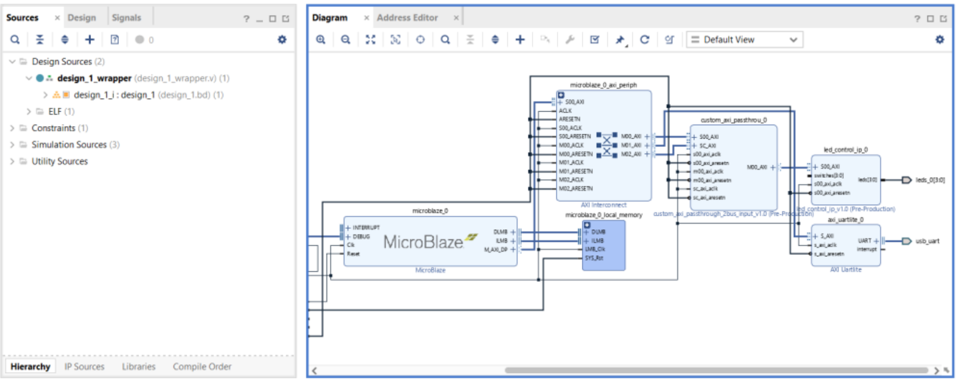
==HOW TO USE==

You should refer to the guide for the original version of this IP: https://github.com/NinetyTwo-TwentyNine/Vivado-AXI-Data-Passthrough-And-Edit-IP/tree/main/doc

This file contains all of the changes related to usage of this version of the IP.

1. Main application circuit:

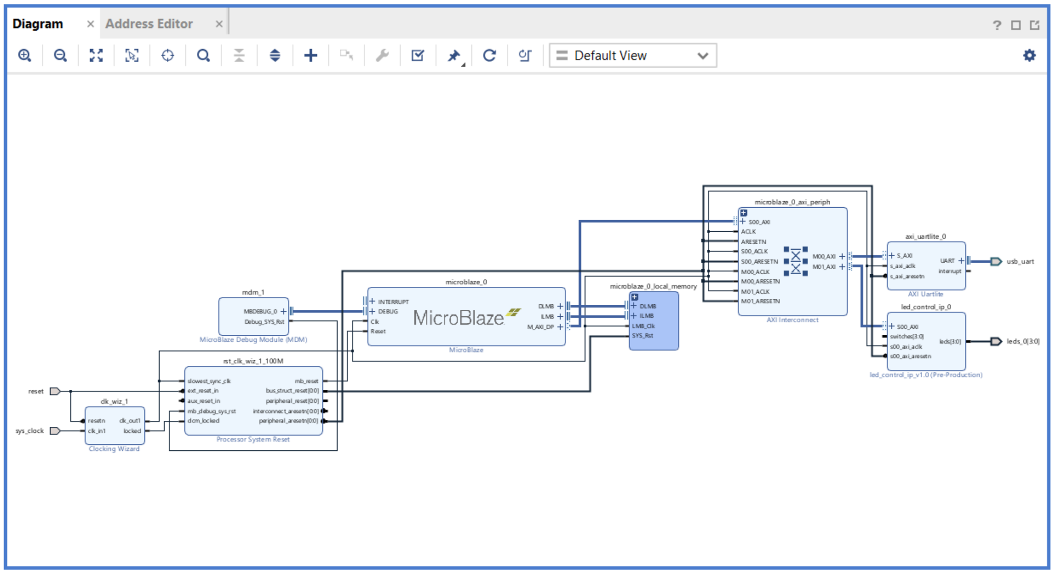


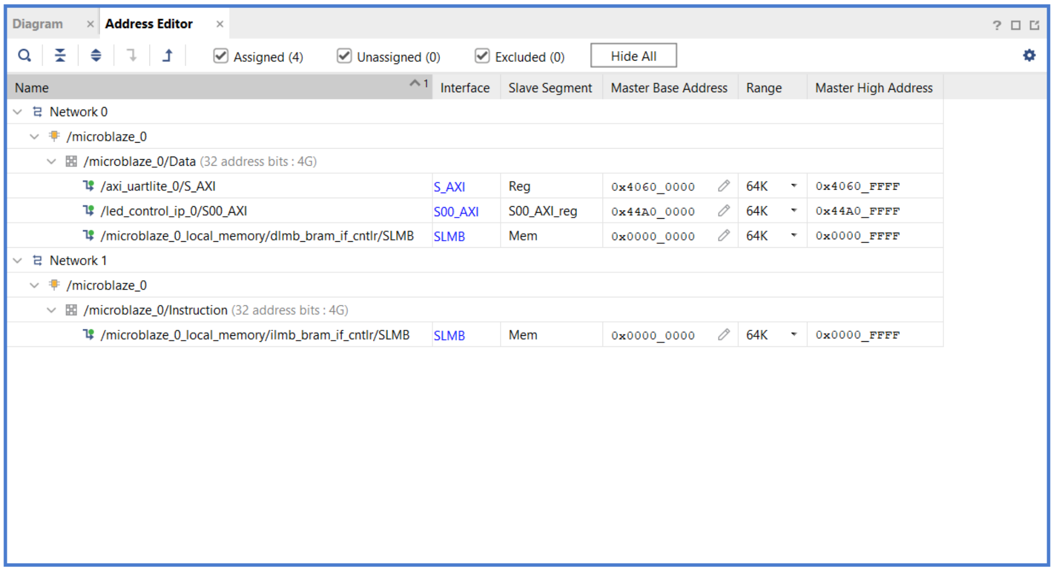
The main purpose of this component is to allow data editing while it passes the AXI bus (which could be useful in testing/debugging, for example).

The main way to use this IP is to put it in between any single AXI-Lite master-slave connection (mainly, a microprocessor and its peripheral). You also now always have to connect to the «SC AXI» bus, as it is the IP programming bus. You could theoretically also use this version of the IP for the same purpose which the original IP was used for (being placed in between an MP and its interconnect, that is), however, the 1-bus input version is much more accustomed to that role.

2. Proper IP setup in the UI and address editor:

An example circuit with some AXI peripherals

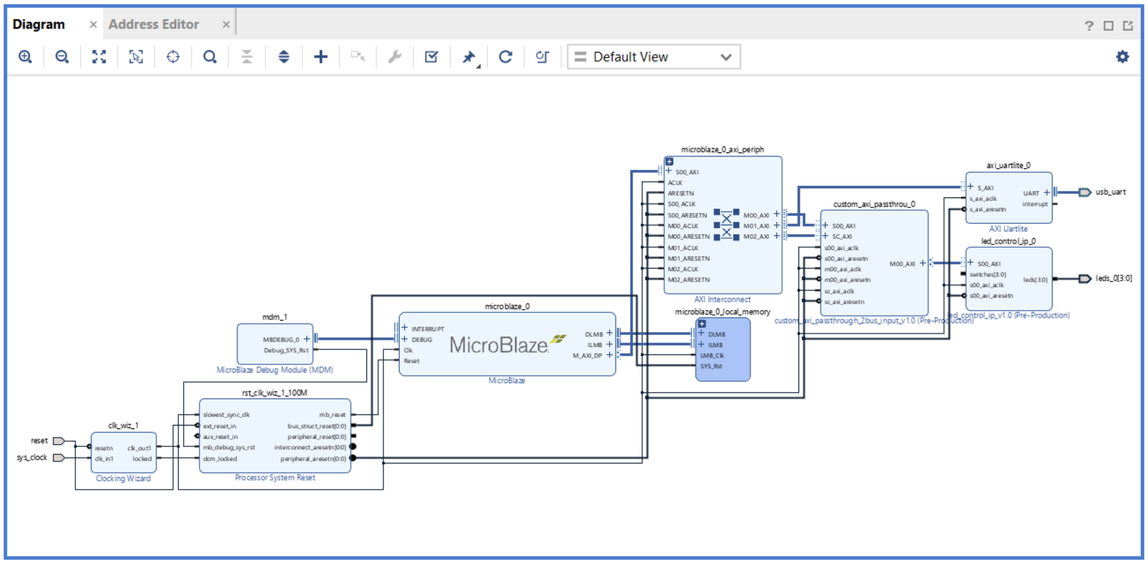


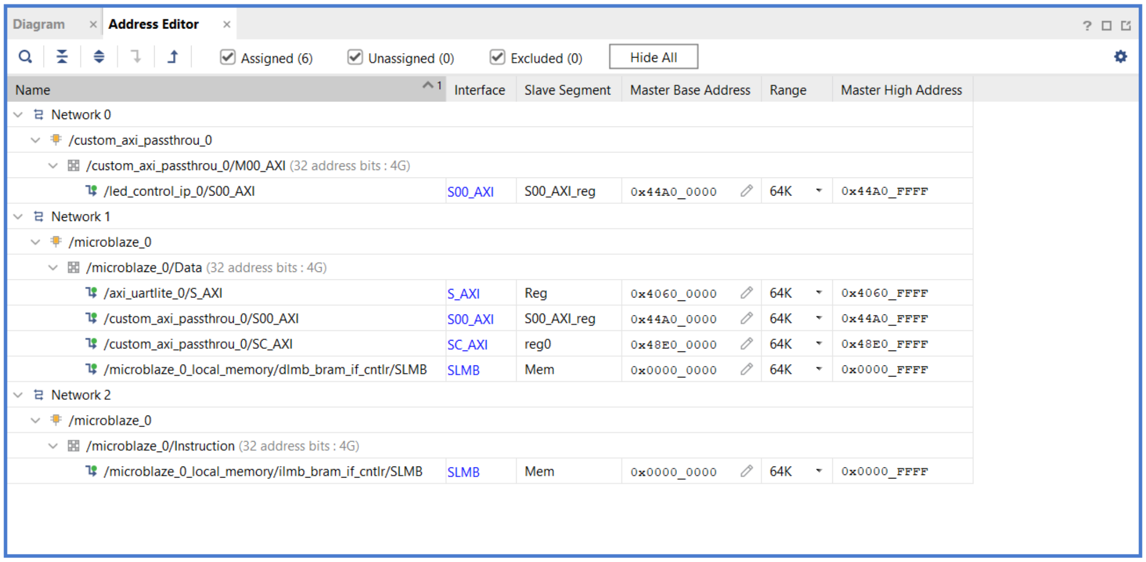


Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/microblaze_0					
/microblaze_0/Data (32 address bits : 4G)					
/axi_uartlite_0/S_AXI	S_AXI	Reg	0x4060_0000	64K	0x4060_FFFF
/led_control_ip_0/S00_AXI	S00_AXI	S00_AXI_reg	0x44A0_0000	64K	0x44A0_FFFF
/microblaze_0_local_memory/dlmb_bram_if_cntlr/SLMB	SLMB	Mem	0x0000_0000	64K	0x0000_FFFF
Network 1					
/microblaze_0					
/microblaze_0/Instruction (32 address bits : 4G)					
/microblaze_0_local_memory/ilmb_bram_if_cntlr/SLMB	SLMB	Mem	0x0000_0000	64K	0x0000_FFFF

Note: this is the point at which you should already (if you will eventually need to) export your XSA-file from the project (before the IP was added). This will be explained later in the tutorial.

That same circuit after proper addition of this IP

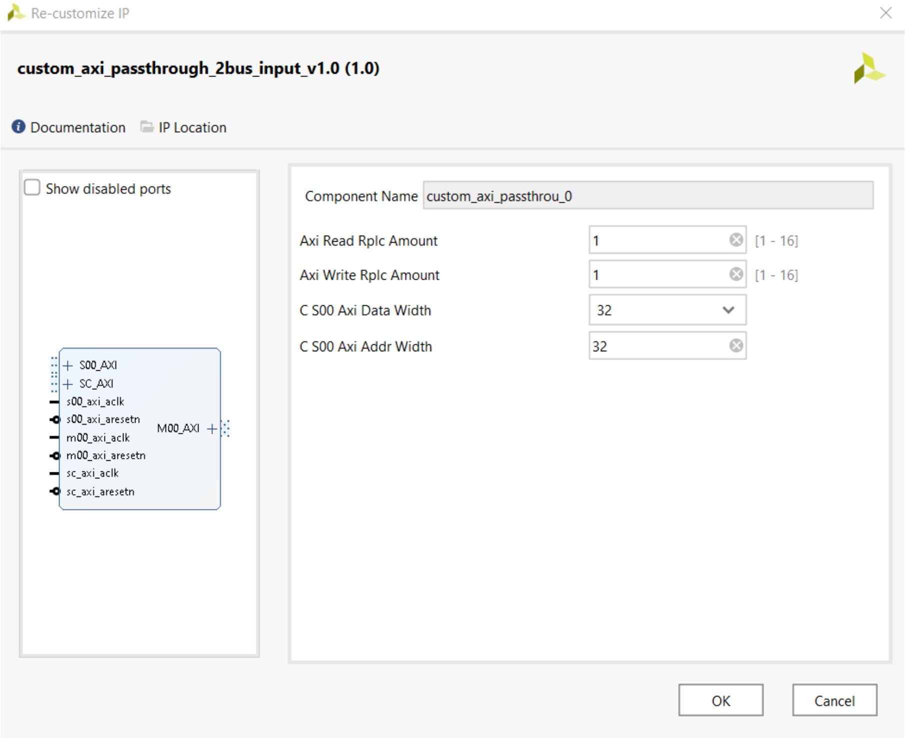




Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/custom_axi_passthrou_0					
/custom_axi_passthrou_0/M00_AXI (32 address bits : 4G)					
/led_control_ip_0/S00_AXI	S00_AXI	S00_AXI_reg	0x44A0_0000	64K	0x44A0_FFFF
Network 1					
/microblaze_0					
/microblaze_0/Data (32 address bits : 4G)					
/axi_uartlite_0/S_AXI	S_AXI	Reg	0x4060_0000	64K	0x4060_FFFF
/custom_axi_passthrou_0/S00_AXI	S00_AXI	S00_AXI_reg	0x44A0_0000	64K	0x44A0_FFFF
/custom_axi_passthrou_0/SC_AXI	SC_AXI	reg0	0x48E0_0000	64K	0x48E0_FFFF
/microblaze_0_local_memory/dlmb_bram_if_cntlr/SLMB	SLMB	Mem	0x0000_0000	64K	0x0000_FFFF
Network 2					
/microblaze_0					
/microblaze_0/Instruction (32 address bits : 4G)					
/microblaze_0_local_memory/ilmb_bram_if_cntlr/SLMB	SLMB	Mem	0x0000_0000	64K	0x0000_FFFF

The addresses for the IP’s «S00 AXI» bus must be set the same as the address of the peripheral you want to replace data for.

An example of UI setup



custom_axi_passthrough_2bus_input_v1.0 (1.0)

□ Show disabled ports

Component Name custom_axi_passthrou_0

Axi Read Rplc Amount	1	[1 - 16]
Axi Write Rplc Amount	1	[1 - 16]
C S00 Axi Data Width	32	
C S00 Axi Addr Width	32	

Since this version of the IP now has a separate bus for its programming, the «C S00 AXI Baseaddr» parameter no longer exists and is no longer required, as all you have to do now is to just write to this separate bus («SC AXI») and its address.

3. Working with Vitis. Programming the IP:

Paragraph 3 remains exactly the same as it is in the guide for the original IP.