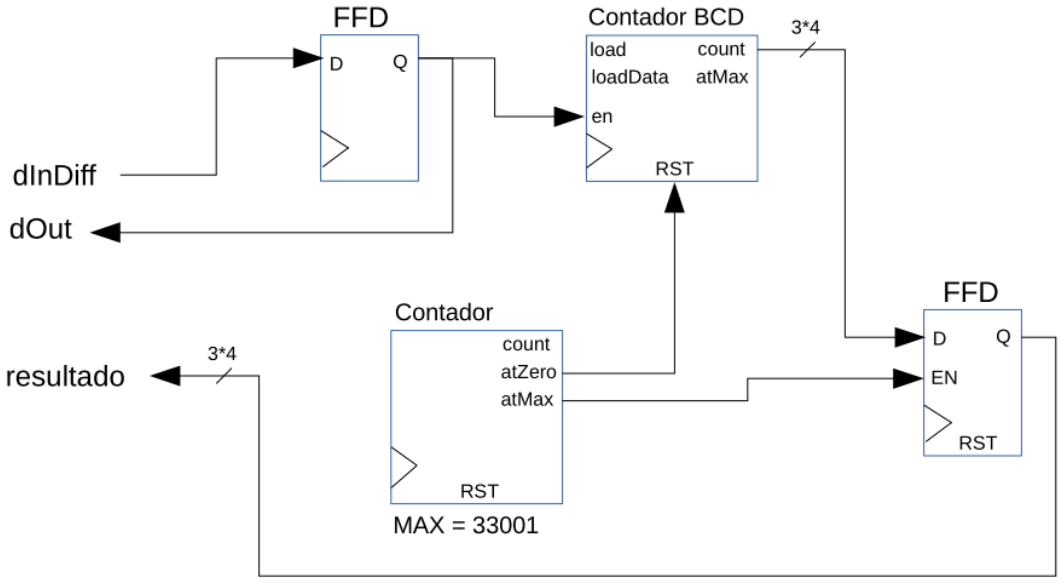

Figura 1: Diagrama de Bloques de componente adc.

3.2. VGA

Ese componente genera las señales VGA hsync, vsync, un par de señales pixelX y pixelY que dicen que pixel enviamos en el siguiente tick del reloj y una señal inActive que dice si está en la región activo de la pantalla. La resolución y los timings están parámetros.

3.2.1. Señales y Parámetros

Nombre	Tipo	Bits	Descripción
Parámetros			
H_ACTIVE	natural		La resolución horizontal en pixeles.
H_FRONT_PORCH	natural		Duración del front porch horizontal en pixeles.
H_SYNC	natural		Duración del pulso sync horizontal en pixeles.
H_BACK_PORCH	natural		Duración del back porch horizontal en pixeles.
V_ACTIVE	natural		La resolución vertical en líneas.
V_FRONT_PORCH	natural		Duración del front porch vertical en líneas.
V_SYNC	natural		Duración del pulso sync vertical en líneas.
V_BACK_PORCH	natural		Duración del back porch vertical en líneas.
Entradas			
clk	std_ulogic	1	El reloj.
rst	std_ulogic	1	Reset asíncrono.
Salidas			
pixelX	unsigned	X_WIDTH	Número de pixel activo horizontal que enviamos siguiente.
pixelY	unsigned	Y_WIDTH	Número de pixel activo vertical que enviamos siguiente.
inActive	std_ulogic	1	Ese pixel es en la región activo.
nHSync	std_ulogic	1	La señal hsync (activo bajo).
nVSync	std_ulogic	1	La señal vsync (activo bajo).

Donde X_WIDTH es el mínimo número de bits necesarios para almacenar H_ACTIVO, y Y_WIDTH es el mínimo número de bits necesarios para almacenar V_ACTIVO.

3.2.2. Diagrama de Bloques

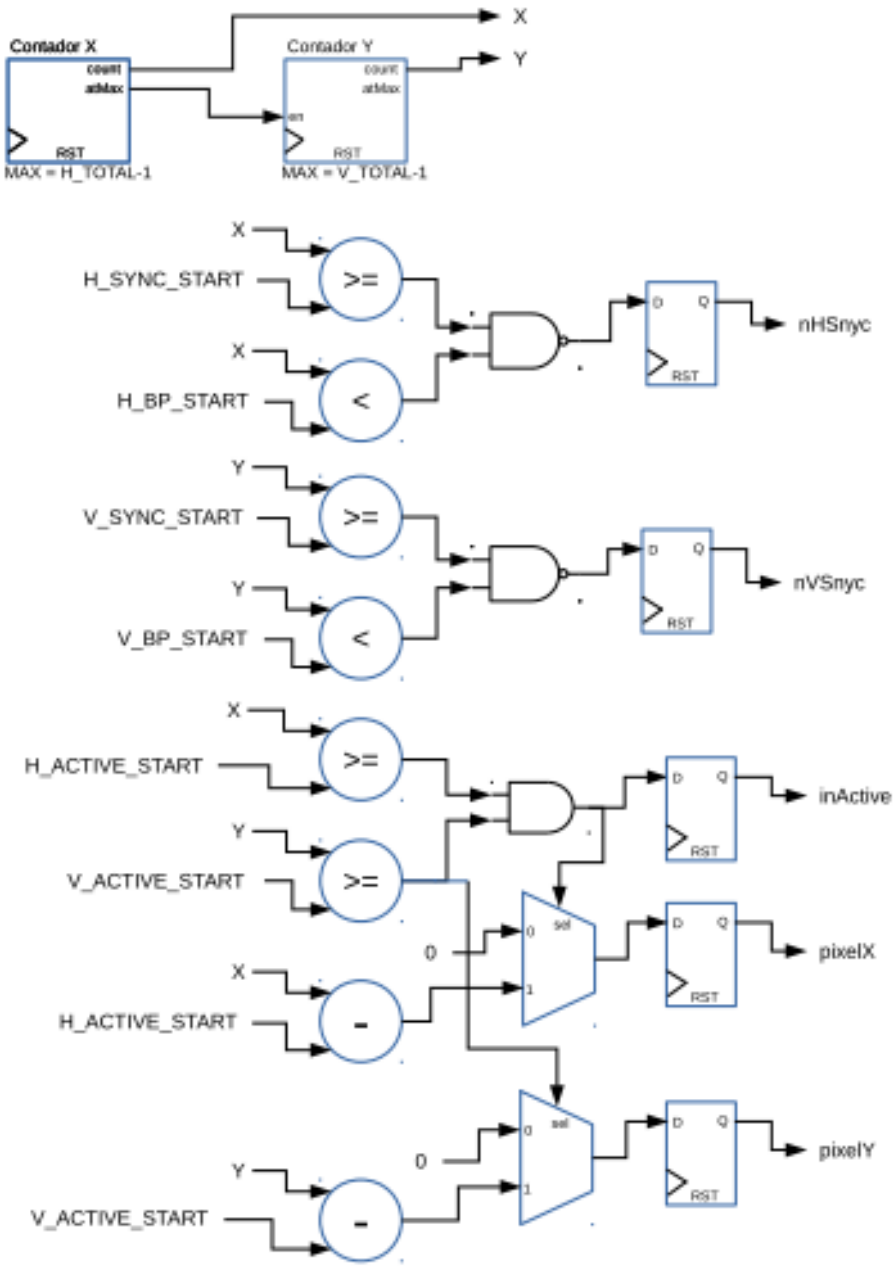

Figura 2: Diagrama de Bloques de componente vga.

3.3. ADV7123

Ese componente es un wrapper del componente VGA para funcionar con el VGA DAC ADV7123.

3.3.1. Señales y Parámetros

Tiene todos los mismas señales de VGA más:

Nombre	Tipo	Bits	Descripción
Entradas			
rIn	std_ulogic_vector	10	El valor rojo para el pixel pixelX.
gIn	std_ulogic_vector	10	El valor verde para el pixel pixelX.
bIn	std_ulogic_vector	10	El valor azul para el pixel pixelX.
Salidas			
clkOut	std_ulogic	1	El reloj hasta el DAC.
rOut	std_ulogic_vector	10	El valor rojo alineado a las otras señales.
gOut	std_ulogic_vector	10	El valor verde alineado a las otras señales.
bOut	std_ulogic_vector	10	El valor azul alineado a las otras señales.
nBlank	std_ulogic	1	Igual a inActive.
nSync	std_ulogic	1	Activo (bajo) cuando hay un pulso de hsync o vsync.

3.4. ROM de caracteres

Decidí a construir mi propio ROM de caracteres usando el font Ubuntu Mono. Usé la herramienta *fnt_cvt* para convertir el .ttf hasta la data de pixeles crudo con 1 bit cada pixel. Después usé la herramienta *srec_cat* para convertir el binario hasta un .mif. Ese data es almacenado en el block RAM del FPGA usando un *Altera Mega Wizzard component*. Cada palabra del ROM es 16 bits que es el ancho de un carácter. Así cada carácter comienza con la fila igual al número de carácter multiplicado por la altura de la carácter.

3.4.1. Señales y Parámetros

Nombre	Tipo	Bits	Descripción
Entradas			
clk	std_ulogic	1	El reloj.
rst	std_ulogic	1	Reset asíncrono.
char	charRomCharacter	1	El carácter querido.
offX	unsigned	4	La pixel querido en la fila corriente.
offY	unsigned	5	La fila querido en la carácter corriente.
Salidas			
px	std_ulogic	1	El pixel querido.

3.5. TP2

Ese componente es el más alto que junta todo los demás. Uso una resolución de 800x600 divido en bloques de 32*16 pixeles, usando los bits mayores del pixelX y pixelY como índices al bloque, y los menores para el offset en la cuadra. A final hay un poco lógica a especificar que carácter necesita dibujar en cada bloque.

3.5.1. Señales y Parámetros

Nombre	Tipo	Bits	Descripción
Entradas			
CLOCK_50	std_ulogic	1	El reloj de 50MHz.
KEY	std_ulogic_vector	1	Una botón para el reset.
DATA_VOLT_IN_DIFF	std_ulogic	1	La entrada LVDS.
Salidas			
DATA_VOLT_OUT	std_ulogic	1	La salida al red RC.
VGA_R	std_ulogic_vector	10	El componente rojo del pixel.
VGA_G	std_ulogic_vector	10	El componente verde del pixel.
VGA_B	std_ulogic_vector	10	El componente azul del pixel.
VGA_CLK	std_ulogic	1	El reloj de pixel.
VGA_BLANK	std_ulogic	1	Activo (bajo) si la pixel corriente es en la región blanking.
VGA_SYNC	std_ulogic	1	Activo (bajo) si la pixel corriente es en una región de SYNC.
VGA_HSYNC	std_ulogic	1	Activo (bajo) si la pixel corriente es en la región de HSYNC.
VGA_VSYNC	std_ulogic	1	Activo (bajo) si la pixel corriente es en la región de VSYNC.

3.5.2. Diagrama de Bloques

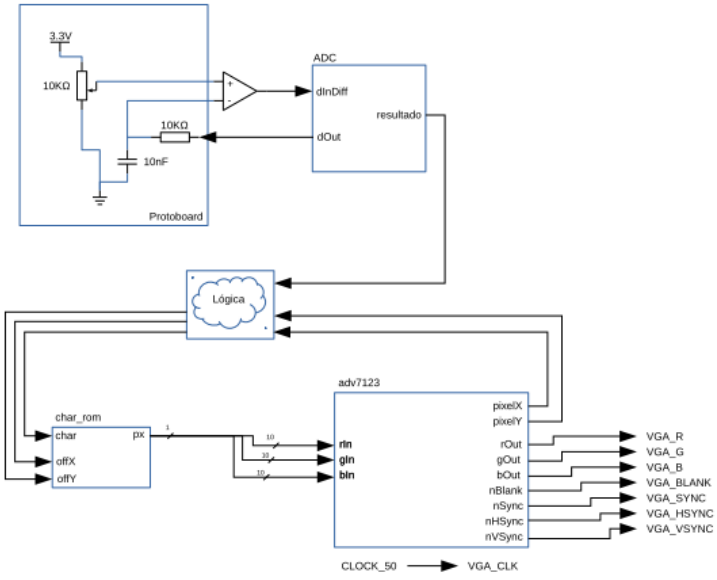

Figura 3: Diagrama de Bloques de componente tp2.

4. Simulación y Verificación

Escribí varios bancas de pruebas para probar la funcionalidad usando asserts, inspección de las ondas y los resultados. Por el componente ADV7123 usé SystemVerilog con un *module*, un *program* y el comando *bind* para conectar el program a mi componented de VHDL. En el *program* tengo algunos asserts y capturo las señales hsync, vsync, r, g, b, y les escribo a un archivo texto. Ese archivo puedo usar con el VGA Simulator.

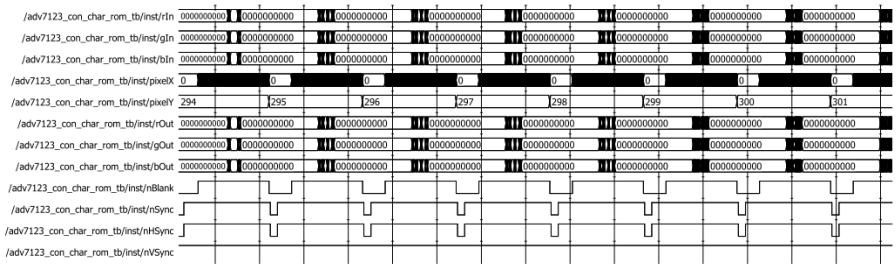

Figura 4: Ondas de ADV7123.

Figura 5: Resultado del banco de prueba adv7123_con_char_rom.

5. Síntesis

He usado Quartus II para sintetizar el diseño.

5.1. Resumen de síntesis

Ítem	Utilizado	Disponible	Porcentaje Utilizado
Logic Elements	266	33,216	<1 %
Registers	148	33,216	<1 %
Bits de memoria	45,600	483,840	9 %
Global Clocks	1	16	6 %

TimeQuest Timing Analyzer me informe que el frecuencia máxima de reloj a la que está operable el circuito es 81.6MHz.

6. Código

Todo el código es disponible en mi github: https://github.com/andrewparlane/fiuba6617/tree/master/TP2.