Adding Custom IP to DMA

LECTURE 12

IOULIIA SKLIAROVA

Universidade de Aveiro

Examples (DMA Coprocessors)

Reverse endianness with DMA

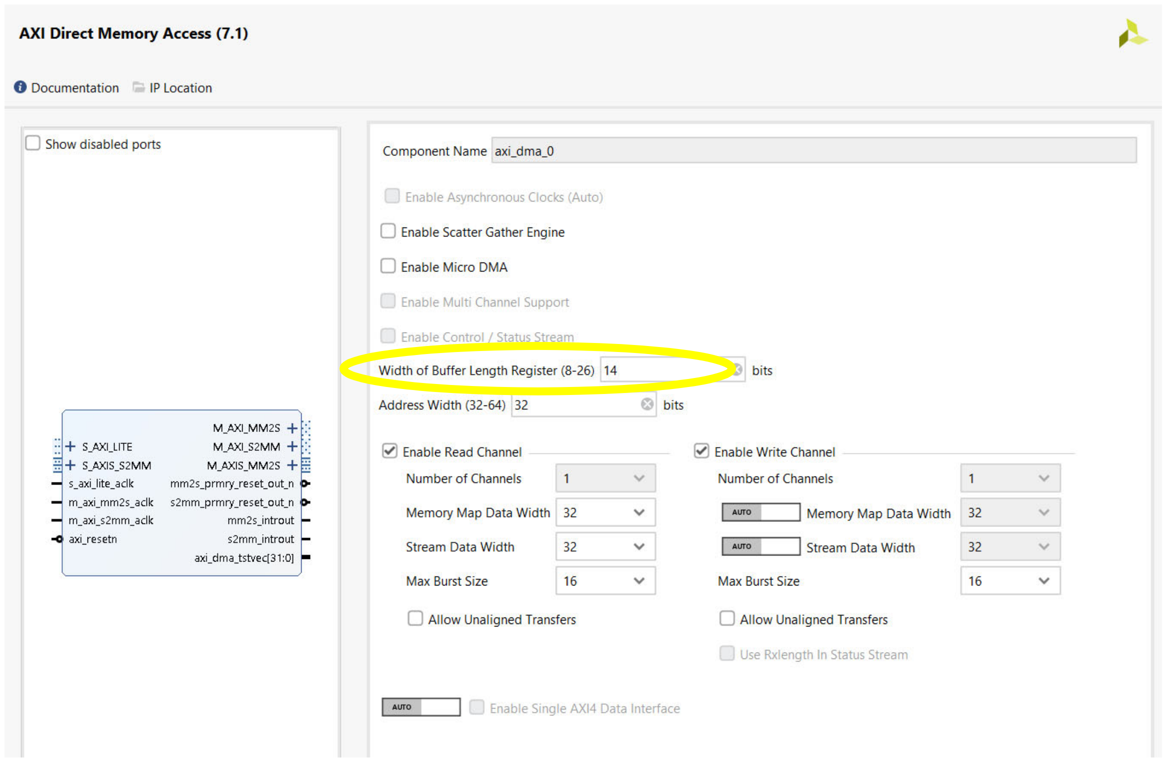
Population count (Hamming weight) with DMA

Example 1 – Starting Point

Create a new project

Import the BD from the previous class (with EMC, DMA, FIFO (depth=4096)…)

In the DMA you can specify the maximum transfer length (to 2^{14}-1 = 16_383 Bytes):


AXI Direct Memory Access (7.1)

Documentation   IP Location

Show disabled ports

Component Name axi_dma_0

- [ ] Enable Asynchronous Clocks (Auto)
- [ ] Enable Scatter Gather Engine
- [ ] Enable Micro DMA
- [ ] Enable Multi Channel Support
- [ ] Enable Control / Status Stream

Width of Buffer Length Register (8-26) 14 bits

Address Width (32-64) 32 bits

✓ Enable Read Channel		✓ Enable Write Channel	
Number of Channels	1	Number of Channels	1
Memory Map Data Width	32	AUTO Memory Map Data Width	32
Stream Data Width	32	AUTO Stream Data Width	32
Max Burst Size	16	Max Burst Size	16
☐ Allow Unaligned Transfers		☐ Allow Unaligned Transfers	
		☐ Use Rxlength In Status Stream	

AUTO ☐ Enable Single AXI4 Data Interface

Adding the IP and Further Steps

Create a new AXI-Stream connected IP (ReverseEndiannessCop) but use a new (modified) VHDL code (available on eLearning)

- processes TLAST and TSTRB signals. Why?

Connect FIFO M_AXIS to **ReverseEndiannessCop** S00_AXIS

Connect **ReverseEndiannessCop** M00_AXIS to DMA S_AXIS_S2MM

Run Connection Automation

Generate Output Products

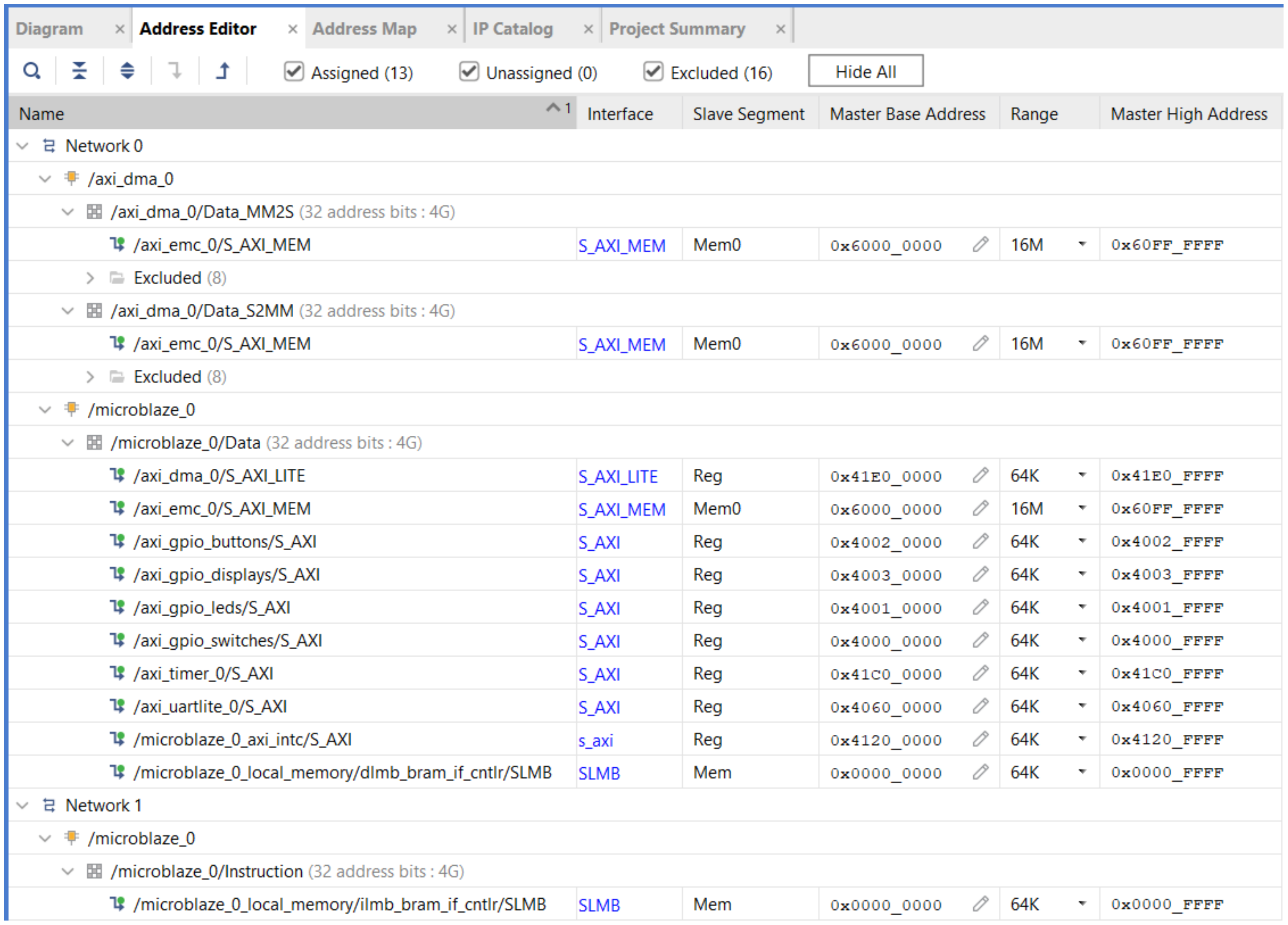
Create HDL Wrapper

Generate Bitstream

Export Hardware (DMA_ReverseEndianness.xsa)

Launch Vitis

Address Editor



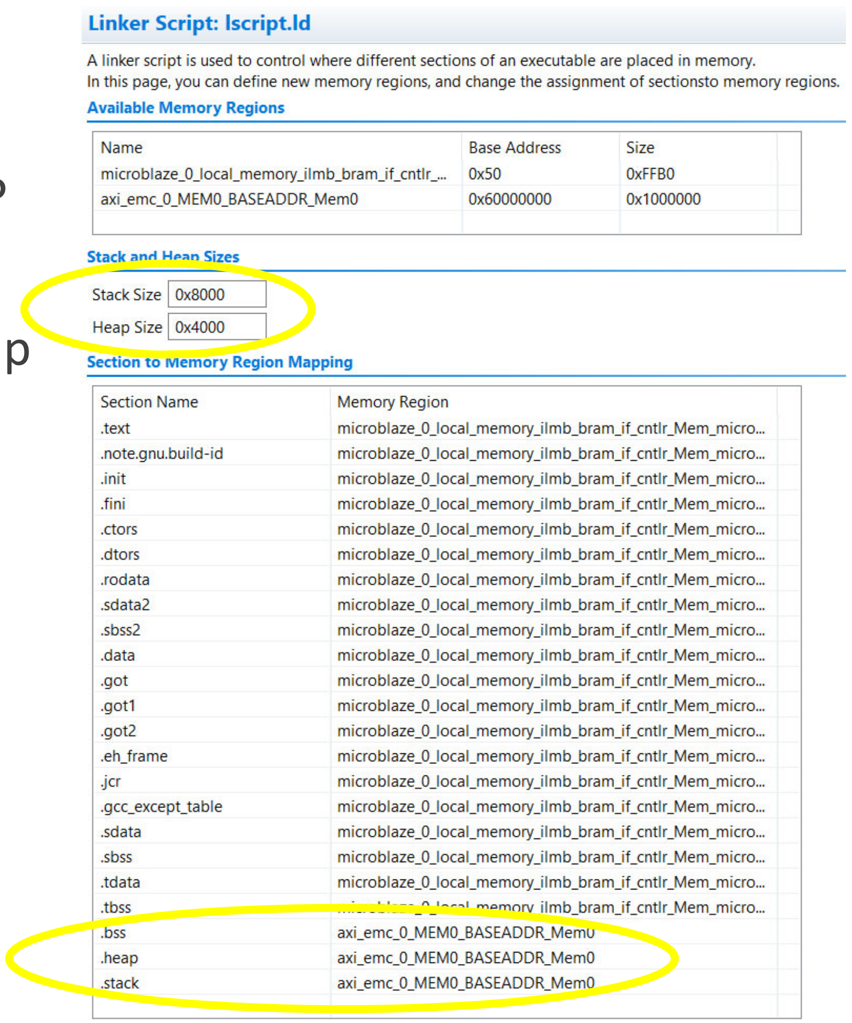
Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/axi_dma_0					
/axi_dma_0/Data_MM2S (32 address bits : 4G)					
/axi_emc_0/S_AXI_MEM	S_AXI_MEM	Mem0	0x6000_0000	16M	0x60FF_FFFF
Excluded (8)					
/axi_dma_0/Data_S2MM (32 address bits : 4G)					
/axi_emc_0/S_AXI_MEM	S_AXI_MEM	Mem0	0x6000_0000	16M	0x60FF_FFFF
Excluded (8)					
/microblaze_0					
/microblaze_0/Data (32 address bits : 4G)					
/axi_dma_0/S_AXI_LITE	S_AXI_LITE	Reg	0x41E0_0000	64K	0x41E0_FFFF
/axi_emc_0/S_AXI_MEM	S_AXI_MEM	Mem0	0x6000_0000	16M	0x60FF_FFFF
/axi_gpio_buttons/S_AXI	S_AXI	Reg	0x4002_0000	64K	0x4002_FFFF
/axi_gpio_displays/S_AXI	S_AXI	Reg	0x4003_0000	64K	0x4003_FFFF
/axi_gpio_leds/S_AXI	S_AXI	Reg	0x4001_0000	64K	0x4001_FFFF
/axi_gpio_switches/S_AXI	S_AXI	Reg	0x4000_0000	64K	0x4000_FFFF
/axi_timer_0/S_AXI	S_AXI	Reg	0x41C0_0000	64K	0x41C0_FFFF
/axi_uartlite_0/S_AXI	S_AXI	Reg	0x4060_0000	64K	0x4060_FFFF
/microblaze_0_axi_intc/S_AXI	s_axi	Reg	0x4120_0000	64K	0x4120_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

Vitis

The C code is given on eLearning (DMAEndianness.c)

There is no need to correct the IP makefiles

Configure the right stack and heap size in the linker script and map memories as indicated:



Linker Script: lscript.ld

A linker script is used to control where different sections of an executable are placed in memory.
In this page, you can define new memory regions, and change the assignment of sectionsto memory regions.

Available Memory Regions

Name	Base Address	Size
microblaze_0_local_memory_ilmb_bram_if_cntlr_...	0x50	0xFFB0
axi_emc_0_MEM0_BASEADDR_Mem0	0x60000000	0x1000000

Stack and Heap Sizes

Stack Size	0x8000
Heap Size	0x4000

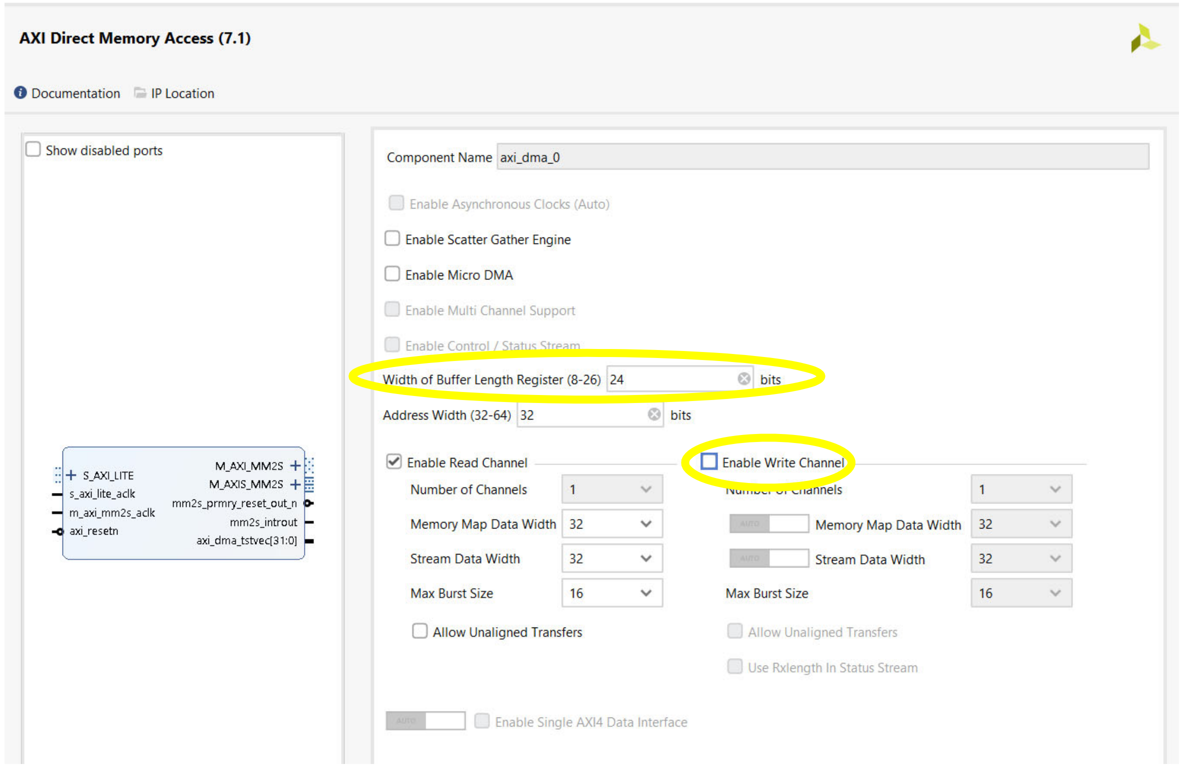
Section to Memory Region Mapping

Section Name	Memory Region
.text	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.note.gnu.build-id	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.init	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.fini	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.ctors	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.dtors	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.rodata	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sdata2	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sbss2	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.data	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.got	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.got1	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.got2	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.eh_frame	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.jcr	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.gcc_except_table	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sdata	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sbss	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.tdata	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.tbss	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.bss	axi_emc_0_MEM0_BASEADDR_Mem0
.heap	axi_emc_0_MEM0_BASEADDR_Mem0
.stack	axi_emc_0_MEM0_BASEADDR_Mem0

Example 2 – Starting Point

Create a new project and import the BD from the previous class (having EMC, DMA, FIFO...)

Configure the DMA maximum transfer length (to 2^{24}-1 = 16_777_215 Bytes) and disable the write channel:



AXI Direct Memory Access (7.1)

Documentation IP Location

☐ Show disabled ports

Component Name axi_dma_0

☐ Enable Asynchronous Clocks (Auto)
☐ Enable Scatter Gather Engine
☐ Enable Micro DMA
☐ Enable Multi Channel Support
☐ Enable Control / Status Stream

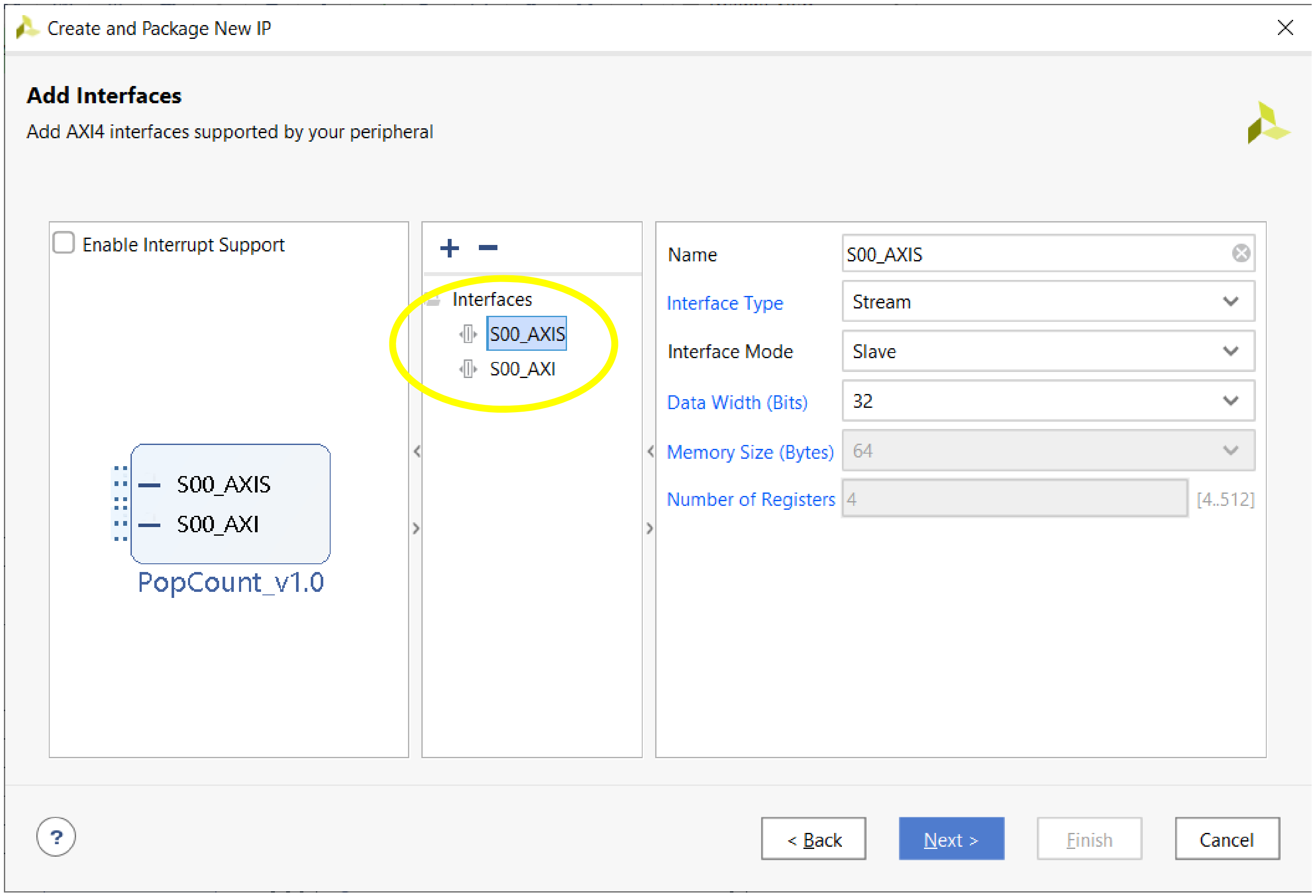
Width of Buffer Length Register (8-26) 24 bits

Address Width (32-64) 32 bits

✓ Enable Read Channel		☐ Enable Write Channel	
Number of Channels	1	Number of Channels	1
Memory Map Data Width	32	Memory Map Data Width	32
Stream Data Width	32	Stream Data Width	32
Max Burst Size	16	Max Burst Size	16
☐ Allow Unaligned Transfers		☐ Allow Unaligned Transfers	
		☐ Use Rxlength In Status Stream	

☐ Enable Single AXI4 Data Interface

Create and Package New IP (PopCount)



Create and Package New IP

Add Interfaces

Add AXI4 interfaces supported by your peripheral

☐ Enable Interrupt Support

Name	S00_AXIS
Interface Type	Stream
Interface Mode	Slave
Data Width (Bits)	32
Memory Size (Bytes)	64
Number of Registers	4 [4..512]

Adding the IP and Further Steps

Add the PopCount IP to the block diagram

Connect FIFO M_AXIS to PopCount S00_AXIS

Connect PopCount S00_AXI to Interconnect M_09_AXI (add one more master port)

Run Connection Automation

Modify the PopCount code

Assign PopCount in the address editor

Generate Output Products

Create HDL Wrapper

Generate Bitstream

Export Hardware (DMA_PopCount.xsa)

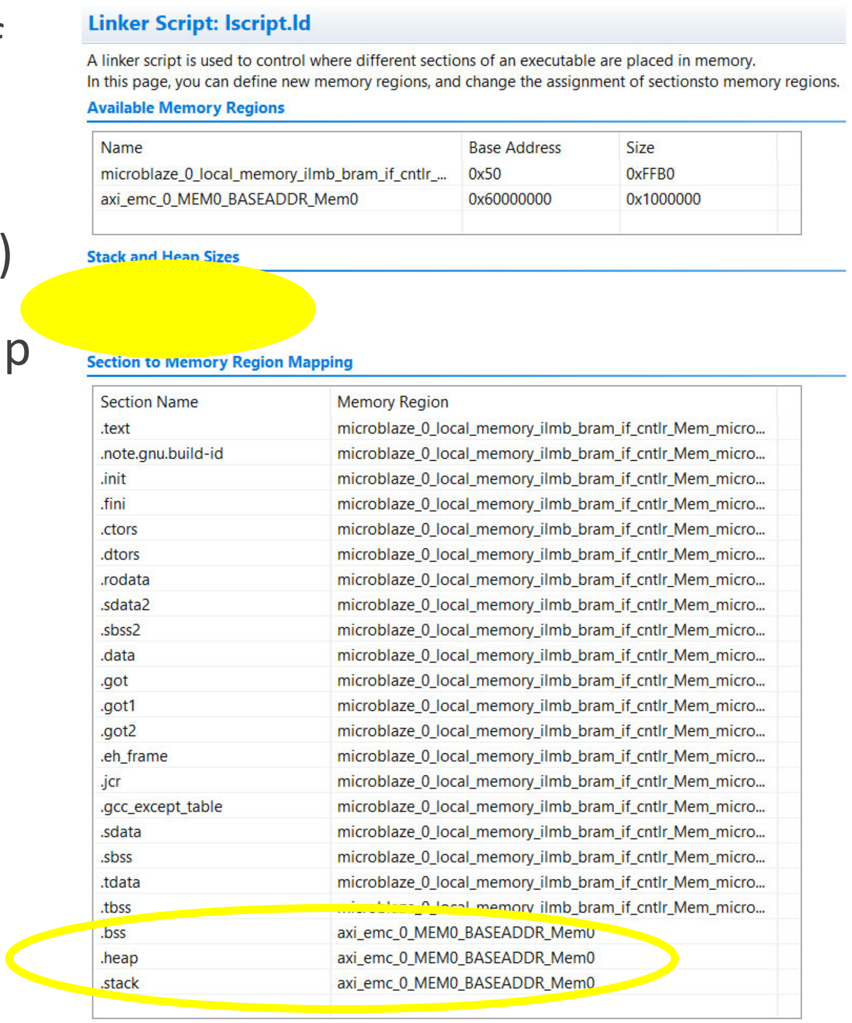
Launch Vitis

Vitis

Design the C code on the basis of the example given on eLearning (DMAEndianness.c)

Correct the IP makefiles (AXI-Lite)

Configure the right stack and heap size in the linker script and map memories as indicated:



Linker Script: lscript.ld

A linker script is used to control where different sections of an executable are placed in memory.
In this page, you can define new memory regions, and change the assignment of sectionsto memory regions.

Available Memory Regions

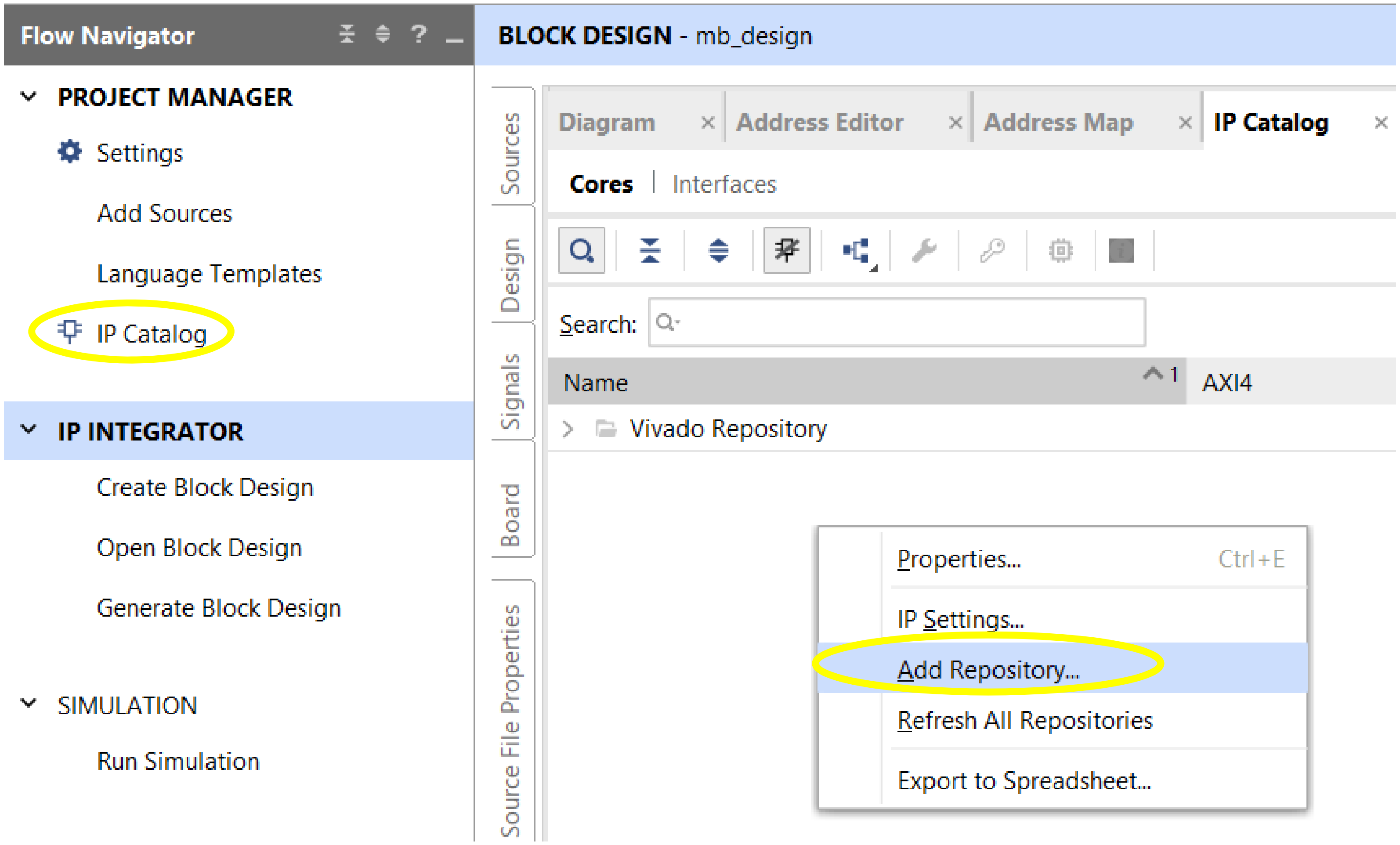
Name	Base Address	Size
microblaze_0_local_memory_ilmb_bram_if_cntlr_...	0x50	0xFFB0
axi_emc_0_MEM0_BASEADDR_Mem0	0x60000000	0x1000000

Stack and Heap Sizes

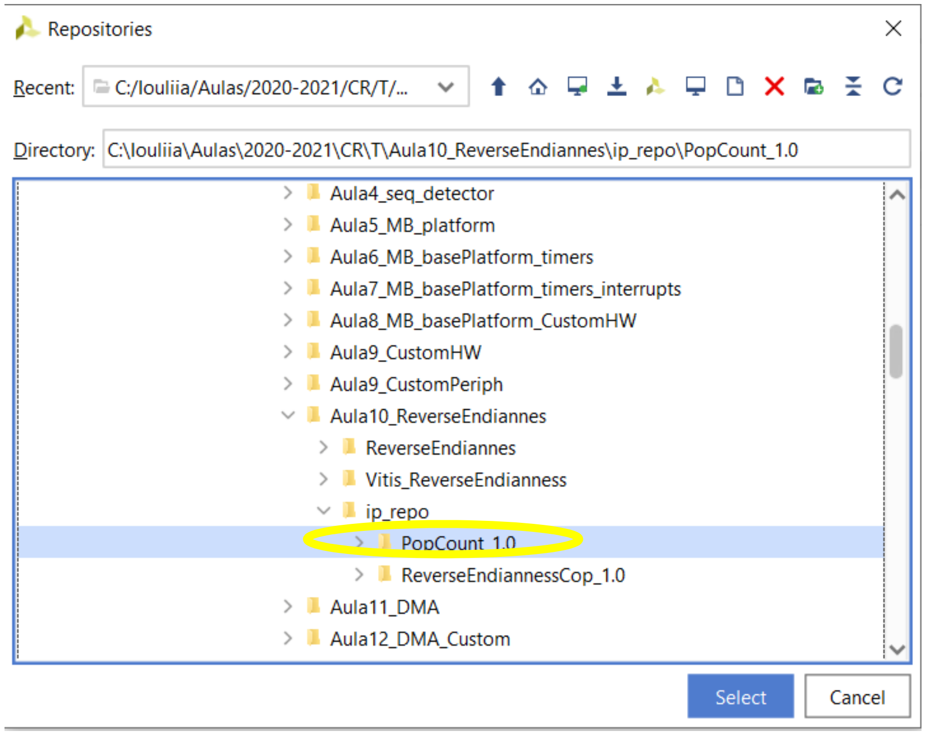
Section to Memory Region Mapping

Section Name	Memory Region
.text	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.note.gnu.build-id	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.init	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.fini	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.ctors	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.dtors	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.rodata	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sdata2	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sbss2	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.data	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.got	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.got1	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.got2	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.eh_frame	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.jcr	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.gcc_except_table	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sdata	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.sbss	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.tdata	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.tbss	microblaze_0_local_memory_ilmb_bram_if_cntlr_Mem_micro...
.bss	axi_emc_0_MEM0_BASEADDR_Mem0
.heap	axi_emc_0_MEM0_BASEADDR_Mem0
.stack	axi_emc_0_MEM0_BASEADDR_Mem0

Add a Repository



Add a Repository

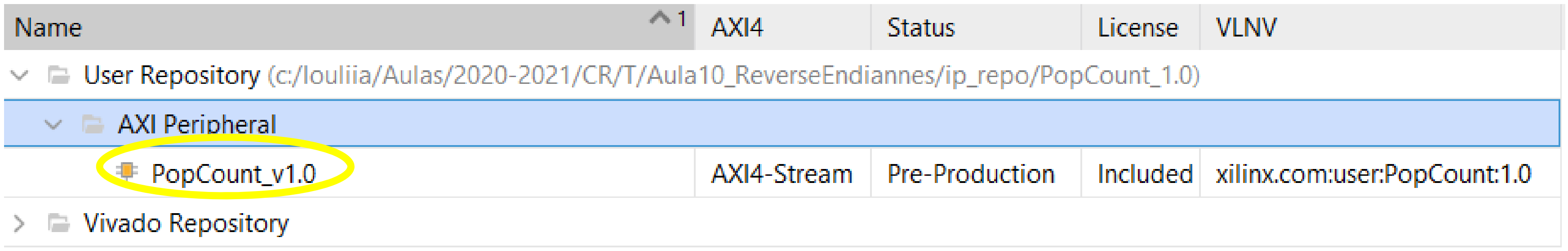


Repositories

Recent: C:/louliia/Aulas/2020-2021/CR/T/...

Directory: C:\louliia\Aulas\2020-2021\CR\T\Aula10_ReverseEndiannes\ip_repo\PopCount_1.0

- Aula4_seq_detector
- Aula5_MB_platform
- Aula6_MB_basePlatform_timers
- Aula7_MB_basePlatform_timers_interrupts
- Aula8_MB_basePlatform_CustomHW
- Aula9_CustomHW
- Aula9_CustomPeriph
- Aula10_ReverseEndiannes
  - ReverseEndiannes
  - Vitis_ReverseEndianness
  - ip_repo
    - PopCount_1.0
    - ReverseEndiannessCop_1.0
- Aula11_DMA
- Aula12_DMA_Custom



Name	AXI4	Status	License	VLNV
User Repository (c:/louliia/Aulas/2020-2021/CR/T/Aula10_ReverseEndiannes/ip_repo/PopCount_1.0)				
AXI Peripheral				
PopCount_v1.0	AXI4-Stream	Pre-Production	Included	xilinx.com:user:PopCount:1.0
Vivado Repository				

Final Remarks

At the end of this lecture you should be able to:

- Design custom hardware modules supporting DMA

To do:

- Construct the considered hardware platforms
- Test the given applications in Vitis