Codec de Audio en lenguaje VHDL

Informe de trabajo práctico

Jorge Salvador Muñoz

(jorgesmunoz@gmail.com)

12/10/2020

versión A

Historial de cambios

Versión	Fecha	Descripción	Autor	Revisores
A	11/04/2020	Versión Original	Jorge Salvador Muñoz	

1. Introducción

1.1. Propósito

1. En el presente informe se detalla el desarrollo e implementación de un codec de audio en lenguaje VHDL para el kit DE2-115 de Altera.
2. Corresponde al trabajo práctico desarrollado para la materia de Circuitos Lógicos Programables de la CESE.

1.2. Alcance

1. El alcance del desarrollo consistió en implementar un bloque de hardware digital descrito en lenguaje VHDL con el fin de escuchar, a la salida del codec de audio, un zumbido de baja frecuencia perceptible al oído humano.

2. Recursos utilizados

2.1. Kit DE2-115 de Altera

1. Como se mencionó en la introducción, la implementación del desarrollo se realizó sobre un kit de Altera que contiene una FPGA Cyclone IV.
2. A continuación, en la **Imagen 1** puede observarse la FGPA y los periféricos que posee el kit, entre los cuales se encuentra el codec de audio utilizado (WM8731).

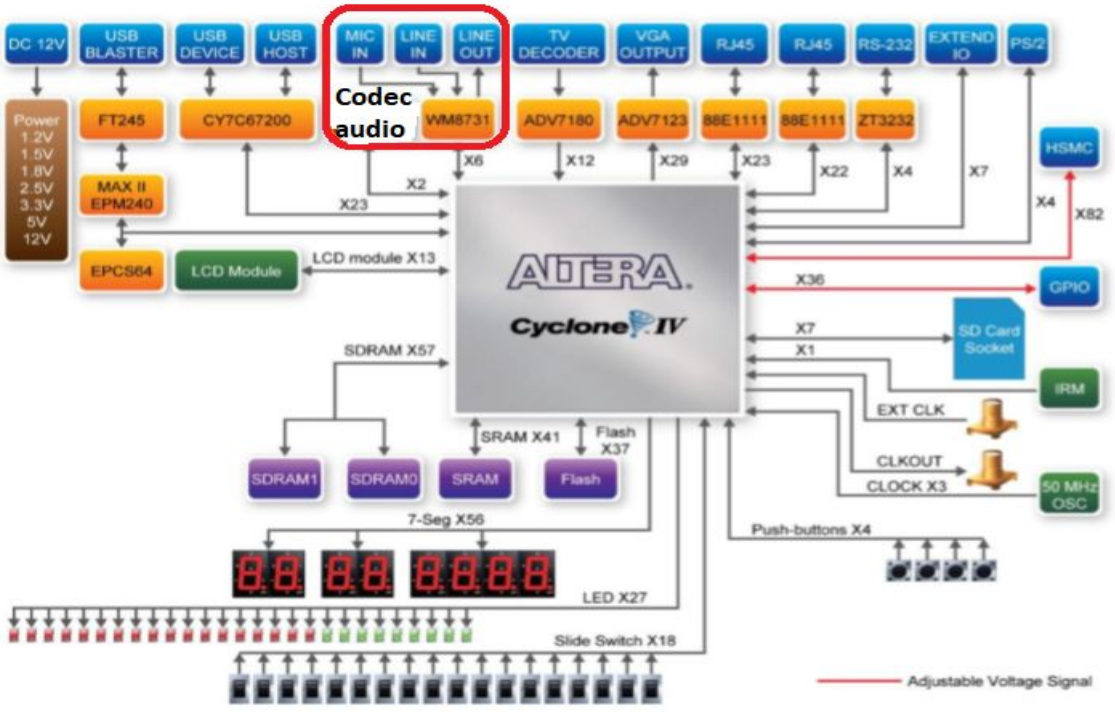

Imagen 1

2.2. Codec de audio WM8731

1. El kit DE2-115 permite obtener una señal de audio de 24 bits a través del code de audio Wolfson WM8731.
2. Características principales:
   - ❏ Codificador / decodificador.
   - ❏ Posee entrada de micrófono, entrada de audio y salida de audio.
   - ❏ Frecuencia de muestreo ajustable entre 8kHz 96 kHz.
   - ❏ Posee una interfaz de control a través del protocolo I2C.
   - ❏ Posee una interfaz digital de audio para procesamiento de señales.
3. Diagrama en bloques del codec de audio:
   - ❏ A continuación, se muestra el diagrama en bloque del codec donde se encuentran señaladas la interfaz digital y de control. Estas interfaces se conectan a la FGPA a través de los pines correspondientes.

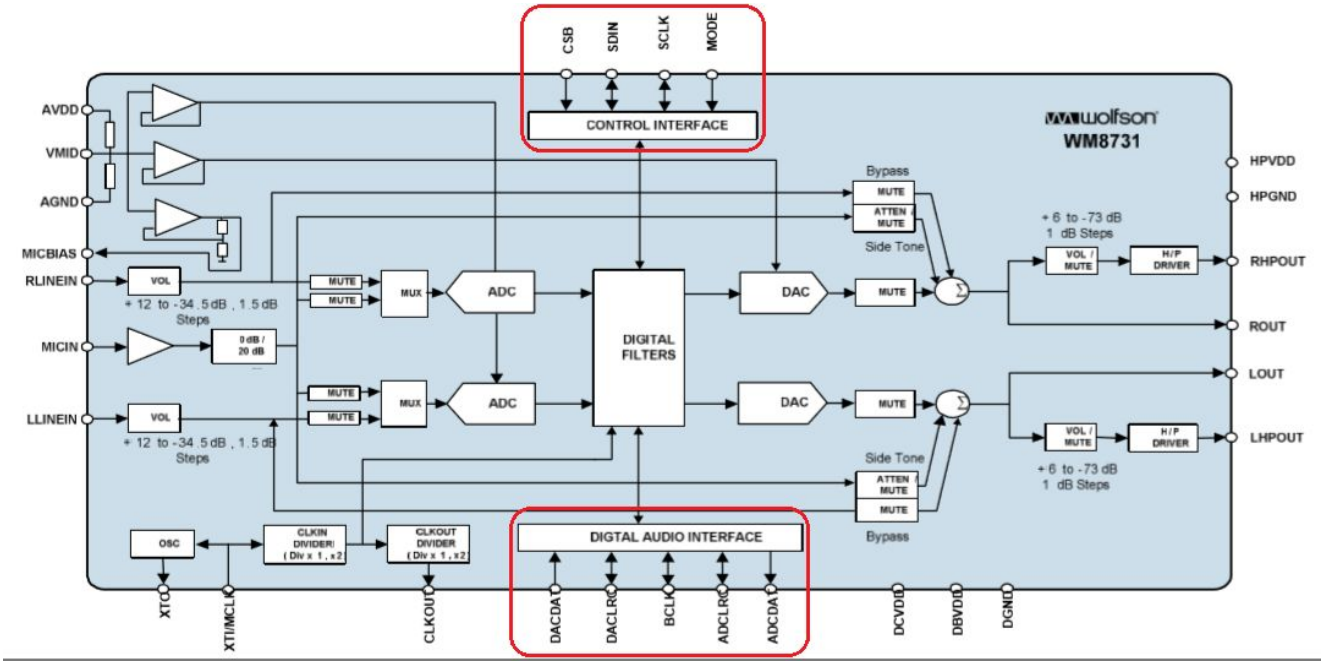

Imagen 2

4. Conexión de interfaz digital de procesamiento de audio con la FGPA.
   - ❏ En la **Imagen 3** se muestra de forma más detallada la conexión entre la interfaz digital de audio y la FPGA.

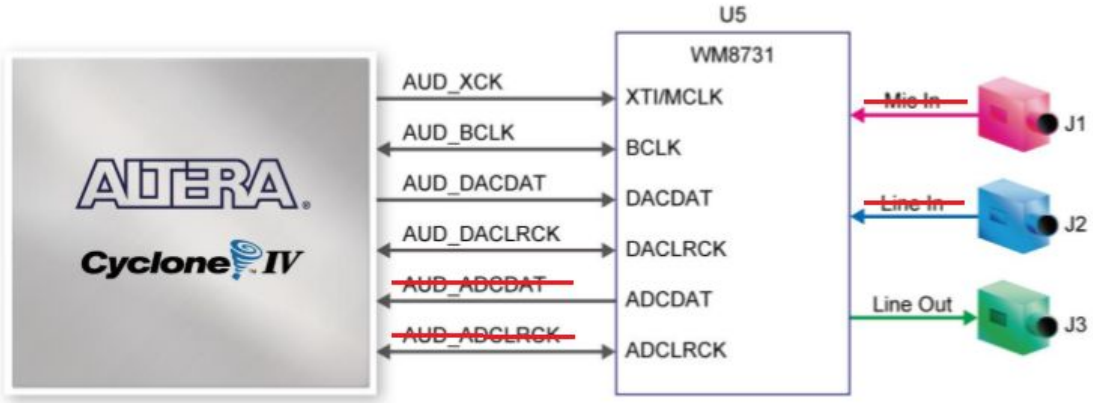

Imagen 3

- ❏ Detalle de cada pin:
  - ❏ AUD_XCK: Master clock.
  - ❏ AUD_BCLK: Digital audio bit clock.
  - ❏ AUD_DACDAT: DAC Digital audio data input.
  - ❏ AUD_DACLRCK: DAC Sample Rate Left/Right Clock.
  - ❏ AUD_ADCDAT: ADC Digital Audio Data Output (no se usa).
  - ❏ AUD_ADCLRCK: ADC Sample Rate Left/Right Clock (no se usa).

5. Registros:
   - ❏ En la siguiente imagen se muestran los registros a configurar de acuerdo a las siguientes consideraciones:
     - ❏ Sample Rate: 48 kHz.
     - ❏ Señal de salida: 16 bits.

REGISTER	BIT[8]	BIT[7]	BIT[6]	BIT[5]	BIT[4]	BIT[3]	BIT[2]	BIT[1]	BIT[0]	DEFAULT
R0 (00h) Left Line In	LRINBOTH	LINMUTE	0	0	LINVOL[4:0]					0_1001_0111
R1 (01h) Right Line In	RLINBOTH	RINMUTE	0	0	RINVOL[4:0]					0_1001_0111
R2 (02h) Left Headphone Out	LRHPBOTH	LZCEN	LHPVOL[6:0]							0_0111_1001
R1 (01h) Right Headphone Out	RLHPBOTH	RZCEN	RHPVOL[6:0]							0_0111_1001
R4 (04h) Analogue Audio Path Control	0	SIDEATT[1:0]		SIDETONE	DACSEL	BYPASS	INSEL	MUTEMIC	MICBOOST	0_0000_1010
R5 (05h) Digital Audio Path Control	0	0	0	0	HPOR	DACMU	DEEMPH[1:0] 48 Khz		ADCHPD	0_0000_1000
R6 (06h) Power Down Control	0	POWEROFF	CLKOUTPD	OSCPD	OUTPD	DACPD	ADCPD	MICPD	LINEINPD	0_1001_1111
R7 (07h) Digital Audio Interface Format	0	BCLKINV	MS	LRSWAP	LRP	IWL[1:0] 16 bit		FORMAT[1:0]		0_1001_1111
R8 (08h) Sampling Control	0	CLKODIV2	CLKIDIV2	SR[3:0]				BOSR	USB/ NORMAL	0_0000_0000
R9 (09h) Active Control	0	0	0	0	0	0	0	0	Active	0_0000_0000
R15 (0Fh) Reset	RESET[8:0]									not reset

Imagen 4

6. Sample Rate.
   - ❏ Se selecciona sample rate de 48kHz configurando el registro R7 como se muestra a en la **Imagen 5**:

SAMPLING RATE		MCLK FREQUENCY	SAMPLE RATE REGISTER SETTINGS					DIGITAL FILTER TYPE
ADC	DAC							
kHz	kHz	MHz	BOSR	SR3	SR2	SR1	SR0	
48	48	12.288	0 (256fs)	0	0	0	0	1
		18.432	1 (384fs)	0	0	0	0	

Imagen 5

   - ❏ De la imagen anterior puede observarse que, para generar una frecuencia de muestreo (sample rate) de 48kHz, es necesario disponer de un clock de 18,432 MHz.

3. Bloques descritos

3.1. Componentes VHDL

1. Los componentes desarrollados en lenguaje VHDL son los detallados a continuación de acuerdo a la jerárquica de cada entidad:
   - ❏ **top_entity**: entidad principal del desarrollo.
     - ❏ **clock_generator**: genera clock de 18,432 Mhz para sample rate.
     - ❏ **i2c_audio**: entidad que contiene los componentes de control y de configuración de registros.
       - ❏ **i2c_audio_config**: configura registros del codec WM8731.
       - ❏ **i2c_controller**: para interfaz de control con protocolo I2C.
     - ❏ **audio_dac**: para interfaz digital de audio. Genera zumbido de baja frecuencia para ser escuchado por la línea de salida.
2. Para un mayor nivel de detalle, en la siguiente imagen se muestra el RTL final con todos los componentes mencionados en el punto anterior.

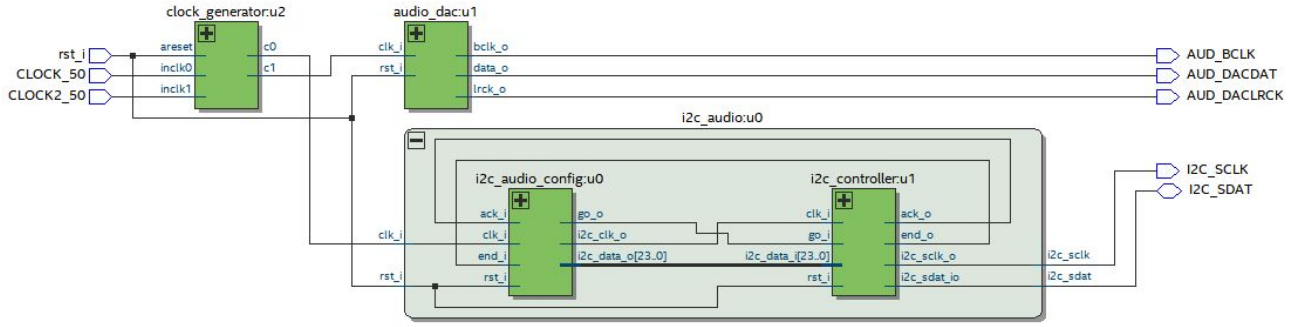

Imagen 6

3. En el presente punto se detalla cada uno de los componentes sintetizado junto con la simulación respectiva.

   ❏ **clock_generator**: Para obtener la frecuencia de clock de 18,432 Mhz se utiliza la IP ALT_PLL proporcionada por el fabricante a través del software Quartus Prime.

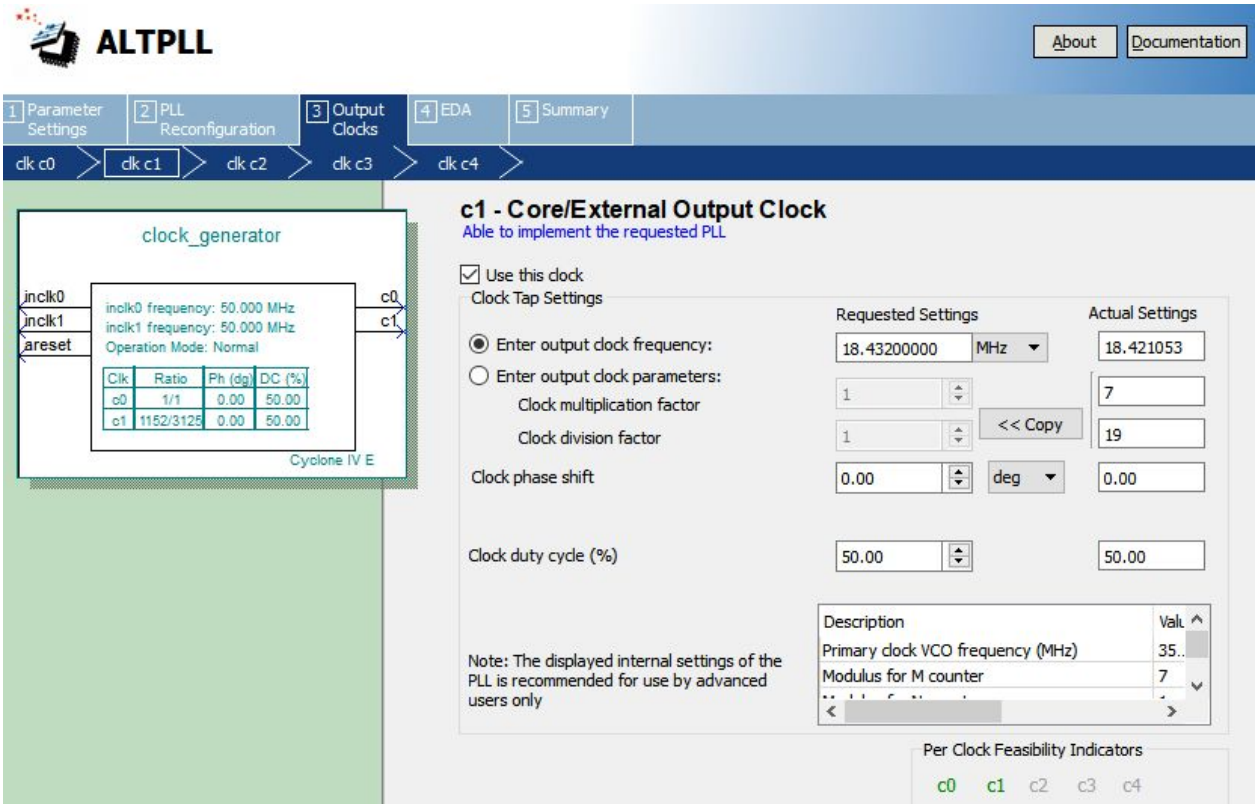

Imagen 7

   ❏ **i2c_audio_config**: Componente que configura los registros del codec de audio WM8731.

      ❏ Son 10 registros principales: se debe enviar la dirección del codec (0x34) antes de enviar el dato de configuración de cada registro.

      ❏ A continuación se muestra la **Imagen 8** con la configuración de cada registro en hexadecimal.

```
with reg_counter select
    stream_data <=  x"34001A" when 1,
                    x"34021A" when 2,
                    x"34047B" when 3,
                    x"34067B" when 4,
                    x"3408F8" when 5,
                    x"340A06" when 6,
                    x"340C00" when 7,
                    x"340E01" when 8,
                    x"341002" when 9,
                    x"341201" when 10,
                    x"340000" when others;
```

Imagen 8

- **i2c_controller**: De acuerdo a la hoja del fabricante, la interfaz de control debe realizarse a través del protocolo I2C (Address: 0x34). Para ello, se genera un componente en VHDL que cumpla con los requerimientos especificados de acuerdo a las condiciones de “Start”, “Stop” y teniendo en cuenta que el codec es solo de lectura si está en modo Slave.
  - En la siguiente imagen puede observarse las condiciones que debe cumplir el protocolo mencionado:

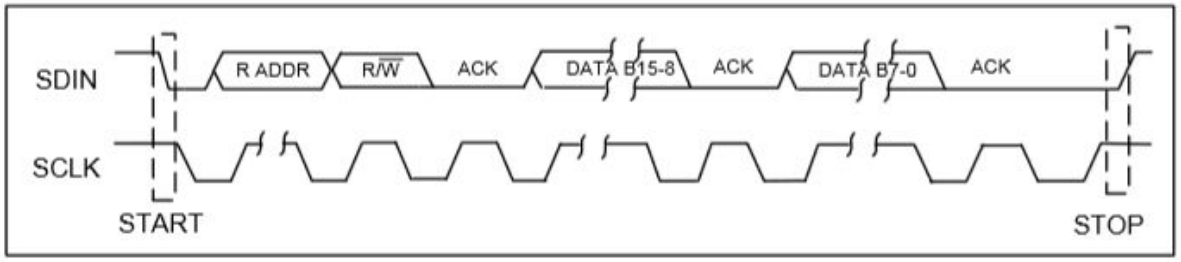

Imagen 9

  - Simulación del componente:

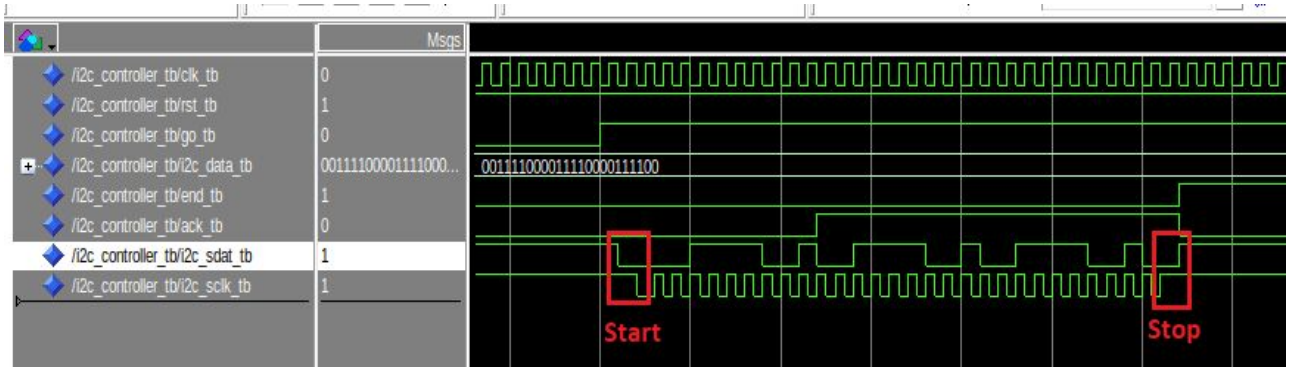

Imagen 10

- **i2c_audio**: Instancia que engloba a los componentes **i2c_controller** e **i2c_audio_config.**
  - En la siguiente imagen se observa el RTL con los componentes mencionados:

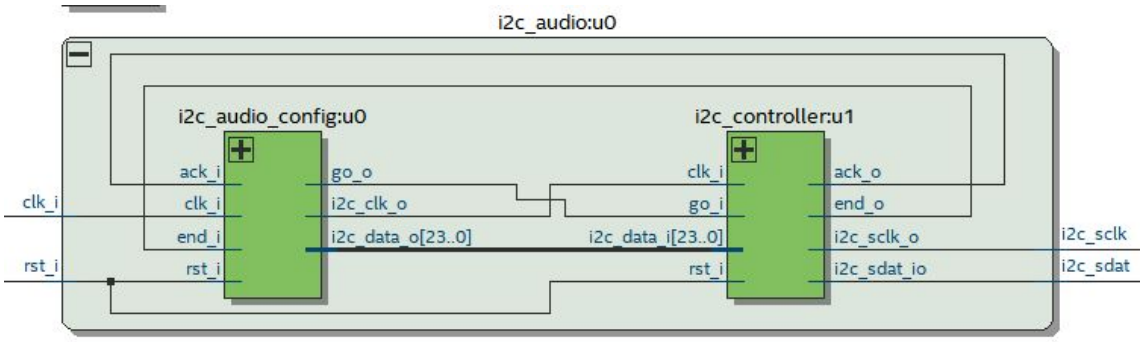

Imagen 11

  - A continuación, se muestra la simulación del componente **i2c_audio**:

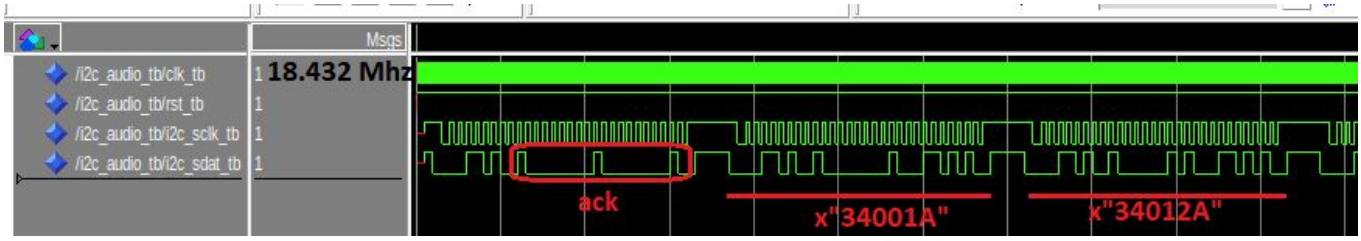

Imagen 12

- **audio_dac**:
  - Componente que genera:
    - Clock BCLK para sincronizar los 16 bits por canal (derecho e izquierdo).
    - Generación de frecuencia de muestreo (sample rate) de 48kHz (DACLRC).
    - Señal de audio de salida (zumbido de baja frecuencia).
  - El componente tiene una señal de entrada correspondiente al clock de 18,432 Mhz para la obtención del clock BCLK y DACLRC. Estas señales deben estar sincronizadas de acuerdo a lo especificado por el fabricante se obtienen a través de dos procesos descritos en VHDL como se muestra en las imágenes a continuación:

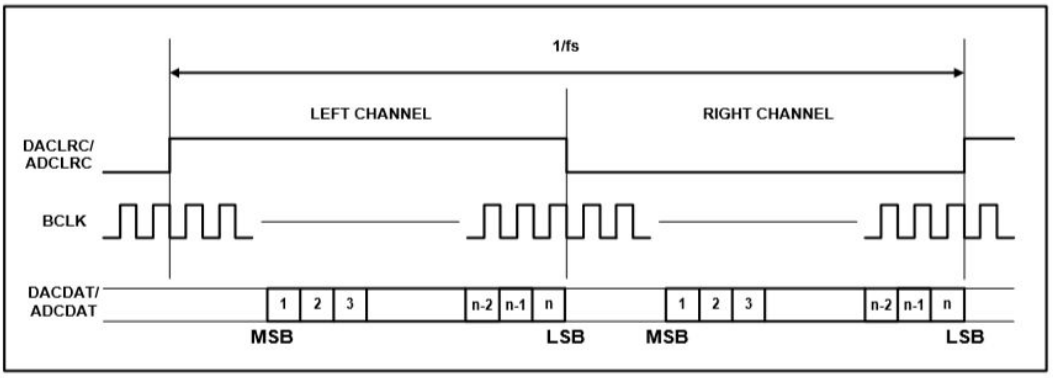

Imagen 13

```vhdl
-- Proceso para generacion de bit clock (BCLK)
bclk_gen: process(clk_i) is
    variable bclk_div: natural;
    begin
        if falling_edge(clk_i) then
            if rst_i = '0' then
                bclk_div := 0;
                bclk_s <= '0';
            elsif (bclk_div >= CLK_REF / (SAMPLE_RATE*DATA_WIDTH*CHANNEL*2) - 1)
                bclk_div := 0;
                bclk_s <= not bclk_s;
            else
                bclk_div := bclk_div + 1;
            end if;
        end if;
    end process;
    bclk_o <= bclk_s;

-- Proceso para generacion de clock DAC (DACLRCK)
lrck_gen: process(clk_i) is
    variable lrck_div: natural;
    begin
        if falling_edge(clk_i) then
            if rst_i = '0' then
                lrck_div := 0;
                lrck_s <= '0';
            elsif (lrck_div >= CLK_REF / (SAMPLE_RATE*2) - 1) then
                lrck_div := 0;
                lrck_s <= not lrck_s;
            else
                lrck_div := lrck_div + 1;
            end if;
        end if;
    end process;
    lrck_o <= lrck_s;
```

BCLK de 16 bit por canal

48 Khz

Imagen 14

- Simulación del componente:

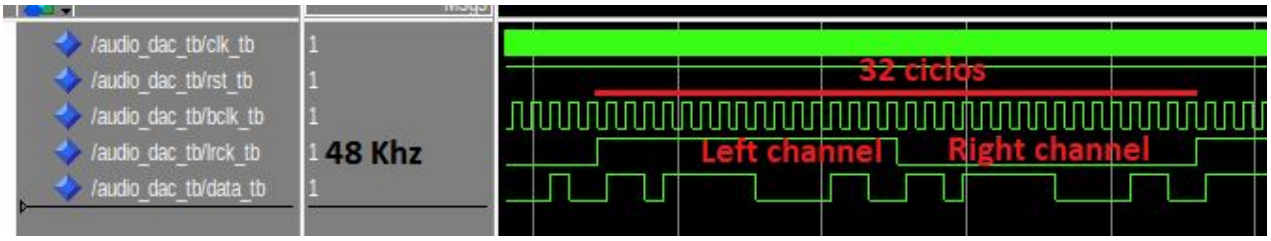

Imagen 15

- **top_entity**: Entidad principal del sistema. En las siguientes imagenes se muestran dos simulaciones en las que se muestran todas las señales de la interfaz de control y la interfaz digital de audio.

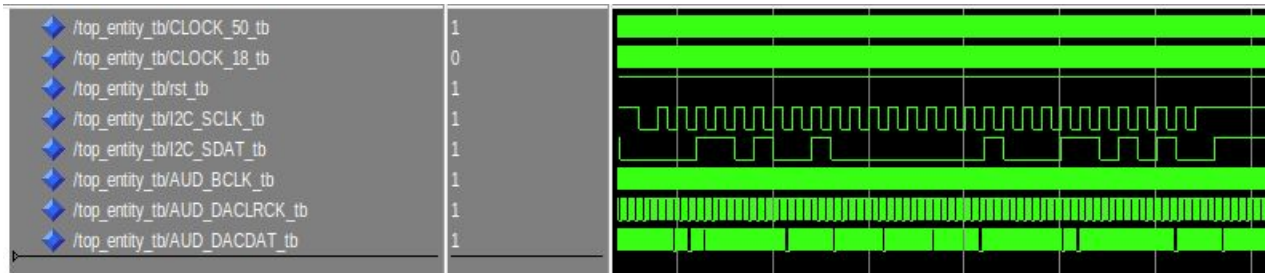

Imagen 16

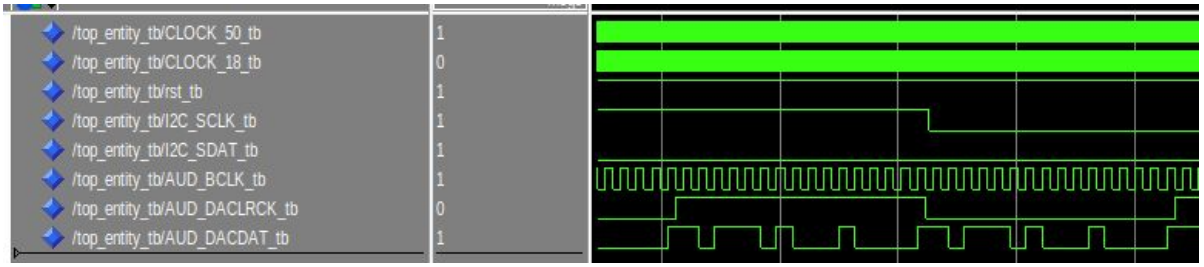

Imagen 17

4. Uso de recursos de la FGPA

Flow Summary

<<Filter>>

Flow Status	Successful - Tue Oct 20 16:25:41 2020
Quartus Prime Version	18.0.0 Build 614 04/24/2018 SJ Lite Edition
Revision Name	top_entity
Top-level Entity Name	top_entity
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	282 / 114,480 (< 1 %)
Total registers	144
Total pins	8 / 529 (2 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	1 / 4 (25 %)

Imagen 18