Ejemplo de Uso M_AXI_LITE y M_AXI y comparación

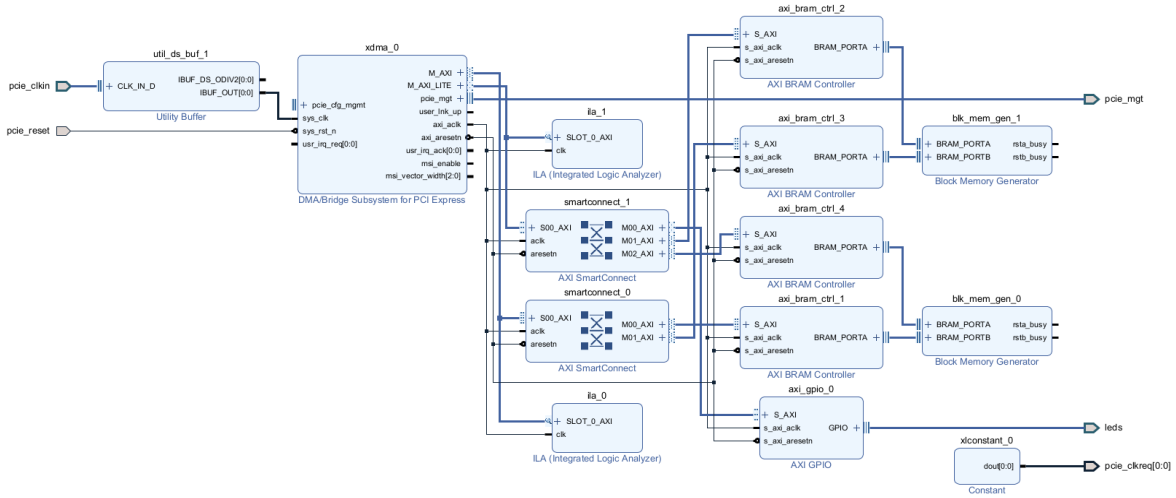


Ilustración 1 Arquitectura

Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/xdma_0					
/xdma_0/M_AXI (64 address bits : 16E)					
/axi_bram_ctrl_1/S_AXI	S_AXI	Mem0	0xC000_0000	1M	0xC00F_FFFF
/axi_bram_ctrl_3/S_AXI	S_AXI	Mem0	0xC200_0000	8K	0xC200_1FFF
Network 1					
/xdma_0					
/xdma_0/M_AXI_LITE (32 address bits : 4G)					
/axi_bram_ctrl_2/S_AXI	S_AXI	Mem0	0x10_0000	8K	0x10_1FFF
/axi_bram_ctrl_4/S_AXI	S_AXI	Mem0	0x0	1M	0xF_FFFF
/axi_gpio_0/S_AXI	S_AXI	Reg	0x10_2000	128	0x10_207F

Ilustración 2 Mapeo de Memoria

Introducción

Esta guía proporciona instrucciones detalladas sobre cómo interactuar con las interfaces M_AXI_LITE y M_AXI utilizando el código C proporcionado. El diagrama de bloques de Vivado incluido ilustra la arquitectura del sistema, que incluye la configuración y las conexiones de varios componentes como los controladores AXI BRAM, los generadores de memoria y el subsistema DMA/Bridge para PCI Express.

Prerrequisitos

- Asegúrese de que el dispositivo esté conectado y que los controladores necesarios estén instalados.
- El código asume que los archivos de dispositivo `/dev/xdma0_h2c_0` y `/dev/xdma0_c2h_0` están disponibles para la comunicación.

Descripción de las Constantes y Macros

- **Direcciones Base y Tamaños de BRAM:**
  - `BRAM0_BASE_ADDR` define la dirección base de BRAM0.
  - `BRAM1_BASE_ADDR` define la dirección base de BRAM1.
  - `BRAM2_BASE_ADDR` define la dirección base de BRAM2.
  - `BRAM0_SIZE`, `BRAM1_SIZE` y `BRAM2_SIZE` definen los tamaños de cada BRAM en bytes.

Funciones para Operaciones con BRAM

Escritura en BRAM

Para escribir datos en una BRAM específica, se utiliza la función `write_to_bram`. Esta función requiere el descriptor de archivo del dispositivo, la dirección base de la BRAM, los datos a escribir y el tamaño de los datos.

Lectura de BRAM

Para leer datos de una BRAM específica, se utiliza la función `read_from_bram`. Esta función requiere el descriptor de archivo del dispositivo, la dirección base de la BRAM, un buffer para almacenar los datos leídos y el tamaño de los datos a leer.

Ejecución del Código Principal

1. **Apertura de Archivos de Dispositivo:**
   - Se abren los archivos de dispositivo `/dev/xdma0_h2c_0` y `/dev/xdma0_c2h_0` para la comunicación host-to-card (h2c) y card-to-host (c2h), respectivamente.
2. **Preparación de Datos:**
   - Se preparan los datos que se escribirán en cada BRAM, llenándolos con valores consecutivos en hexadecimal.
3. **Medición del Tiempo de Escritura:**
   - Se mide el tiempo que tarda en escribir los datos en cada BRAM y se calcula la velocidad de escritura en Gb/s.
4. **Medición del Tiempo de Lectura:**
   - Se mide el tiempo que tarda en leer los datos de cada BRAM y se calcula la velocidad de lectura en Gb/s.
5. **Visualización de Resultados:**
   - Se muestran los tiempos y las velocidades de escritura y lectura para cada BRAM.

Uso de la Interfaz M_AXI_LITE

Para utilizar la interfaz M_AXI_LITE, se pueden enviar datos directamente mapeando el recurso. Esta interfaz es sencilla y no requiere un controlador complejo.

Hay que tener en cuenta que hay que seleccionar en vivado los rangos de direcciones de memoria para que empiecen desde 0 y que se desperdicien en numero menor de direcciones de memoria entre todos los componentes ya que estos luego estarán mapeados en la memoria del PC.

Uso de la Interfaz M_AXI

Para utilizar la interfaz M_AXI, se necesita un controlador de Xilinx ya que hace uso del DMA y los datos se envían en modo burst. Este controlador facilita la transferencia eficiente de grandes bloques de datos.

Resultados

Ya que la interfaz full permite hacer transacciones en modo burst se logra una enorme mejora en las prestaciones al hacer uso de ella.
Resultados transmisión de 1Mb y 8Kb:

```
rodrigo@ubuntu:~/test_pcie/build$ /home/rodrigo/test_pcie/build/dma_example
AXI LITE
BRAM0 - Elapsed time for writing: 0.023182 seconds, Write speed: 0.045232 Gb/s
BRAM0 - Elapsed time for reading: 0.434333 seconds, Write speed: 0.002414 Gb/s
BRAM1 - Elapsed time for writing: 0.000211 seconds, Write speed: 0.038825 Gb/s
BRAM1 - Elapsed time for reading: 0.003396 seconds, Write speed: 0.002412 Gb/s
AXI FULL
BRAM0 - Elapsed time for writing: 0.000140 seconds, Write speed: 7.489829 Gb/s
BRAM0 - Elapsed time for reading: 0.000090 seconds, Write speed: 11.650844 Gb/s
BRAM1 - Elapsed time for writing: 0.000072 seconds, Write speed: 0.113778 Gb/s
BRAM1 - Elapsed time for reading: 0.000019 seconds, Write speed: 0.431158 Gb/s
```

Ilustración 3 Tiempos de transmisión recepción

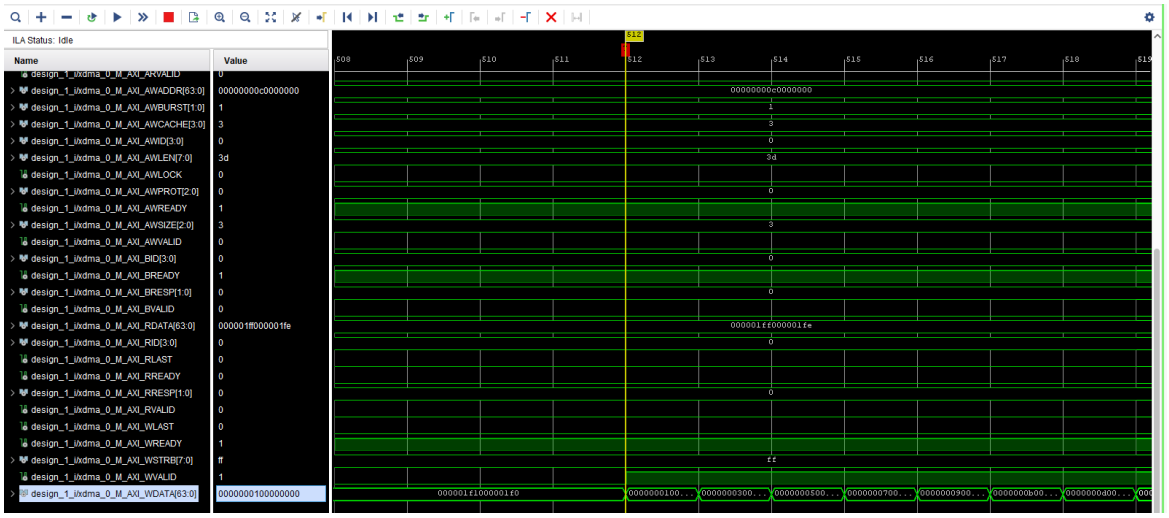


Ilustración 4 Transmisión burst

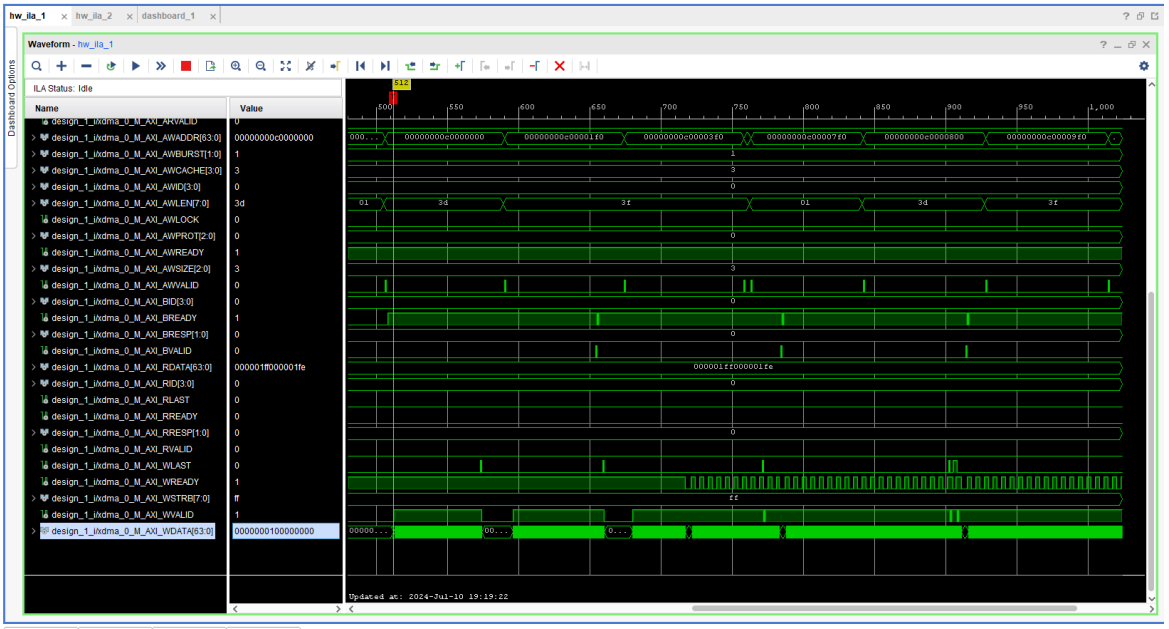
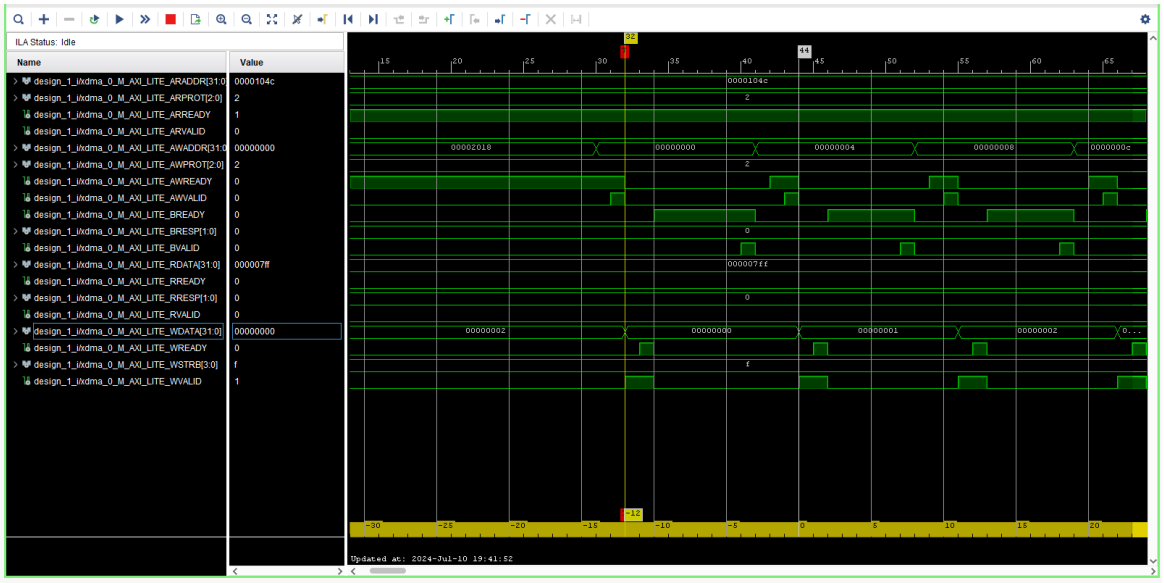
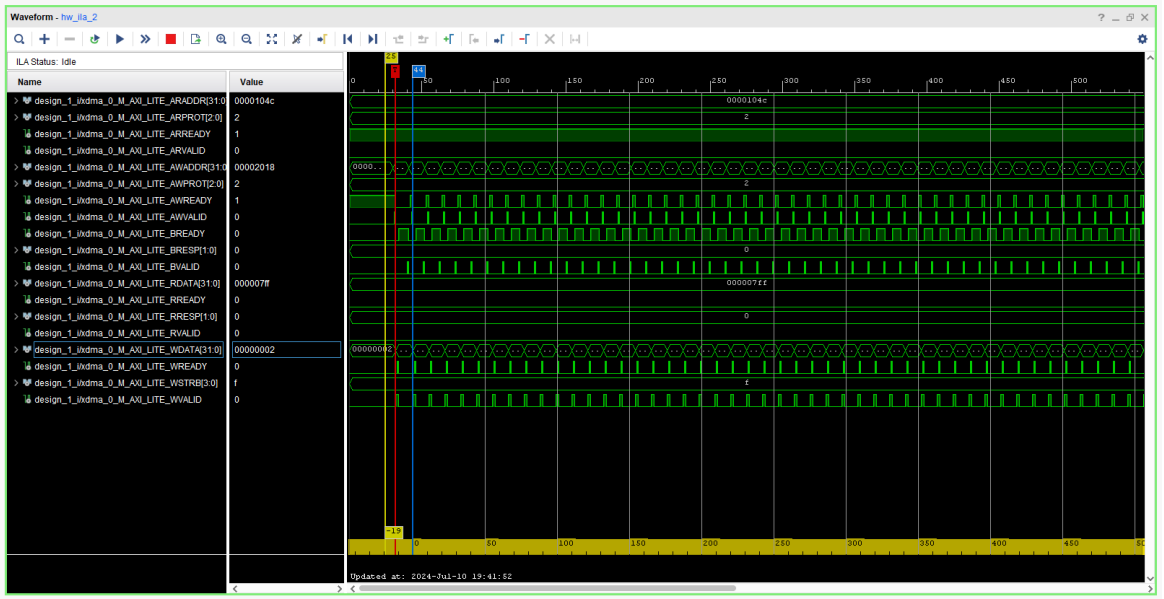


Ilustración 5 Transmisión burst





Como se aprecia en las imágenes de arriba con axi full podemos transmitir 64 bits por ciclo de reloj mientras que en axi lite transmitimos 32 bits cada 12 ciclos 24 veces menos, de ahí la gran mejora.

Ejemplo de Mapeo de Recursos PCIe

El código incluye un ejemplo de cómo mapear un recurso PCIe y leer/escribir valores. Este ejemplo muestra cómo abrir el dispositivo PCIe, mapear el recurso a la memoria del usuario, escribir un valor y leerlo de vuelta para verificar la operación.