

Diseño de un sistema de adquisición de datos de un micrófono utilizando una FPGA

Experimental III: Introducción a la Microfabricación y FPGA - Instituto Balseiro

Mauricio Tosi

Diciembre de 2013

Resumen

Desarrollamos un sistema de adquisición de datos de un micrófono utilizando una placa FPGA y una interface serie para comunicarnos con la computadora, verificamos su funcionamiento y concluimos que debido a la velocidad acotada de transmisión de datos del protocolo RS232, solo pudimos almacenar aproximadamente 1,6 seg de audio.

1. Introducción

Una FPGA (del inglés *Field Programe Gate Array*) es un dispositivo lógico que contiene un arreglo bidimensional de celdas lógicas y estructuras programables como puede verse en la Fig. (1). Una celda lógica puede ser configurada (programada) para realizar una función simple, y un interruptor programable se puede modificar para proveer interconexión entre las celdas lógicas. Un diseño particular puede ser implementado al especificar la función de cada celda lógica y determinando selectivamente la conexiones de cada interruptor programable. Una vez que el diseño y conexión esta completa es posible utilizar un simple cable adaptador para descargar la celda lógica deseada y la configuración de interruptores en el dispositivo FPGA y obtener el circuito personalizado. Dado que este proceso puede realizarse "in the field."^{en} lugar de venir configurado de fabrica, es que el dispositivo recibe el nombre de "field programmable". En el marco del curso Experimental III - Introducción a la Microfabricación y FPGA, en este trabajo desarrollamos un sistema de adquisición de datos de un micrófono a la computadora utilizando una FPGA a través de una interfase serie.

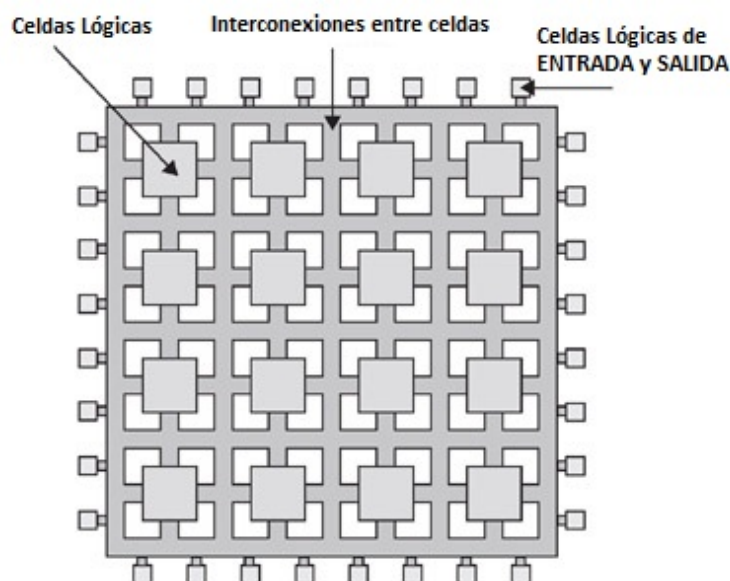


Figura 1: Estructura interna de una FPGA

2. Descripción Funcional

Se utilizó una placa de desarrollo Nexys 3 basada en la FPGA Spartan 6 de Xilinx y el micrófono PmodMIC de Digilent. Este último está compuesto por un micrófono y un conversor analógico/digital de 12 bits. La placa de desarrollo cuenta con un puerto USB-UART que permitió la comunicación con los periféricos externos (salida de datos a la computadora). En la Fig. (2) se muestra el esquema de conexión PmodMIC-FPGA y FPGA-PC a través del puerto USB-UART.

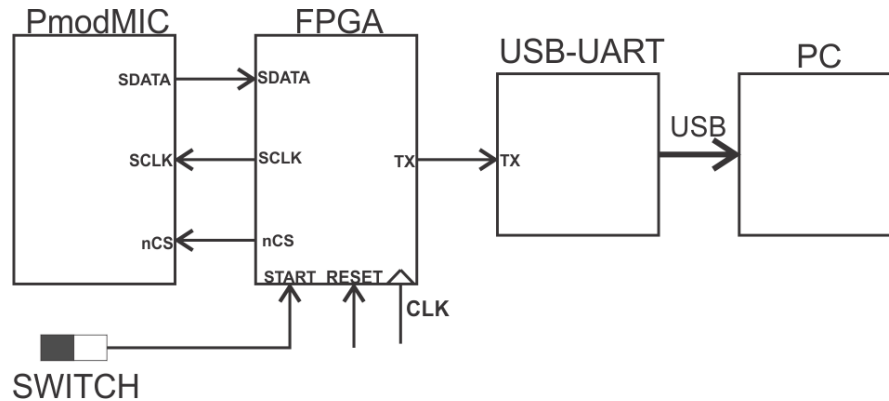


Figura 2: Esquema general de conexión entre el micrófono, la FPGA y la computadora

La entrada SDATA a la FPGA es la salida de datos serie del ADC del PmodMIC, el cual necesita para funcionar un clock de 12,5 MHz (SCLK), generado en la placa. La conexión nCS habilita una conversión del ADC cuando tiene un 0 lógico a la entrada.

La salida TX de la placa transmite los datos serie a la computadora (mediante el puerto USB-UART) a través del protocolo RS232.

La adquisición de datos comienza al activarse el SWITCH que habilita la entrada START.

2.1. Módulos VHDL

En la Fig. (3) se muestran los módulos VHDL que se programaron en la FPGA.

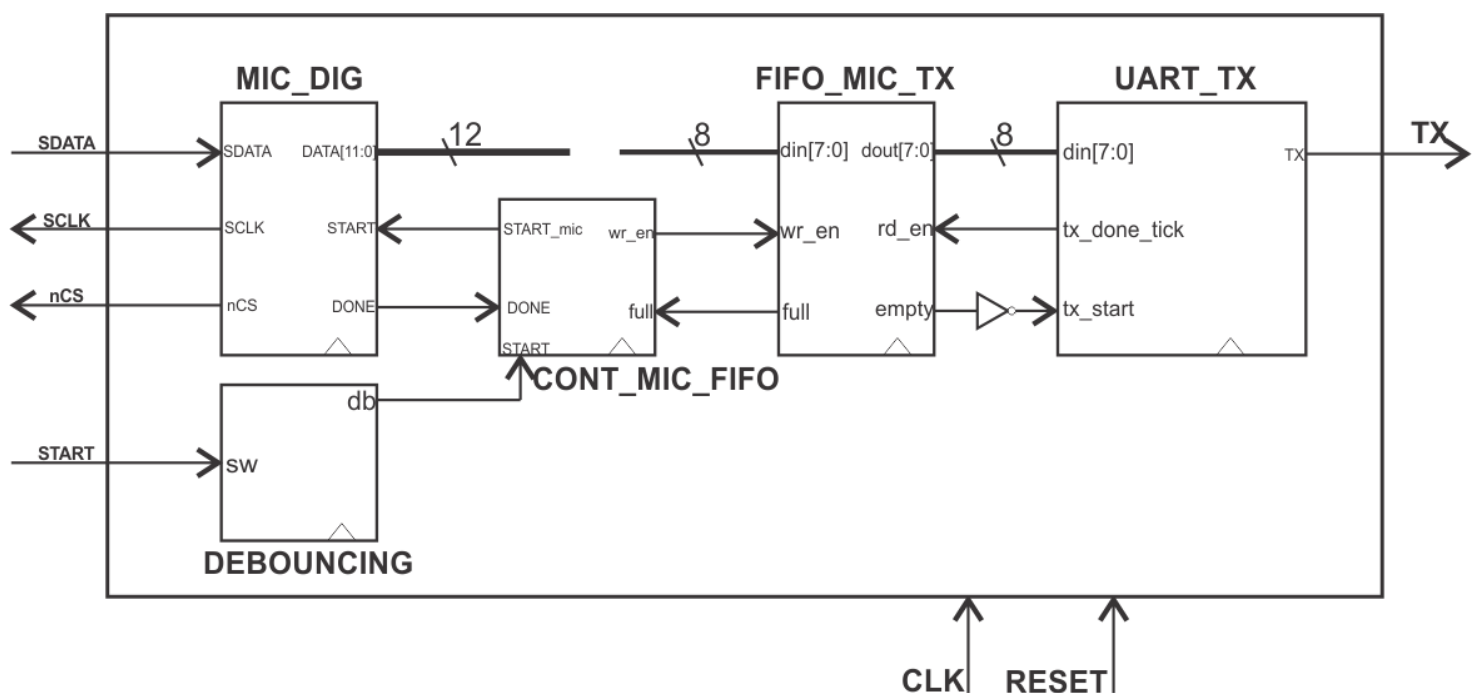


Figura 3: Módulos VHDL implementados en la FPGA

Se implementó el módulo **DEBOUNCING** para evitar los posible rebotes al activar el SWITCH de START.

El módulo **MIC_DIG** compuesto por 4 entradas y 4 salidas sincroniza la comunicación de datos entre la placa de desarrollo y el micrófono PmodMIC del cual recibe un vector serie de 16 bits y genera una palabra de 12 bits con el valor digital convertido. Las entradas del módulo son: un reloj de 50 MHz (CLK), un RESET asincrónico, los datos serie del ADC (SDATA) y una señal de START que se utiliza para decirle al componente cuando comenzar la conversión. Las salidas, en cambio, son: la señal que habilita el ADC (nCS), un reloj de 12,5 MHz (SCLK), la señal de DONE que avisa cuando la conversión está completa y el vector de datos (DATA) de 12 bits.

Con esta configuración una conversión es iniciada al ponerse la señal START en alto, y finaliza cuando se activa la salida DONE.

Los 8 bits más significativos del vector de datos generados en MIC_DIG son guardado en el módulo **FIFO_MIC_TX** que consiste en una memoria FIFO (*First In - First Out*) de 64 Kb. Además de las entradas de reloj y reset, este módulo tiene dos entradas (wr_en y rd_en) que habilitan la escritura y lectura respectivamente y dos salidas (full y empty) que se activan cuando la memoria está llena o vacía.

El módulo **CONT_MIC_FIFO** regula la comunicación entre MIC_DIG y la memoria FIFO, además de controlar la frecuencia a la cual se generan las conversiones de nuevos datos. La Fig. (4) muestra la maquina de estado de funcionamiento de este módulo.

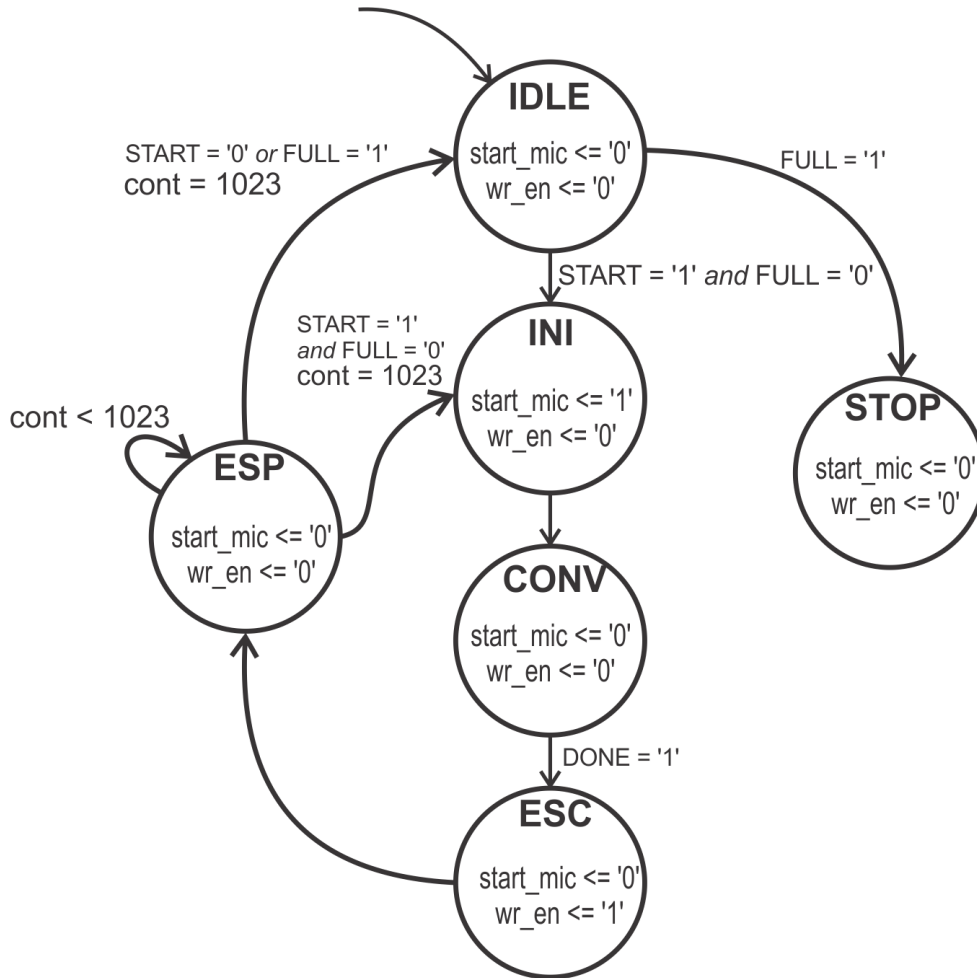


Figura 4: Maquina de estados que describe el funcionamiento del módulo CONT_MIC_FIFO

Las salidas del módulo de control son *start_mic* que inicia una conversión y *wr_en* que guarda la salida actual DATA de MIC_DIG en la memoria. Mientras que las entradas son el START generado por el SWITCH externo a la FPGA y la señal de *full* para evitar realizar conversiones cuando la memoria está llena.

La maquina de estado comienza en IDLE hasta que se activa el SWITCH y pasa al estado INI donde se activa la señal *start_mic* para iniciar una conversión. En el siguiente pulso de reloj pasa a un estado CONV donde espera que finalice la conversión, es decir espera que la entrada DONE se ponga en alto. Cuando esto ocurre transcurre a

ESC donde se activa la salida *wr_en* para escribir el dato recién convertido en la memoria. Por último el sistema permanece en el estado ESP hasta que un contador llega a 1023, luego si todavía el SWITCH permanece activado y la memoria no está llena se inicia una nueva conversión, en caso contrario la maquina de estados pasa al estado IDLE. Luego si la memoria está llena pasa al estado STOP donde permanece hasta que resetee todo el sistema.

El estado ESP se utilizó para controlar la frecuencia a la cuál se generaron los datos. Como el reloj utilizado es de 50 MHz y se realiza una conversión cada 1024 ciclos, la frecuencia de muestreo es aproximadamente 49 KHz.

Los datos almacenados en la memoria son enviados a la computadora utilizando el protocolo RS232. Con este fin se implementó el módulo **UART_TX**, este recibe el vector de datos de la memoria y genera la señal serie TX.

En el módulo se configuró la tasa de transferencia en 57600 baudios, utilizado un bit de stop y sin bit de paridad.

La entrada *tx_start* se encuentra en alto siempre y cuando la memoria no esté vacía, es decir, que haya algún data para transmitir. La salida *tx_done_tick* se activa cuando se completa una transferencia y habilita la lectura de un nuevo dato de la memoria para continuar enviando datos.

3. Implementación

A través de un puerto USB se bajamos el programa completo a la FPGA, conectamos el PmodMIC al conector JA de la placa y utilizamos el puerto USB-UART para conectarnos con la computadora.

A través del programa Realterm pudimos tener acceso a los datos obtenidos y guardarlos en un archivo de textos.

Para probar el funcionamiento del proyecto y comprobar que la frecuencia de muestreo es la deseada se grabó un tono de 440 Hz, cargamos el vector de datos adquiridos a Matlab y realizamos su transformada de Fourier seteando la frecuencia de muestreo en 49 KHz, ver Fig. (5).

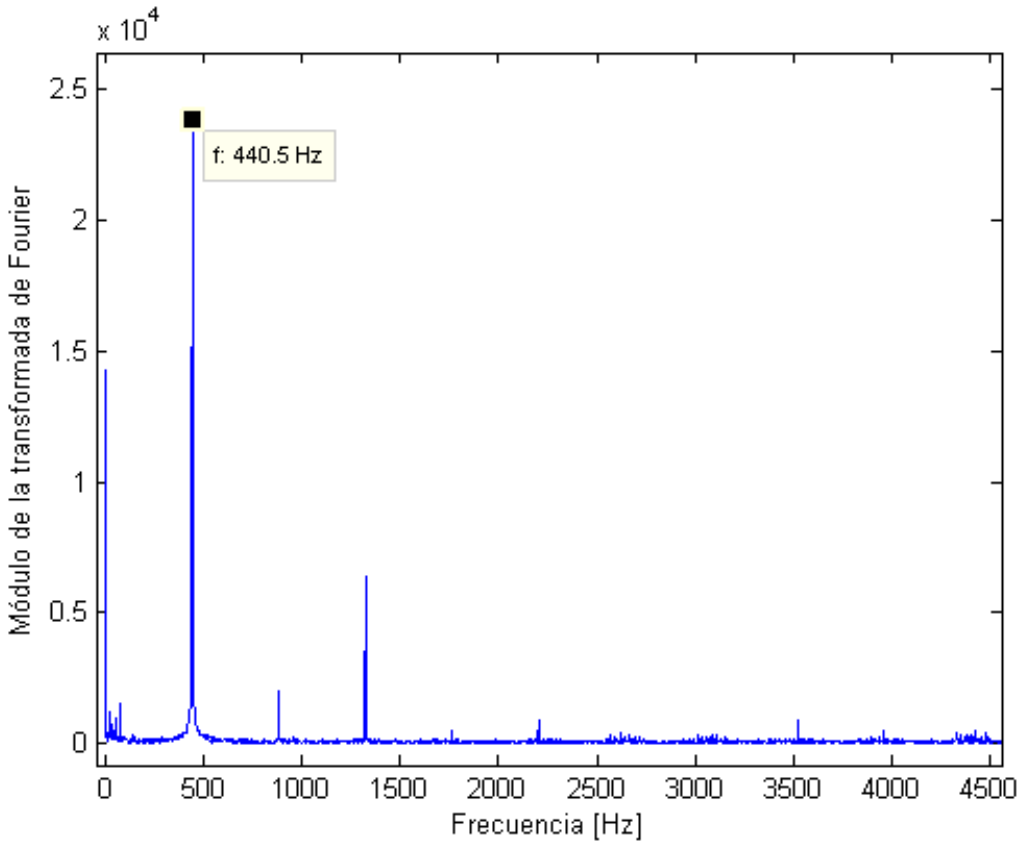


Figura 5: Transformada de Fourier de los datos adquiridos con el micrófono, al grabar un tono de 440 Hz

Como esperabamos se observa un pico en la frecuencia de 440Hz, lo que nos indica que la frecuencia de muestreo es la correcta.

Por último calculamos el tiempo máximo de grabación hasta que se llena la memoria FIFO: La velocidad de escritura en la memoria está dada por la frecuencia de muestreo, 48800 bytes/seg mientras que la velocidad de la comunicación serie está seteada en 57600 bits/seg es decir 7200 bytes/seg. Como el tamaño de la memoria es de 64 Kb tenemos que:

$$t_{max} = \frac{2^{16}}{48800 - 7200} \simeq 1,6 \text{ seg} \quad (1)$$

4. Conclusiones

Desarrollamos un sistema de adquisición de datos de un micrófono a la computadora utilizando una FPGA a través de una interfase serie RS232. Realizamos la programación de la FPGA en el lenguaje de programación de hardware VHDL, este consistió en 4 módulos principales: un módulo para sincronizar la comunicación entre la FPGA y el PmodMIC, otro la transmisión de datos serie a través del puerto USB de la placa, una memoria FIFO de 64Kb para independizar la adquisición y transmisión de datos en la FPGA y por último un módulo de control necesario para coordinar la conexión entre la memoria y la unidad del micrófono, además este último módulo controlaba la frecuencia a la cual el ADC del PmodMIC tomaba los datos digitales.

Configuramos la frecuencia de muestreo en 49 KHz para evitar efectos de aliasing o solapamiento en frecuencias menores a 20 KHz.

La comunicación serie la configuramos con un solo bit de stop, sin bit de paridad y a una tasa de transferencia de 57600 baudios.

Luego de bajar el programa a la placa verificamos su correcto funcionamiento grabando un tono de 440 Hz para analizarlo en Matlab.

Por último vimos que el tiempo máximo de adquisición de datos es de 1,6 seg.