FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA URUGUAY
NEON Network of Competence on Internet of Things
Co-funded by the Erasmus+ Programme of the European Union

Práctica 2.1: Técnicas de reducción de consumo en FPGAs

En esta práctica se implementarán bloques que realizan la multiplicación binaria en una FPGA y se aplicarán distintas técnicas para reducir el consumo dinámico. Para verificar la efectividad de las técnicas se utilizarán las herramientas de estimación de consumo provistas por el fabricante y también se realizarán mediciones reales de consumo en la FPGA.

Esta práctica se realizará en dos etapas (2.1 y 2.2) y se debe entregar un único reporte al finalizar la segunda parte. En esta primera etapa se busca familiarizarse con las herramientas, probar un diseño en la FPGA y realizar algunas medidas y estimaciones de consumo.

Objetivos

Poner en práctica diferentes técnicas para reducir el consumo de potencia en diseños con FPGAs.

Estimar consumo utilizando las herramientas de diseño del fabricante de FPGAs.

Medir el consumo real del diseño en una FPGA.

Materiales

Placa de desarrollo Cyclone 10 LP Evaluation Kit (FPGA Cyclone 10 LP de Intel).

Placa Analog Discovery 2 en modo analizador lógico para ver salidas del circuito.

Software Quartus Prime para síntesis y programación de FPGAs de Intel (Altera) y ModelSim para las simulaciones. Se proveerá un proyecto archivado de Quartus con todos los archivos y configuraciones necesarias.

Breve descripción del circuito a utilizar

Se busca evaluar un diseño completo en un escenario lo más realista posible. Para ello es necesario generar distintos patrones a la entrada del circuito bajo test y poder analizar la salida. También se requiere simular el diseño, para poder estimar el consumo. Para lograr todo esto se plantea el setup de la figura 1. Además del circuito a evaluar, el sistema de medida consta de otros bloques dentro de la FPGA que asisten en la medida: un LFSR (Linear-Feedback shift register) parametrizable, un controlador de reloj, una barrera de registros y un generador de paridad (árbol de reducción por xor). En la figura 1 se muestra además el método para medir el consumo interno de la FPGA, que es un resistor shunt en

serie con la fuente de alimentación. Midiendo la caída de tensión en el shunt se puede obtener la corriente consumida por el core de la FPGA.

El LFSR genera una señal pseudo-aleatoria que se conecta a la entrada del circuito para producir todos los patrones de entradas posibles. El Control de reloj permite desactivar el reloj del sistema. La barrera de Flip-Flops mantiene estable durante un período de reloj la salida del circuito a testear. El árbol de XOR combina todas las salidas del circuito de test en una única señal que se coloca en un puerto de salida. Se utiliza el XOR en lugar de otras operaciones lógicas (AND, OR) para evitar que la herramienta de síntesis simplifique el diseño.

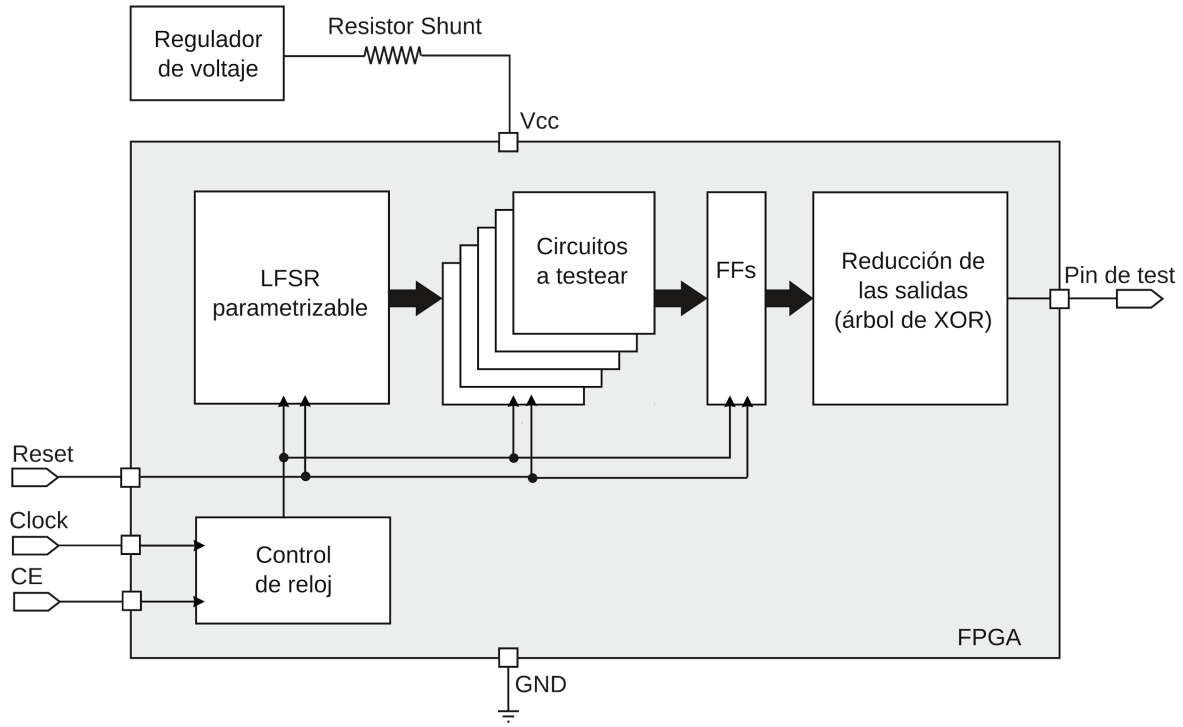

Figura 1: Diagrama del setup experimental para las medidas de consumo.

Para los circuitos de test se eligen multiplicadores porque son fáciles de usar, su salida es fácilmente verificable, las herramientas de diseño ya proveen bloques multiplicadores (lo que evita tener que diseñarlos), son configurables (cantidad de bits de entradas y salidas) y por último, tienen un consumo de potencia relativamente alto (y no es sencillo medir consumos muy bajos).

Proyecto en Quartus

En primer lugar deberán familiarizarse con las herramientas de diseño, las FPGAs y los archvos fuente que se utilizarán en esta práctica. Para ello deben descargar el archivo `LPDD_LAB2_MULT.qar` que es un proyecto archivado del Quartus. Luego abren el proyecto en Quartus: “Project>Restore Archived Project”.

Los archivos presentes en el proyecto son los siguientes:

```
LPDD_LAB2_MULT/
├── src/
│   ├── intel_clkctrl/
│   ├── intel_pll/
│   ├── lpm_mult_32/
│   ├── lpdd_lab2_mult.v
│   ├── lfsr_64_bits.vhd
│   └── parity.vhd
├── tb/
│   └── lpdd_lab2_mult_tb.v
├── LPDD_LAB2_MULT.qpf
├── LPDD_LAB2_MULT.qsf
└── LPDD_LAB2_MULT.sdc
```

A continuación se describen someramente los archivos incluidos en el proyecto. Los archivos con el código fuente HDL (directorio `src/`) de los bloques `intel_clkctrl`, `intel_pll y lpm_mult_32` son generados automáticamente por Quartus. Cualquiera de estos IP de Intel puede ser editado dando doble-click al bloque en la ventana “Project Navigator>IP Components”.

`intel_clkctrl/` Controla las líneas de reloj de la FPGA. Este bloque permite aplicar la técnica clock gating. Configurado mediante el *Quartus Platform Designer* al instanciar el bloque “ALTCLKCTRL Intel FPGA IP” (Tools>IP Catalog).

`intel_pll/` Permite configurar y añadir al diseño un Phase-Locked Loop (PLL), el cual permite obtener distintas frecuencias de reloj a partir del reloj de 50 MHz de la placa. Configurado mediante el MegaWizard al instanciar el bloque “PLL Intel FPGA IP” (Tools>IP Catalog).

`lpm_mult_32/` Multiplica dos números enteros sin signo de 32 bits y entrega el resultado en 64 bits sin signo. Se puede configurar cantidad de etapas de pipeline. Configurado mediante el MegaWizard al instanciar el bloque “LPM_MULT” (Tools>IP Catalog).

`lpdd_lab2_mult.v` Archivo de diseño Top-Level. Implementa el diseño mostrado en el diagrama de la figura 2. Este archivo se debe modificar para cambiar la frecuencia de operación (parte A).

`lfsr_64_bits.vhd` Linear-Feedback Shift Register de 64 bits.

`parity.vhd` Bloque cuya salida es el XOR de todas sus entradas. La cantidad de entradas es un parámetro configurable.

`lpdd_lab2_mult_tb.v` Testbench del Top-Level para simular el diseño. Esto es necesario para estimar el consumo (ver Anexo).

`LPDD_LAB2_MULT.qpf` Archivo de proyecto del Quartus.

`LPDD_LAB2_MULT.qsf` Archivo de configuraciones del proyecto. Incluye, por ejemplo, la asignación de pines de la FPGA a las señales del diseño.

`LPDD_LAB2_MULT.sdc` Archivo de restricciones (*constraints*) del diseño.

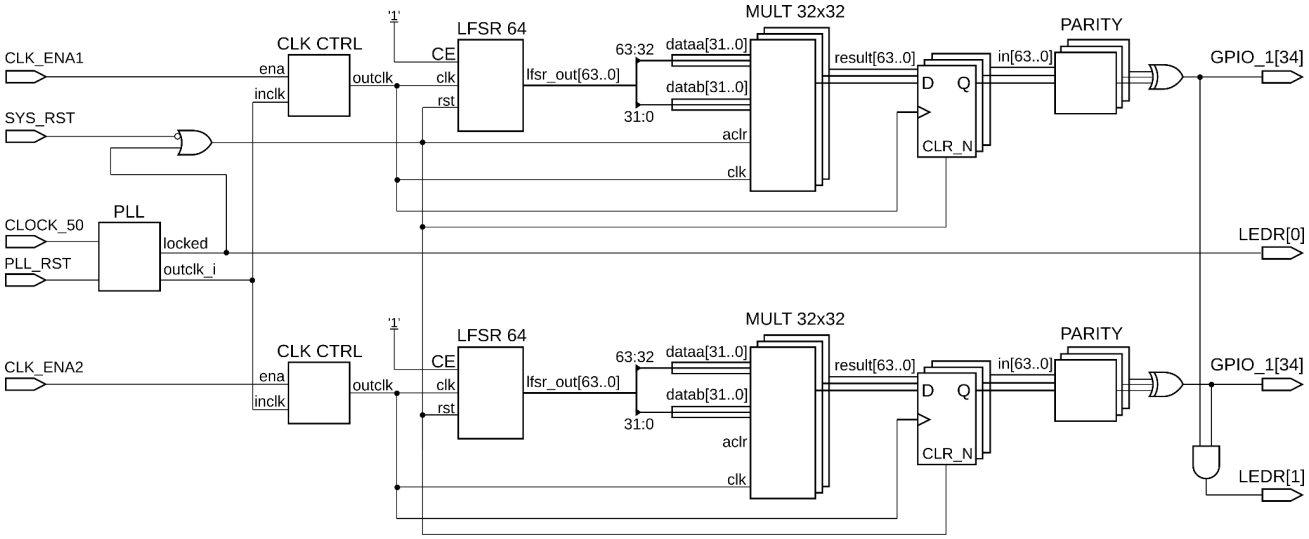

Figura 2: Diagrama de bloques del diseño del laboratorio (Top-Level).

Prueba en FPGA

Compilar el diseño y cargarlo en la placa.

Para verificar el funcionamiento deben conectar dos puertos digitales de la Analog Discovery 2 a los GPIO34 y GPIO35 de la placa (estos corresponden a las salidas de los XOR en el diseño). En el Waveform utilicen la función *Logic* agregando las señales correspondientes a los pines conectados. Si todo funciona bien se debería ver una señal que cambia de estado continuamente de forma aparentemente irregular.

Tengan en cuenta que el estado de los switches y los botones modifican el comportamiento del sistema. En la Tabla 1 se indican las señales de entrada del diseño, su nombre en la placa y en el esquemático y la función que cumplen (indicando además en que nivel lógico son activas). En la Figura 3 se muestra una imágen y un diagrama indicando en la placa cuáles son botones y los swtiches conectados al circuito en la FPGA.

Signal	Board Reference	Schematic	Función
PLL_RST	SW1.1	USER_DIP2	Reset PLL: Mantiene el PLL en estado de reset con un ‘1’.
SYS_RST	S3	USER_PB0	Reset sistema: Mantiene el sistema en estado de reset con un ‘1’.
CLK_ENA1	SW1.2	USER_DIP1	Enable clkctrl 1: habilita el bloque con un ‘1’.
CLK_ENA2	SW1.3	USER_DIP0	Enable clkctrl 2: habilita el bloque con un ‘1’.

Tabla 1: Señales de entrada del diseño y su función.

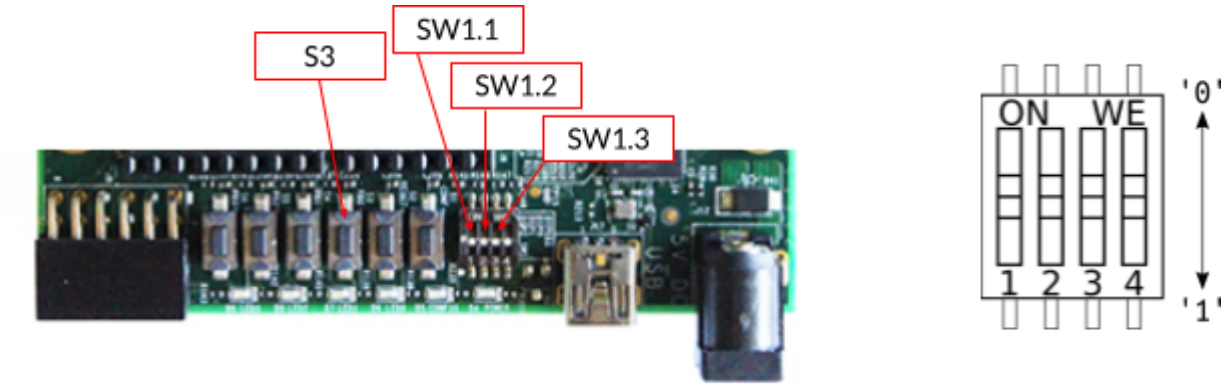

Figura 3: Indicación de los switches y botones conectados al diseño (izquierda). Diagrama representando que nivel lógico recibe la FPGA según el estado de los switches (derecha).

Parte A: Frecuencia de operación

En esta parte es necesario configurar el bloque PLL para modificar la frecuencia de reloj del circuito. Las frecuencias a evaluar son: 30 MHz, 40 MHz y 50 MHz. Para ello, ir a "Project Navigator>IP Components" y dar doble click en `intel_pll` para abrir el GUI de configuración. Luego ir a "Output Clocks>clk c0" y seleccionar "Enter output clock frequency". En cuadro bajo "Requested Settings" ingresar la frecuencia de salida deseada y finalmente presionar el botón Copy para fijar los parámetros. Para terminar presionar Finish (abajo a la derecha) y recompilar todo el diseño.

El bloque que instancia al PLL es el siguiente:

```
intel_pll intel_pll_inst (
            .areset   (pll_rst),
            .inclk0   (CLOCK_50),
            .c0       (pll_out_clk), // 30MHz
            .locked   (pll_locked)
      );
```

Clock del sistema conectado a salida del PLL de 30MHz

Para cada una de las frecuencias compilar el diseño, cargarlo en la placa y verificar el funcionamiento, observando las salidas en el analizador lógico del Analog Discovery 2.

Nota: Los tiempos de compilación del diseño pueden ser largos. Para evitar demoras entre las distintas pruebas se sugiere crear tres proyectos, uno para cada frecuencia de operación.

Parte A.1: Estimación de consumo

Siguiendo la guía provista en el Anexo 1 simular el circuito para las tres frecuencias y obtener la estimación de consumo en cada caso. A efectos de poder realizar una comparación justa con las medidas reales, al estimar el consumo ingresar una temperatura ambiente lo más cercana posible a la que se realizarán las medidas.

Parte A.2: Medidas reales

Medir el consumo del core de la FPGA para el circuito funcionando en las tres frecuencias mencionadas. Además, medir el consumo de los circuitos sin reloj, activando el reset del PLL (`SW[0]='1'`).

Nota: Para ordenar los resultados se sugiere utilizar la tabla del Anexo 2 para anotar las condiciones del experimento y las medidas realizadas.

Se deberá entregar un reporte conteniendo lo siguiente:

1. Resultados de todos los experimentos.
2. Cálculo del error en la estimación vs medidas.

Se deberá entregar un único reporte al finalizar las dos partes de esta práctica.

Anexo 1: Estimación de consumo

Para realizar estimaciones de consumo lo más parecidas a la realidad es necesario seguir los siguientes pasos:

1. Configurar el Quartus para que registre la información de switcheo de todos los nodos durante la simulación en un archivo (formato VCD). Para esto ir a Settings>EDA Tool Settings>Simulation>Options for Power Estimation y seleccionar la opción: *Generate VCD file script*. En *Script Settings...* marcar *All signals.*

En el menú de simulación también se debe configurar el Testbench. Para ello seleccionar la opción *Compile test bench:*, luego ir a *Test Benches…*>New… y configurar los siguientes campos:

- Ingresar el nombre del test bench: `lpdd_lab2_mult_tb`
- Ingresar el top level module del test bench: `lpdd_lab2_mult_tb`
- Seleccionar la opción *Use test bench to perform VHDL timing simulation*
- Ingresar el design instance name: `DUT`
- Seleccionar *End simulation at:* e ingresar el tiempo de simulación: 100us
- En **Test bench and simulation files** agregar el archivo del test bench: `lpdd_lab2_mult_tb.v`

Se debe tener cuidado al elegir el tiempo de simulación, si este es muy corto no alcanzará para capturar el funcionamiento completo del circuito y si es muy largo tomaría demasiado tiempo ya que la simulación Gate-Level es lenta.

2. Luego de compilar, simular el diseño a nivel de compuertas (Gate-Level) para generar el archivo VCD: “Tools>Run Simulation Tool>Gate Level Simulation…”

Esto ejecuta el simulador que se haya configurado en “Tools>Options...>General>EDA Tool Options”. En nuestro caso vamos a utilizar ModelSim-Altera. Asegurarse que la ruta del ejecutable del ModelSim está bien configurada en el menú mencionado anteriormente.

3. Por último, para realizar la estimación abrir la herramienta “Processing>Power Analyzer Tool” y realizar las siguientes configuraciones:

Seleccionar “Input file>Use input file to initialize toggle rates and static probabilities during power analysis”. Luego ir a “*Add Power Input Files…”* y seleccionar el archivo VCD generado durante la simulación Gate-Level. Se deben establecer los tiempos iniciales y finales a considerar. El tiempo inicial conviene elegirlo de forma de no considerar el transitorio al comenzar la ejecución y el final debe coincidir con el tiempo de simulación.

Luego ir a “*Cooling Solutions and Temperature*” e introducir la temperatura ambiente (a efectos de comparar con la medida se debería poner la misma temperatura que había cuando se realizaron las medidas reales). En “Thermal resistance>Use cooling solution” elegir la opción “*No heat sink with still air”*, ya que la placa utilizada no posee ni disipador ni ventilación forzada.

Para estimar presionar Start y luego ver el reporte.

Para ver el resultado de la estimación en el reporte ir a: “Power Analyzer>Current Drawn from Voltage Supplies>Summary”. El consumo que interesa (el del core de la FPGA) es el correspondiente a la fila VCC.

Anexo 2: Test para medidas de consumo

La siguiente tabla ayudará a ordenar los resultados de los experimentos. Se da como ejemplo el experimento de medidas de consumo a distintas frecuencias.

Condiciones del experimento

- Temperatura ambiente:
- Tiempo de ejecución previo a medida*:
- R shunt:
- Instrumento medida:
- Configuración del instrumento**:

*- Tiempo necesario para que se estabilice la temperatura interna del chip.
**- Base de tiempo, rango de voltaje, puntas, etc.

Medidas reales

Frecuencia (MHz)	Voltaje medido shunt (mV)	Corriente CORE (mA)	Voltaje CORE (V)
30			
40			
50			
Sin Reloj[#]			

[#]- PLL Reset

Estimaciones

Frecuencia (MHz)	Tiempo de simulación (us)	Configuraciones	Corriente CORE (mA)	Voltaje CORE (V)
30		Glitch Filt Power: ON Glitch Filt Power: OFF		
40		Glitch Filt Power: ON Glitch Filt Power: OFF		
50		Glitch Filt Power: ON Glitch Filt Power: OFF		