



Análisis del Protocolo AXI y Arquitectura MicroBlaze V

Índice

1. Introducción	3
1.1. Propósito del documento	3
1.2. Alcance	3
2. Protocolo AMBA AXI	3
2.1. Historia y contexto	3
2.2. Tipos de interfaces AXI4	4
2.2.1. AXI4 (Full)	4
2.2.2. AXI4-Lite (Usado en este proyecto)	4
2.2.3. AXI4-Stream	4
3. Arquitectura del Protocolo AXI4-Lite	5
3.1. Características generales	5
3.2. Arquitectura de canales	5
3.2.1. Grupo de Escritura (3 canales)	5
3.2.2. Grupo de Lectura (2 canales)	6
3.3. Señales globales	7
4. Protocolo de Handshake VALID/READY	8
4.1. Mecanismo fundamental	8
4.2. Implicaciones para diseño	8
5. Transacciones de Escritura y Lectura en AXI4-Lite	9
5.1. Secuencia de escritura	9
5.2. Secuencia de lectura	9
6. Códigos de Respuesta	10
6.1. Tipos de respuesta	10
6.2. Interpretación de respuestas en el contexto del proyecto	11
7. Señales Específicas de AXI4-Lite	11
7.1. Tabla de señales	11
7.1.1. Señales Globales	11
7.1.2. Canal Write Address (AW)	12
7.1.3. Canal Write Data (W)	12
7.1.4. Canal Write Response (B)	12
7.1.5. Canal Read Address (AR)	12
7.1.6. Canal Read Data (R)	12
7.2. Señales AWPROT y ARPROT	13
7.3. Señal WSTRB (Write Strobes)	13
8. Arquitectura del Procesador MicroBlaze V	13
8.1. Descripción general	13

8.2. MicroBlaze V como maestro AXI	14
8.3. Mapa de memoria del MicroBlaze V	14
8.4. Integración con AXI Interconnect	15
9. Consideraciones de Diseño para Módulos AXI-Slave	16
9.1. Requerimientos de temporización y Manejo de reset	16
9.2. Mapa de registros (Register Map)	16
10. Aplicación en los Módulos IP Desarrollados	16
10.1. Módulos implementados	16
10.2. Flujo de integración en Vivado y Vitis	17

1. Introducción

1.1. Propósito del documento

El propósito de este documento es proporcionar una descripción detallada del protocolo AMBA AXI (Advanced eXtensible Interface), con énfasis particular en la variante AXI4-Lite utilizada en los módulos IP desarrollados, así como una caracterización de la arquitectura del procesador AMD MicroBlaze V.

Este análisis sirve como fundamento técnico para la implementación de los módulos AXI-slave que permiten la comunicación entre el procesador MicroBlaze V y periféricos personalizados en la plataforma FPGA Basys 3 (Artix-7 XC7A35T).

1.2. Alcance

El documento cubre:

- **Protocolo AXI:** Descripción general del estándar AMBA AXI4
- **AXI4-Lite:** Análisis detallado de la variante simplificada para acceso a registros
- **Arquitectura de canales:** Señales, handshakes y secuencias de transacciones
- **MicroBlaze V:** Características relevantes del procesador como maestro AXI
- **Integración:** Consideraciones para módulos IP AXI-slave.

El documento no cubre:

- Detalles exhaustivos de AXI4 Full (con soporte burst)
- AXI4-Stream (protocolo orientado a flujos de datos)
- Aspectos avanzados como coherencia de caché (ACE)

2. Protocolo AMBA AXI

2.1. Historia y contexto

El protocolo AXI (Advanced eXtensible Interface) es parte de la familia de buses AMBA (Advanced Microcontroller Bus Architecture) desarrollada por ARM Limited. La especificación AXI fue introducida en AMBA 3.0 (2003) y evolucionó a AXI4 en AMBA 4.0 (2010).

Versiones principales:

- **AXI3:** Primera versión (AMBA 3.0, 2003)
- **AXI4:** Segunda versión mayor (AMBA 4.0, 2010) - utilizada en este proyecto
- **AXI5:** Versión más reciente (AMBA 5.0, 2019)

2.2. Tipos de interfaces AXI4

AXI4 define tres tipos de interfaces o variantes, cada una optimizada para diferentes casos de uso:

2.2.1. AXI4 (Full)

- **Propósito:** Comunicación memory-mapped de alto rendimiento
- **Características clave:**
 - Soporte para transacciones burst (hasta 256 transferencias de datos)
 - Múltiples transacciones pendientes (outstanding transactions)
 - Reordenamiento de respuestas mediante IDs de transacción
- **Uso típico:** Interfaces de memoria, DMA, periféricos de alto rendimiento

2.2.2. AXI4-Lite (Usado en este proyecto)

- **Propósito:** Comunicación memory-mapped simple para registros de control/estado
- **Características clave:**
 - Transacciones de una sola transferencia (sin bursts)
 - Menor complejidad lógica (footprint reducido)
 - Subset de señales de AXI4 Full
 - Protocolo sencillo de implementar y depurar

- **Uso típico:** Acceso a registros de configuración de periféricos (MMIO)

Nota: Los módulos IP desarrollados en este proyecto (PRNG, TRNG, 7-Segment, PulseGen, Perceptron) utilizan AXI4-Lite por su simplicidad y adecuación para operaciones de lectura/escritura de registros.

2.2.3. AXI4-Stream

- **Propósito:** Transferencias de datos de alto rendimiento sin dirección
- **Características clave:**
 - Sin fase de dirección (address-less)
 - Bursts de tamaño ilimitado
 - Ideal para procesamiento de datos en streaming
- **Uso típico:** Video, audio, DSP, comunicaciones

3. Arquitectura del Protocolo AXI4-Lite

3.1. Características generales

AXI4-Lite se basa en una arquitectura de canales independientes con mecanismo de handshake bidireccional. Las características principales son:

1. **Canales separados para lectura y escritura:**

- Permite operaciones concurrentes de lectura y escritura
- Simplifica la lógica de arbitraje

2. **Transacciones de una sola transferencia:**

- Cada transacción transfiere un solo dato (32 o 64 bits típicamente)
- No hay soporte para burst (simplifica diseño)

3. **Protocolo VALID/READY:**

- Cada canal usa señales VALID (emisor) y READY (receptor)
- La transferencia ocurre cuando ambas señales están en alto

4. **Respuestas de transacción:**

- Código RRESP (lectura) o BRESP (escritura)
- Indica éxito (OKAY) o error (SLVERR, DECERR)

3.2. Arquitectura de canales

AXI4-Lite utiliza cinco canales independientes organizados en dos grupos:

3.2.1. Grupo de Escritura (3 canales)

Canal AW (Write Address):

- **Dirección:** Maestro → Esclavo
- **Función:** Transmite la dirección de escritura
- **Señales principales:**
 - AWADDR[31:0] - Dirección del registro destino
 - AWVALID - Emisor indica dirección válida
 - AWREADY - Receptor indica capacidad de aceptar

Canal W (Write Data):

- **Dirección:** Maestro → Esclavo
- **Función:** Transmite el dato a escribir
- **Señales principales:**
 - WDATA[31:0] - Dato a escribir (ancho configurable)
 - WSTRB[3:0] - Byte enable (máscara de bytes válidos)
 - WVALID - Emisor indica dato válido
 - WREADY - Receptor indica capacidad de aceptar

Canal B (Write Response):

- **Dirección:** Esclavo → Maestro
- **Función:** Confirma completitud de escritura
- **Señales principales:**
 - BRESP[1:0] - Código de respuesta (OKAY, SLVERR, DECERR)
 - BVALID - Emisor indica respuesta válida
 - BREADY - Receptor indica capacidad de aceptar

3.2.2. Grupo de Lectura (2 canales)

Canal AR (Read Address):

- **Dirección:** Maestro → Esclavo
- **Función:** Transmite la dirección de lectura
- **Señales principales:**
 - ARADDR[31:0] - Dirección del registro fuente
 - ARVALID - Emisor indica dirección válida
 - ARREADY - Receptor indica capacidad de aceptar

Canal R (Read Data):

- **Dirección:** Esclavo → Maestro
- **Función:** Devuelve el dato leído
- **Señales principales:**
 - RDATA[31:0] - Dato leído (ancho configurable)
 - RRESP[1:0] - Código de respuesta (OKAY, SLVERR, DECERR)

- RVALID - Emisor indica dato válido
- RREADY - Receptor indica capacidad de aceptar

La Figura 1 muestra cómo una transacción de escritura utiliza los canales de dirección de escritura, datos de escritura y respuesta de escritura. La Figura 2 muestra cómo una transacción de lectura utiliza los canales de dirección de lectura y datos de lectura.

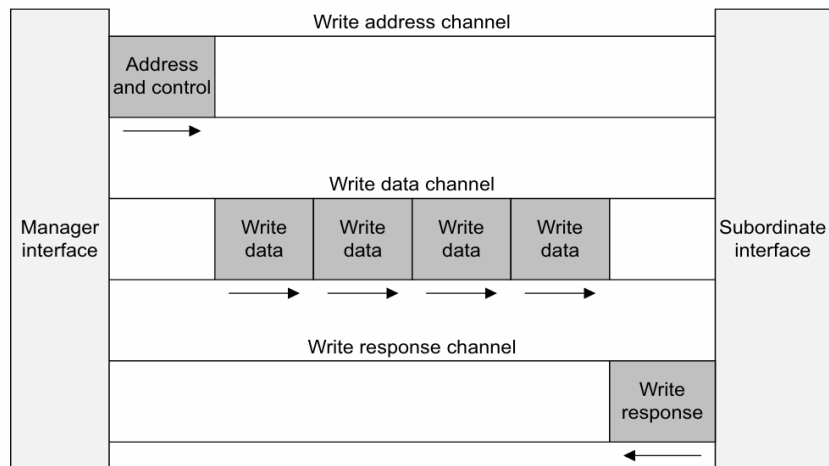


Figura 1: Arquitectura de canales de escritura [1]

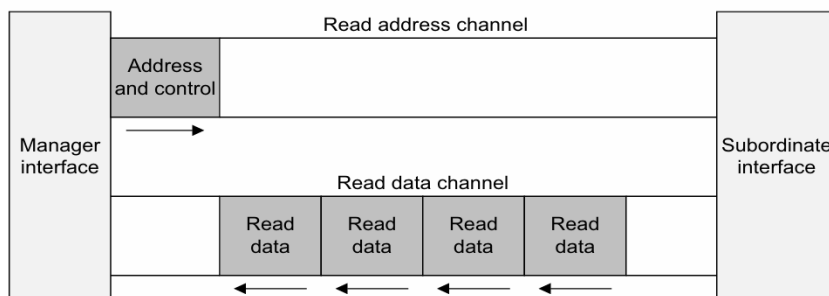


Figura 2: Arquitectura de canales de lectura [1]

3.3. Señales globales

Además de los canales de datos, AXI4-Lite requiere dos señales globales de sistema:

Señal	Tipo	Descripción
ACLK	Entrada (reloj)	Reloj del bus AXI. Todas las señales son síncronas a este reloj.
ARESETn	Entrada (reset activo bajo)	Reset asíncrono. Estado bajo inicializa la lógica del esclavo.

4. Protocolo de Handshake VALID/READY

4.1. Mecanismo fundamental

El protocolo AXI se basa en un handshake bidireccional entre emisor (source) y receptor (destination). Este mecanismo garantiza que ambos lados están listos antes de realizar la transferencia.

Reglas del handshake:

1. **El emisor activa VALID** cuando tiene información válida disponible
2. **El receptor activa READY** cuando puede aceptar la información
3. **La transferencia ocurre** en el flanco ascendente de ACLK cuando $VALID \wedge READY = 1$
4. **VALID no puede depender de READY** (evita ciclos combinacionales)
5. **READY puede depender de VALID** (es opcional activar READY antes de VALID)

4.2. Implicaciones para diseño

Para el módulo AXI-slave (implementados en este proyecto):

1. **Señales READY (entradas desde el esclavo):**
 - AWREADY, WREADY (escritura)
 - ARREADY (lectura)
 - Deben activarse cuando el esclavo puede procesar la transacción
 - No deben depender combinacionalmente de VALID (evita bucles)
2. **Señales VALID (salidas desde el esclavo):**
 - BVALID (respuesta de escritura)
 - RVALID (respuesta de lectura)
 - Deben activarse cuando el esclavo tiene datos/respuesta válida
3. **Persistencia de datos:**
 - Mientras $VALID = 1$ y $READY = 0$, los datos deben mantenerse estables
 - Una vez completada la transferencia ($VALID \wedge READY$), los datos pueden cambiar

5. Transacciones de Escritura y Lectura en AXI4-Lite

5.1. Secuencia de escritura

Una transacción de escritura en AXI4-Lite involucra los siguientes pasos:

1. **Paso 1:** Maestro presenta dirección en canal AW
2. **Paso 2:** Maestro presenta dato en canal W
3. **Paso 3:** Esclavo acepta dirección y dato (handshakes independientes)
4. **Paso 4:** Esclavo procesa la escritura internamente
5. **Paso 5:** Esclavo envía respuesta en canal B
6. **Paso 6:** Maestro acepta respuesta

Nota importante: Los canales AW y W son independientes. El maestro puede presentar dirección y dato en cualquier orden o simultáneamente. Sin embargo, el esclavo debe esperar ambos handshakes antes de ejecutar la escritura.

5.2. Secuencia de lectura

Una transacción de lectura en AXI4-Lite es más simple que escritura (solo 2 canales):

1. **Paso 1:** Maestro presenta dirección en canal AR
2. **Paso 2:** Esclavo acepta dirección
3. **Paso 3:** Esclavo procesa la lectura internamente
4. **Paso 4:** Esclavo envía dato y respuesta en canal R
5. **Paso 5:** Maestro acepta dato

En la Figura 3 se muestra un diagrama con ambas secuencias.

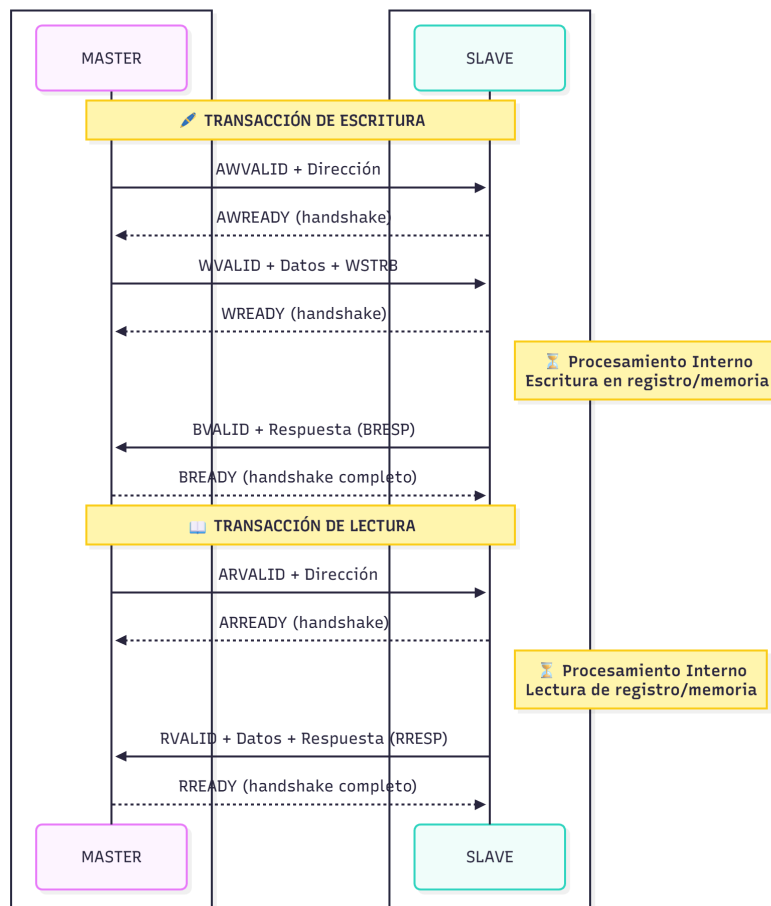


Figura 3: Protocolo AXI4-Lite - Transacciones de Escritura y Lectura

6. Códigos de Respuesta

6.1. Tipos de respuesta

AXI4-Lite define tres códigos de respuesta en las señales **BRESP[1:0]** (escritura) y **RRESP[1:0]** (lectura):

Código	Valor	Nombre	Descripción
OKAY	2' b00	Éxito	La transacción se completó exitosamente.
EXOKAY	2' b01	Exclusivo OK	(No usado en AXI4-Lite) Transacción exclusiva exitosa.
SLVERR	2' b10	Error del esclavo	El esclavo detectó un error (ej. dirección válida pero operación no permitida).
DECERR	2' b11	Error de decodificación	La dirección no corresponde a ningún registro del esclavo.

Nota: AXI4-Lite no soporta transacciones exclusivas, por lo que EXOKAY nunca debe usarse. Los módulos IP desarrollados solo utilizan OKAY, SLVERR y DECERR.

6.2. Interpretación de respuestas en el contexto del proyecto

OKAY (2'b00):

- Usado cuando la dirección es válida y está dentro del mapa de registros del esclavo
- La operación (lectura/escritura) se ejecutó correctamente
- Ejemplo: Escribir en registro de control, leer dato de PRNG

SLVERR (2'b10):

- Usado cuando la dirección es válida pero la operación no está permitida
- Ejemplos en el proyecto:
 - Intentar escribir en un registro de solo lectura (como dato de salida del TRNG)
 - Parámetro fuera de rango permitido

DECERR (2'b11):

- Usado cuando la dirección está fuera del rango de registros del esclavo
- Ejemplo: Intentar acceder a dirección 0x1000 cuando el módulo solo tiene registros en 0x00-0x0F

7. Señales Específicas de AXI4-Lite

7.1. Tabla de señales

La siguientes tablas presentan todas las señales de la interfaz AXI4-Lite, organizadas por canal:

7.1.1. Señales Globales

Señal	Ancho	Fuente	Descripción
ACLK	1	Sistema	Reloj del bus AXI
ARESETn	1	Sistema	Reset activo en bajo

7.1.2. Canal Write Address (AW)

Señal	Ancho	Fuente	Descripción
AWADDR	32*	Maestro	Dirección de escritura
AWPROT	3	Maestro	Indicadores de protección
AWVALID	1	Maestro	Dirección de escritura válida
AWREADY	1	Esclavo	Listo para aceptar dirección

*Ancho configurable, típicamente 32 bits

7.1.3. Canal Write Data (W)

Señal	Ancho	Fuente	Descripción
WDATA	32*	Maestro	Dato de escritura
WSTRB	4*	Maestro	Byte enable (máscara)
WVALID	1	Maestro	Dato de escritura válido
WREADY	1	Esclavo	Listo para aceptar dato

*Anchos relacionados: WSTRB = WDATA/8 bytes

7.1.4. Canal Write Response (B)

Señal	Ancho	Fuente	Descripción
BRESP	2	Esclavo	OKAY/SLVERR/DECERR
BVALID	1	Esclavo	Respuesta válida
BREADY	1	Maestro	Listo para aceptar respuesta

7.1.5. Canal Read Address (AR)

Señal	Ancho	Fuente	Descripción
ARADDR	32*	Maestro	Dirección de lectura
ARPROT	3	Maestro	Indicadores de protección
ARVALID	1	Maestro	Dirección de lectura válida
ARREADY	1	Esclavo	Listo para aceptar dirección

7.1.6. Canal Read Data (R)

Señal	Ancho	Fuente	Descripción
RDATA	32*	Esclavo	Dato de lectura
RRESP	2	Esclavo	OKAY/SLVERR/DECERR
RVALID	1	Esclavo	Dato de lectura válido
RREADY	1	Maestro	Listo para aceptar dato

Nota: Para más detalles revisar el Apéndice A de [2]

7.2. Señales AWPROT y ARPROT

Las señales de protección AWPROT[2:0] y ARPROT[2:0] indican características de la transacción:

Bit	Nombre	Valores	Descripción
[0]	Privilegio	0: Normal 1: Privilegiado	Indica si la transacción es privilegiada
[1]	Seguridad	0: Segura 1: No segura	Indica nivel de seguridad de la transacción
[2]	Acceso	0: Dato 1: Instrucción	Indica si es acceso a dato o instrucción

Nota: En los módulos IP desarrollados, estas señales típicamente se ignoran ya que el acceso a registros es siempre permitido desde MicroBlaze V. Sin embargo, se mantienen en la interfaz para cumplir con el estándar.

7.3. Señal WSTRB (Write Strobes)

WSTRB[N-1:0] es una máscara de bits que indica qué bytes de WDATA son válidos:

Para WDATA de 32 bits (4 bytes):

WSTRB	Bytes válidos	Descripción
4'b0001	Byte 0	Solo byte menos significativo
4'b0010	Byte 1	Segundo byte
4'b0100	Byte 2	Tercer byte
4'b1000	Byte 3	Byte más significativo
4'b0011	Bytes 0-1	Half-word inferior (16 bits)
4'b1100	Bytes 2-3	Half-word superior (16 bits)
4'b1111	Bytes 0-3	Word completo (32 bits)

Nota: MicroBlaze V típicamente genera WSTRB = 4'b1111 (escritura completa de 32 bits), pero es importante soportar escrituras parciales para cumplir con el estándar.

8. Arquitectura del Procesador MicroBlaze V

8.1. Descripción general

MicroBlaze V es un procesador soft-core desarrollado por AMD/Xilinx, basado en la arquitectura RISC-V. Está optimizado para implementación en FPGAs de AMD y es el sucesor del MicroBlaze clásico (arquitectura propietaria).

Características principales:

- **ISA:** RISC-V RV32I (base) + extensiones configurables (M, C, etc.)
- **Pipeline:** 3-stage (Fetch, Decode, Execute) o 5-stage (configurable)
- **Frecuencia:** ~100 MHz en Artix-7
- **Bus principal:** AXI4 para memoria e instrucciones
- **Interfaces periféricos:** AXI4-Lite para acceso a registros

8.2. MicroBlaze V como maestro AXI

En el contexto de este proyecto, MicroBlaze V actúa como maestro AXI, generando transacciones de lectura/escritura hacia los módulos IP AXI-slave desarrollados.

Interfaces AXI del MicroBlaze V:

Interfaz	Tipo	Propósito	Características
M_AXI_DP	AXI4	Data Port (memoria de datos)	Burst, hasta 16 transacciones pendientes
M_AXI_DC	AXI4	Data Cache	Burst, coherencia de caché
M_AXI_IC	AXI4	Instruction Cache	Burst, fetch de instrucciones
M_AXI_PERIPH	AXI4-Lite	Periféricos	Usado para conectar los IPs desarrollados

Nota: Los módulos IP desarrollados se conectan a la interfaz M_AXI_PERIPH (AXI4-Lite) del MicroBlaze V, que está dedicada a acceso a periféricos mediante MMIO.

8.3. Mapa de memoria del MicroBlaze V

MicroBlaze V utiliza un espacio de direcciones de 32 bits (4 GB) con las siguientes regiones típicas:

Región	Rango de Direcciones	Uso
Memoria de programa	0x00000000 - 0x1FFFFFFF	Instrucciones (código)
Memoria de datos	0x20000000 - 0x3FFFFFFF	Datos del programa
Periféricos	0x40000000 - 0x7FFFFFFF	Registros MMIO (IPs AXI-slave)
Memoria externa	0x80000000 - 0xFFFFFFFF	DDR, Flash, etc.

Los módulos IP desarrollados se mapean típicamente en la región de periféricos (0x40000000+), con un offset específico para cada módulo:

Ejemplo de asignación de direcciones:

0x44a00000 - 0x44a0ffff : AXI TRNG
0x44a10000 - 0x44a1ffff : AXI 7-Segment

Importante: Las direcciones deben estar alineadas al tamaño del dato (para accesos de 32 bits, direcciones múltiplo de 4). AXI4-Lite requiere que $AWADDR[1:0] = 00$ y $ARADDR[1:0] = 00$.

8.4. Integración con AXI Interconnect

En un diseño típico con MicroBlaze V, se utiliza un AXI SmartConnect para conectar el maestro (MicroBlaze V) con múltiples esclavos (módulos IP). Ver Figura 4 y 5:

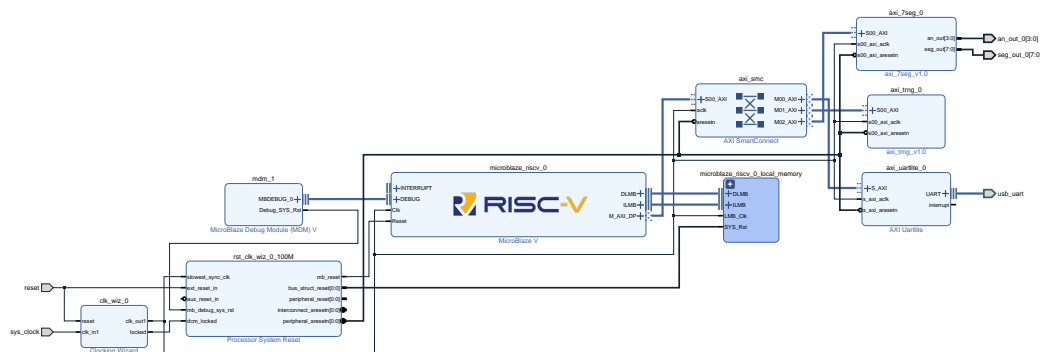


Figura 4: Block Design con los módulos axi_trng_v1.0 y axi_7seg_v1.0

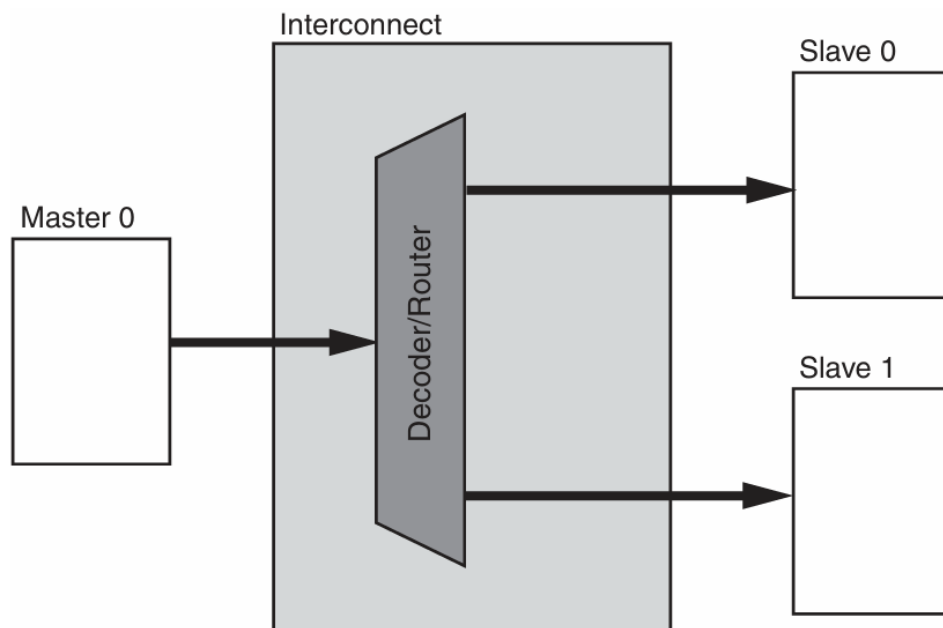


Figura 5: Interconexión AXI de 1 a N [2]

El AXI SmartConnect realiza:

1. **Decodificación de direcciones:** Rutea transacciones al esclavo correcto
2. **Conversión de protocolo:** Puede convertir AXI4 → AXI4-Lite si es necesario
3. **Arbitraje:** Si hay múltiples maestros (no aplica en este proyecto)
4. **Generación de DECERR:** Si la dirección no coincide con ningún esclavo

9. Consideraciones de Diseño para Módulos AXI-Slave

9.1. Requerimientos de temporización y Manejo de reset

Los módulos AXI-slave deben cumplir con las restricciones de timing del sistema. En este caso la frecuencia es de 100 MHz para la Basys 3 con Artix-7.

El reset (ARESETn) debe inicializar correctamente todos los registros.

9.2. Mapa de registros (Register Map)

Cada módulo IP debe definir claramente su mapa de registros:

Ejemplo: AXI PRNG

Offset	Nombre	Descripción
0x00	CONTROL	Control
0x04	SAMPLE_DIV	Divisor de muestras
0x08	RANGE_LOW	Límite inferior para salida escalada
0x0C	RANGE_HIGH	Límite superior para salida escalada
0x10	RANDOM_RAW	Número aleatorio crudo
0x14	RANDOM_IN_RANGE	Número aleatorio escalado
0x18	STATUS	Estado

10. Aplicación en los Módulos IP Desarrollados

10.1. Módulos implementados

En el contexto de este proyecto, se han desarrollado los siguientes módulos IP AXI-slave:

Módulo	Función	Registros	Aplicación
axi_pulsegen	Generador de pulsos	4	Temporizadores, control de actuadores, etc
axi_7seg	Display 7 segmentos	4	Visualización de datos en Basys 3
axi_prng	Generador pseudo-aleatorio	8	Números aleatorios deterministas
axi_trng	Generador verdadero aleatorio	8	Entropía para criptografía
axi_perceptron	Perceptrón binario	8	Machine learning básico

Todos estos módulos utilizan AXI4-Lite como interfaz de comunicación con el procesador MicroBlaze V.

10.2. Flujo de integración en Vivado y Vitis

Nota: Ver Manual de Usuario.

Referencias

- [1] ARM Limited, "AMBA AXI and ACE Protocol Specification," [En línea]. Disponible en: <https://developer.arm.com/documentation/ih0022/h>.
- [2] AMD Xilinx, "Vivado Design Suite AXI Reference Guide", User Guide UG1037, 2025. [En línea]. Disponible en: <https://docs.amd.com/v/u/en-US/ug1037-vivado-axi-reference-guide>.
- [3] AMD Xilinx, "MicroBlaze V Embedded Design," User Guide UG1711, 2025. [En línea]. Disponible en: <https://docs.amd.com/r/en-US/ug1711-microblaze-v-embedded-design/>.
- [4] AMD Xilinx, "MicroBlaze V User Guide," User Guide UG1629, 2025. [En línea]. Disponible en: <https://docs.amd.com/r/en-US/ug1629-microblaze-v-user-guide/>.