Projektiranje programabilnih SoC platformi

Predavanja FER, 2022.

VDMA

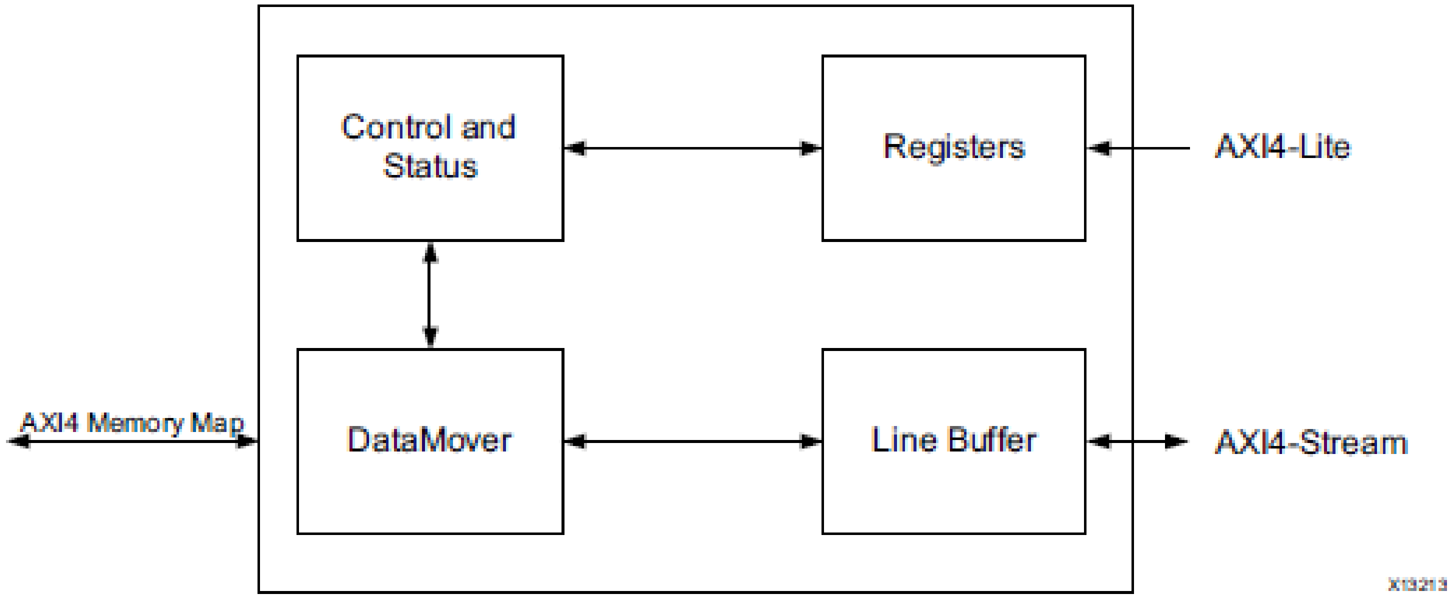
VDMA

- Specification:
  - https://docs.xilinx.com/r/en-US/pg020_axi_vdma

- Features:
  - AXI4 Compliant
  - Primary AXI4 data width support of 32, 64, 128, 256, 512, and 1,024 bits
  - Primary AXI4-Stream data width support of multiples of 8 up to 1,024 bits
  - Optional frame advance or repeat on error
  - Supports up to 32 frame buffers
  - Supports up to 64-bit address space
  - Supports Vertical Flip

VDMA

- The AXI VDMA is designed to allow for efficient high-bandwidth access between the AXI4-Stream video interface and the AXI4 interface.

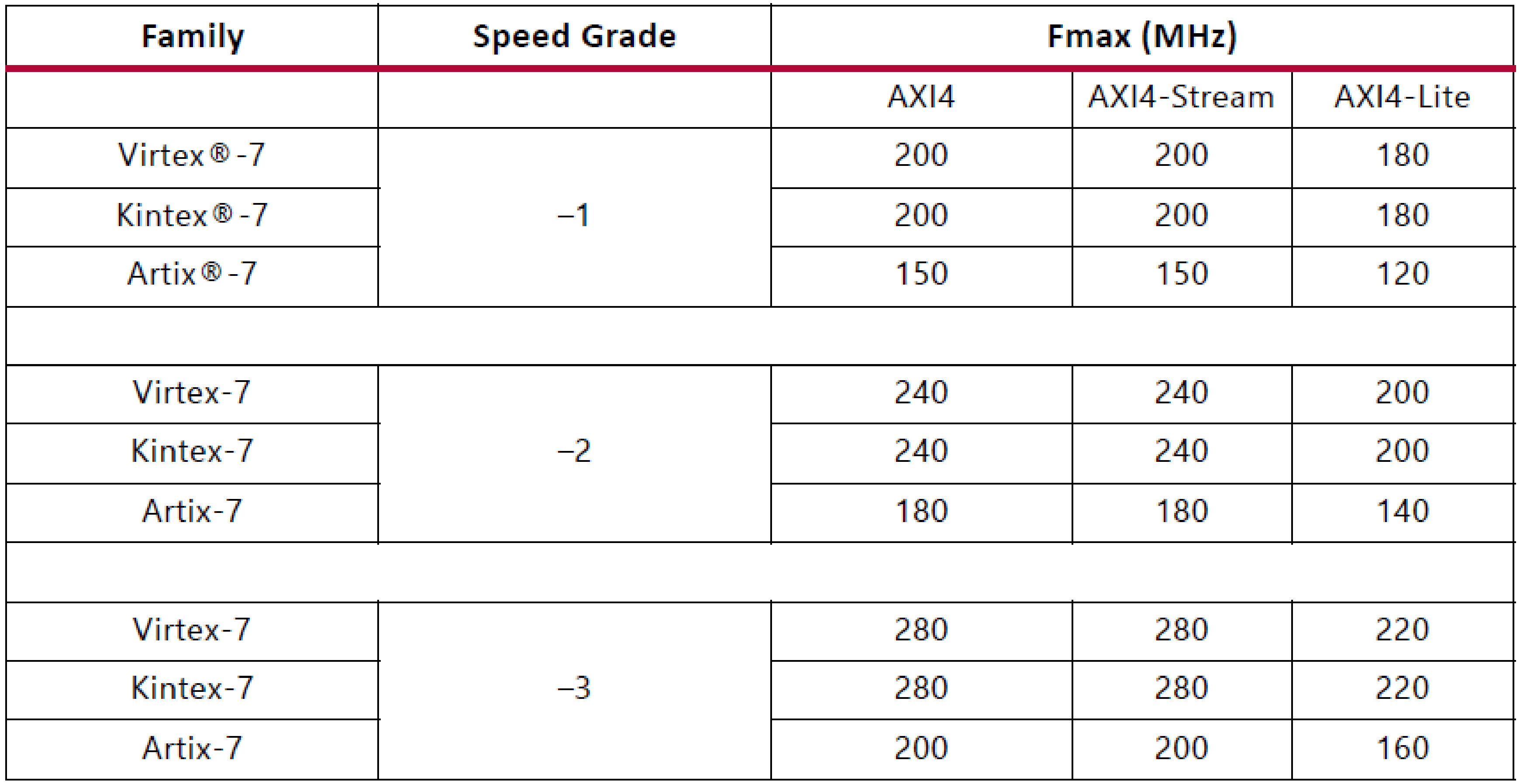


VDMA

- After registers are programmed through the AXI4-Lite interface, the Control/ Status logic block generates appropriate commands to the DataMover to initiate Write and Read commands on the AXI4 Master interface.
- A configurable asynchronous line buffer is used to temporarily hold the pixel data prior to writing it out to the AXI4-Memory Map interface or the AXI4-Stream interface.
- In the Write path, the AXI VDMA accepts frames on the AXI4-Stream Slave interface and writes it to system memory using the AXI4 Master interface.
- In the Read path, the AXI VDMA uses the AXI4 Master interface for reading frames from system memory and outputs it on the AXI4-Stream Master interface.

VDMA

Table 2-1: **Maximum Frequencies**



Family	**Speed Grade**	**Fmax (MHz)**		
		AXI4	AXI4-Stream	AXI4-Lite
Virtex®-7	–1	200	200	180
Kintex®-7		200	200	180
Artix®-7		150	150	120
Virtex-7	–2	240	240	200
Kintex-7		240	240	200
Artix-7		180	180	140
Virtex-7	–3	280	280	220
Kintex-7		280	280	220
Artix-7		200	200	160

VDMA

Tthe AXI VDMA throughput measured for different data widths. It was measured using standard High Definition (HD) frames on hardware.

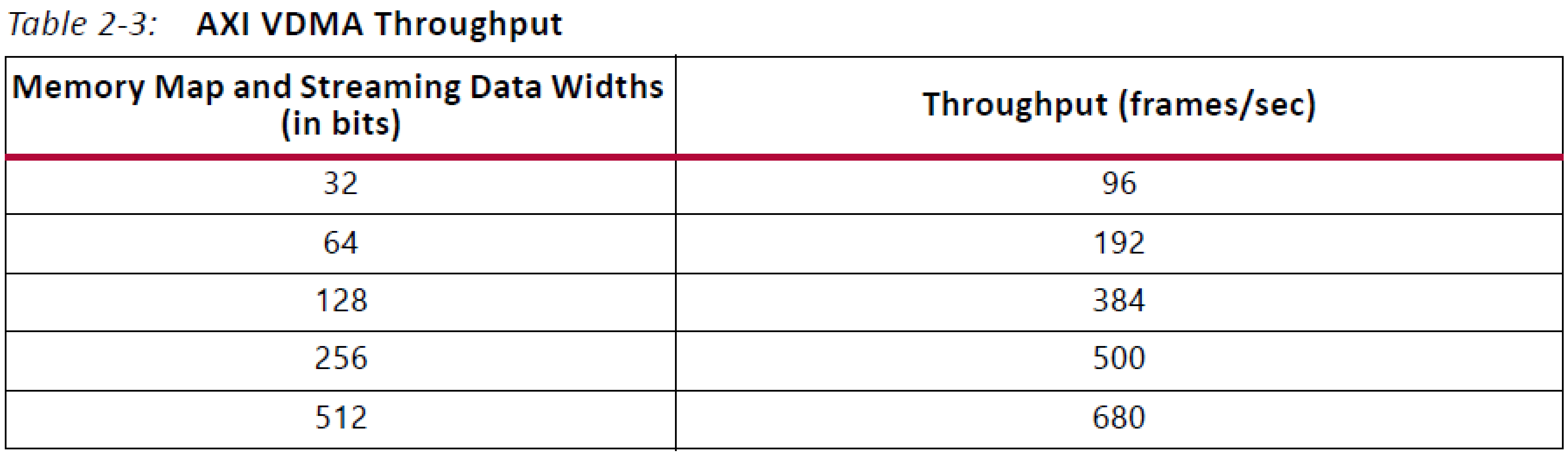


Table 2-3: **AXI VDMA Throughput**

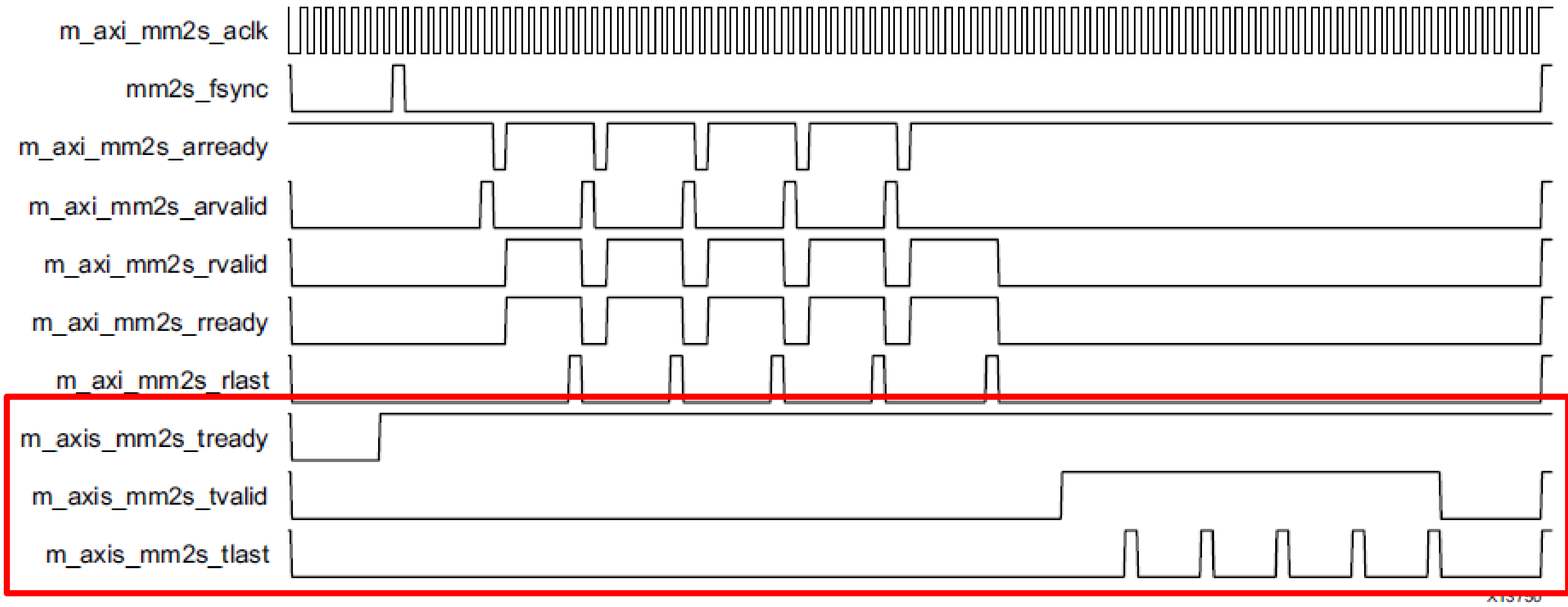
Memory Map and Streaming Data Widths (in bits)	Throughput (frames/sec)
32	96
64	192
128	384
256	500
512	680

VDMA

- Two separate paths:
  - Read (MM2S) Path
  - Write (S2MM) Path

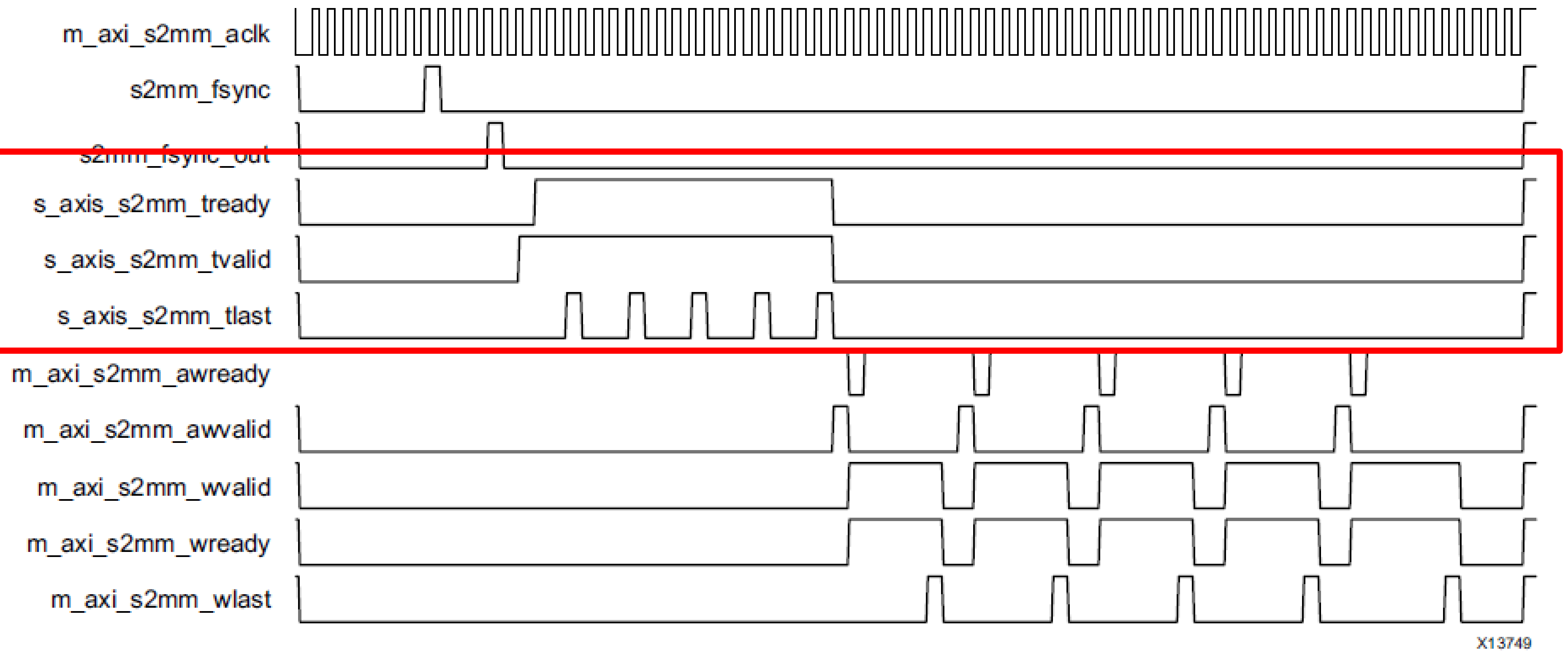
VDMA

- Read (MM2S) Path Timing

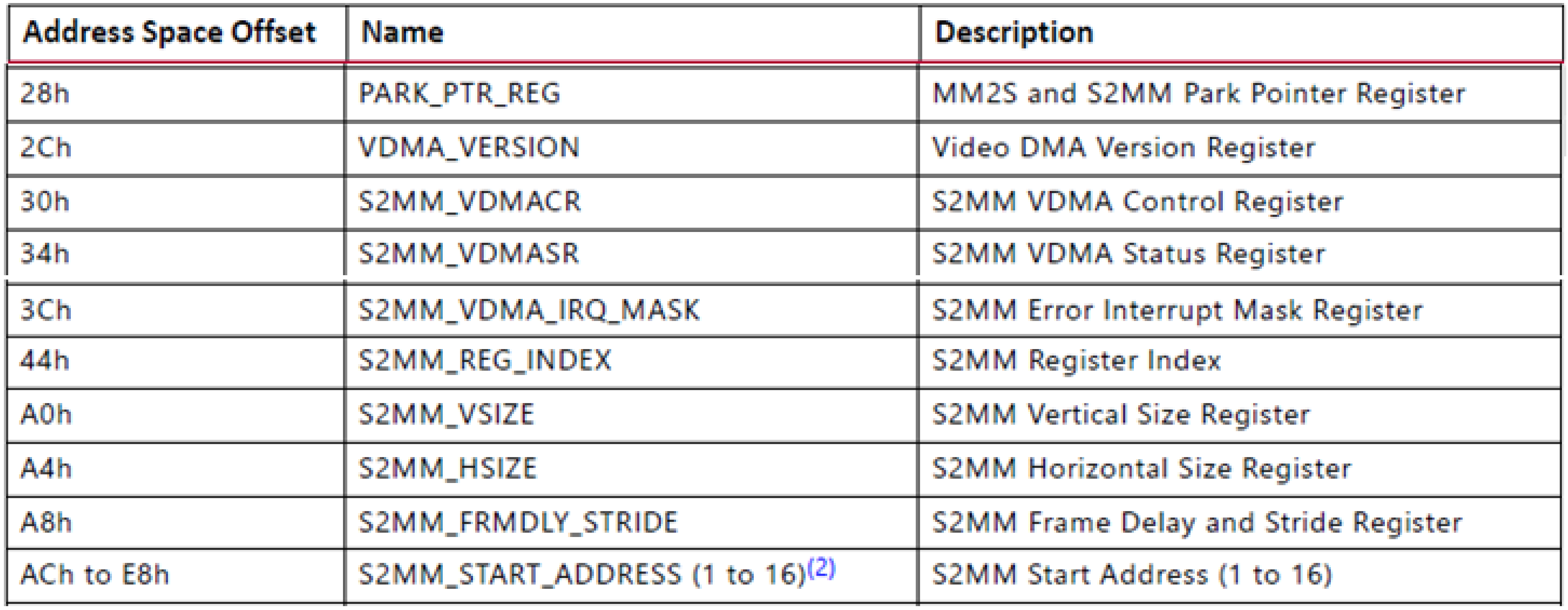


VDMA

- Write (S2MM) Path Timing



AXI VDMA Register Address Map



Address Space Offset	Name	Description
28h	PARK_PTR_REG	MM2S and S2MM Park Pointer Register
2Ch	VDMA_VERSION	Video DMA Version Register
30h	S2MM_VDMACR	S2MM VDMA Control Register
34h	S2MM_VDMASR	S2MM VDMA Status Register
3Ch	S2MM_VDMA_IRQ_MASK	S2MM Error Interrupt Mask Register
44h	S2MM_REG_INDEX	S2MM Register Index
A0h	S2MM_VSIZE	S2MM Vertical Size Register
A4h	S2MM_HSIZE	S2MM Horizontal Size Register
A8h	S2MM_FRMDLY_STRIDE	S2MM Frame Delay and Stride Register
ACh to E8h	S2MM_START_ADDRESS (1 to 16)[2]	S2MM Start Address (1 to 16)

VDMA Inicijalizacija

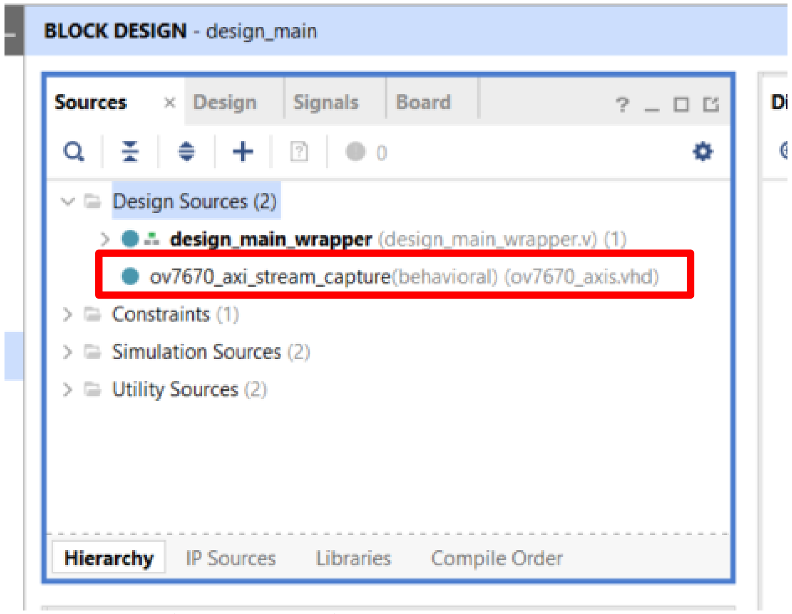
- 1) Resetirati VDMA
  - U registru `S2MM_CONTROL_REGISTER` postaviti reset bit
- 2) Provjeri jeli se VDMA resetirao
  - Čitaj reset bit u `S2MM_CONTROL_REGISTER` i provjeravaj jel se VDMA pokrenuo
- 3) Do not mask interrupts
  - U S2MM_IRQ_MASK upiši 0xf
- 4) Start up buffer number
  - U registar S2MM_CONTROL_REGISTER upišite:
    - Da imate tri buffera
    - Te da se koristi:
      - VDMA_CONTROL_REGISTER_START |
      - VDMA_CONTROL_REGISTER_GENLOCK_ENABLE |
      - VDMA_CONTROL_REGISTER_INTERNAL_GENLOCK |
      - VDMA_CONTROL_REGISTER_CIRCULAR_PARK

VDMA Inicijalizacija

- 5) Čekajte dok nije ispunjen uvijet:
  - Da je zaustavljen (0x30) ili VDMA kanal „Halted” (0x34)
- 6) Postavi registar S2MM_REG_INDEX u nula
  - Iako po specifikaciji to nema utjecaja
- 7) Postavi adrese FrameBuffera (S2MM Start Addresses)
  - VDMA_S2MM_FRAMEBUFFER1
  - VDMA_S2MM_FRAMEBUFFER2
  - VDMA_S2MM_FRAMEBUFFER3
- 8) Write Park pointer register
  - PARK_PTR_REG = 0
- 9) Frame delay and stride (bytes)
  - S2MM_FRMDLY_STRIDE = width * pixelLength
- 10) Write horizontal size (bytes)
  - S2MM_HSIZE, width * pixelLength
- 11) Write vertical size (lines), this actually starts the transfer
  - S2MM_VSIZE, height
- 12) Clear all error bits in status register

VDMA - Vivado

- 1) Dodajte VHDL cod za dohvat slike sa kamere
  - U tabu sources -> Design Sources – desni klik (add sources)
  - Kad ste dodali datoteku trebao bi vam se pojaviti:



BLOCK DESIGN - design_main

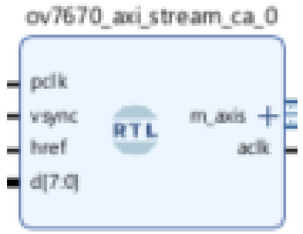
Sources | Design | Signals | Board

Design Sources (2)
- design_main_wrapper (design_main_wrapper.v) (1)
- ov7670_axi_stream_capture(behavioral) (ov7670_axis.vhd)

Constraints (1)
Simulation Sources (2)
Utility Sources (2)

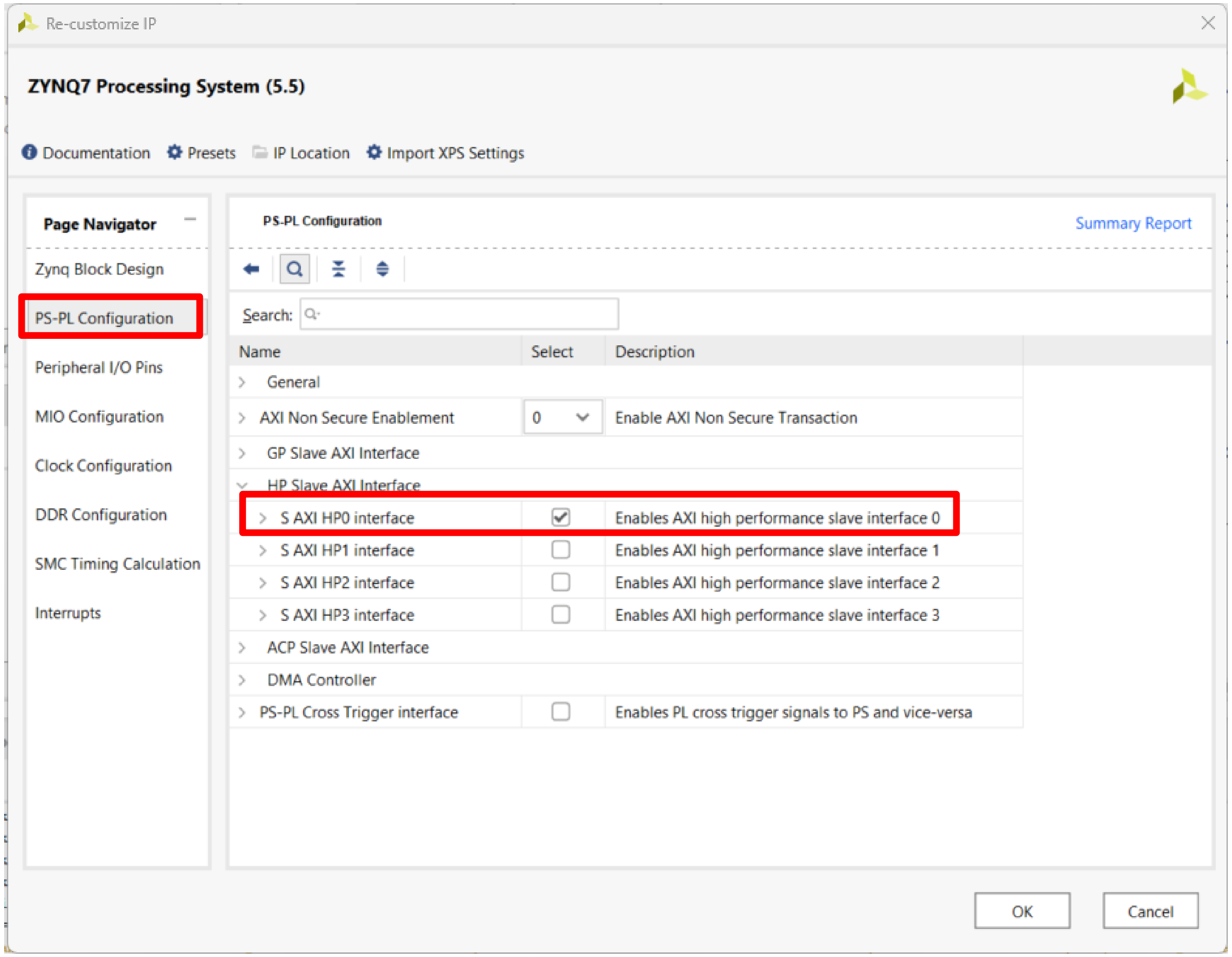
Hierarchy | IP Sources | Libraries | Compile Order

  - 2) Prevucite datoteku u Diagram prozor i dobit će te novu komponentu u diagramu:



VDMA - Vivado

- 3) Podesite porcesor:
  - Potrebno je uključiti HP0 na koji spajamo VDMA



Re-customize IP

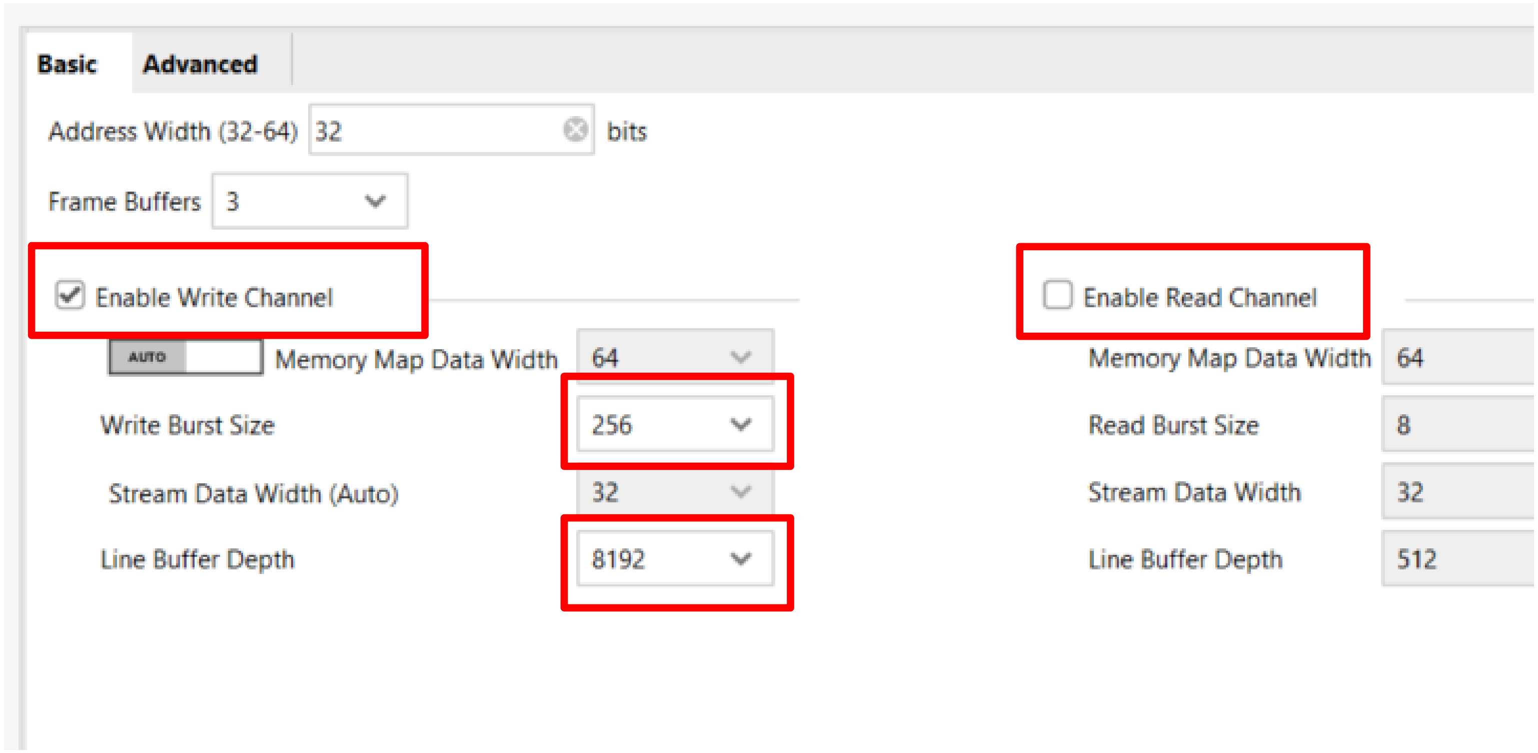
ZYNQ7 Processing System (5.5)

PS-PL Configuration

Name	Select	Description
General		
AXI Non Secure Enablement	0	Enable AXI Non Secure Transaction
GP Slave AXI Interface		
HP Slave AXI Interface		
S AXI HP0 interface	✓	Enables AXI high performance slave interface 0
S AXI HP1 interface	☐	Enables AXI high performance slave interface 1
S AXI HP2 interface	☐	Enables AXI high performance slave interface 2
S AXI HP3 interface	☐	Enables AXI high performance slave interface 3
ACP Slave AXI Interface		
DMA Controller		
PS-PL Cross Trigger interface	☐	Enables PL cross trigger signals to PS and vice-versa

VDMA - Vivado

- 4) Dodajte AXI Video Direct Memory Access komponentu i izvršite konfiguraciju:



Address Width (32-64) 32 bits

Frame Buffers 3

[✓] Enable Write Channel

AUTO Memory Map Data Width 64

Write Burst Size 256

Stream Data Width (Auto) 32

Line Buffer Depth 8192

[ ] Enable Read Channel

Memory Map Data Width 64

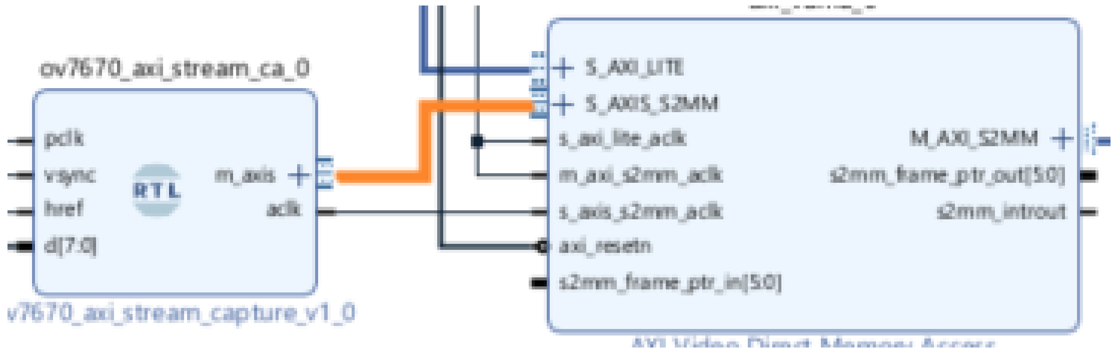
Read Burst Size 8

Stream Data Width 32

Line Buffer Depth 512

VDMA - Vivado

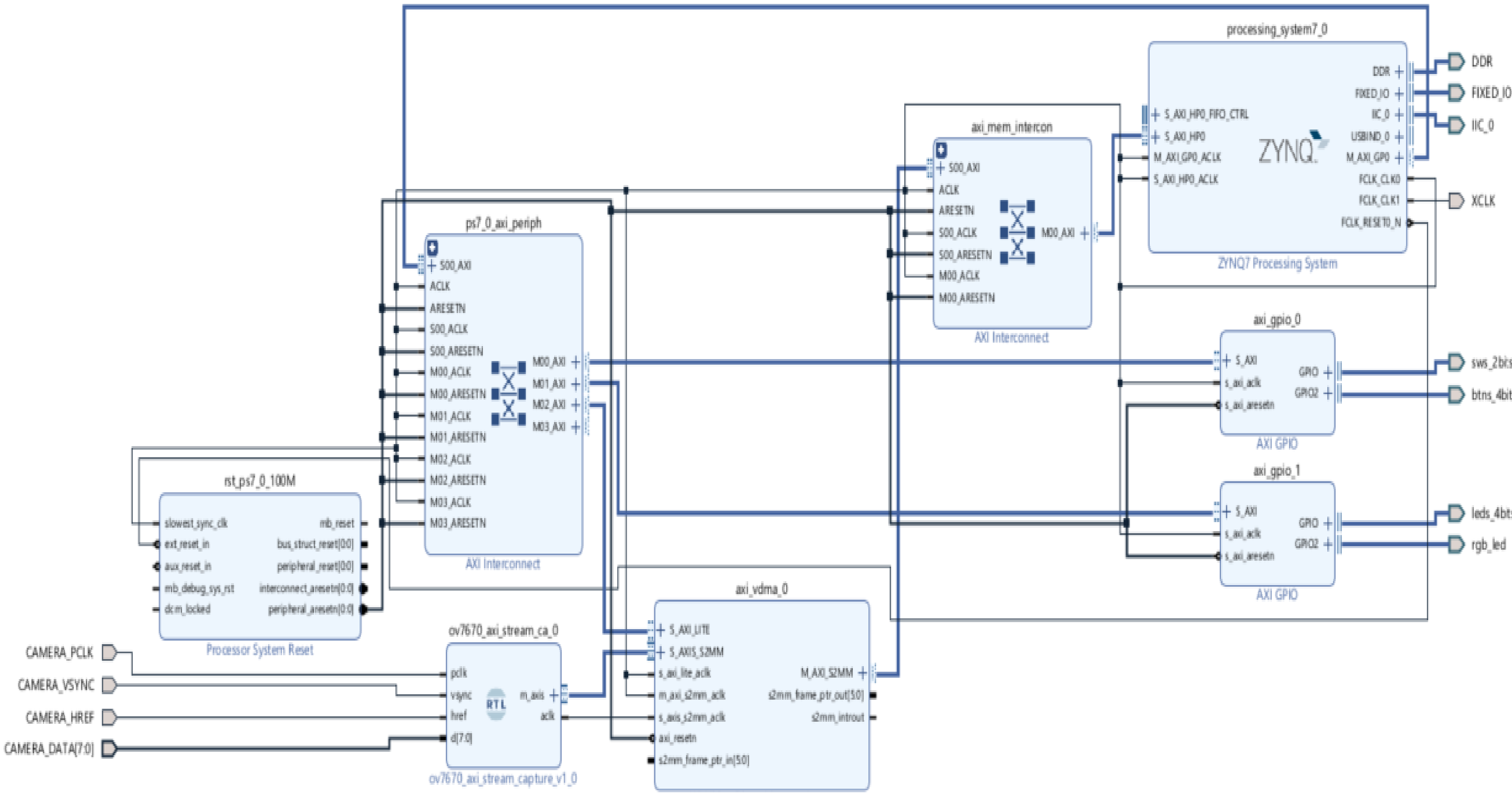
- 5) Pokrenite auto konekciju koju nude alati
- 6) Spojite komponentu za dohvat slike sa VDMA sklopom



- 7) Spojite i ackl ili opet pokrenite automatsko povezivanje
- 8) Postavite preostale priključke (pclk, vsync, href, d[7:0]) na komponenti za dohvat slike u external („Make External”)
- 9) Podesite sve postavke u constrainst datoteci (zadnji put smo dodali samo potrebne za I2C i XCLK, sad moramo dodati novo dodate external priključke)

VDMA - Vivado

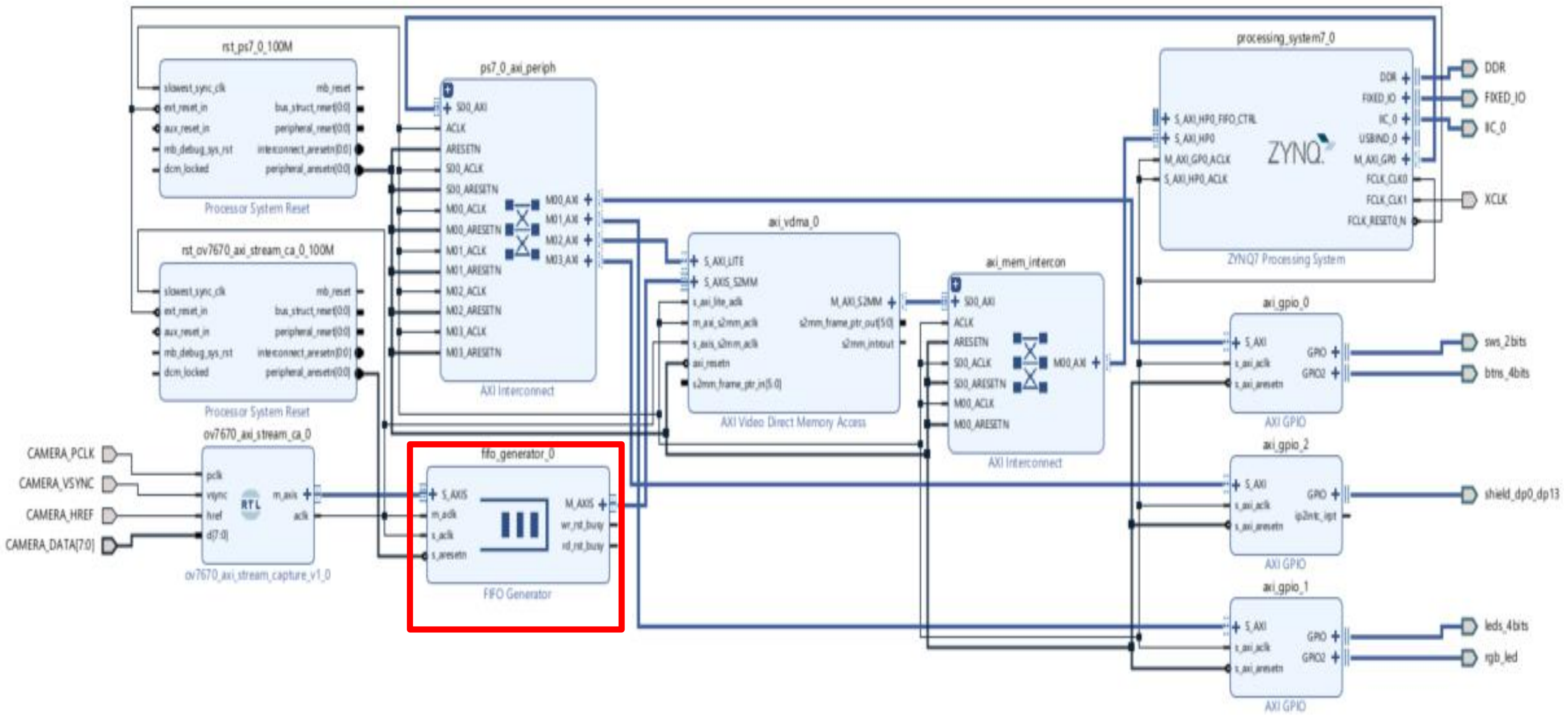
- U konačnici trebate dobiti



VDMA - Vivado

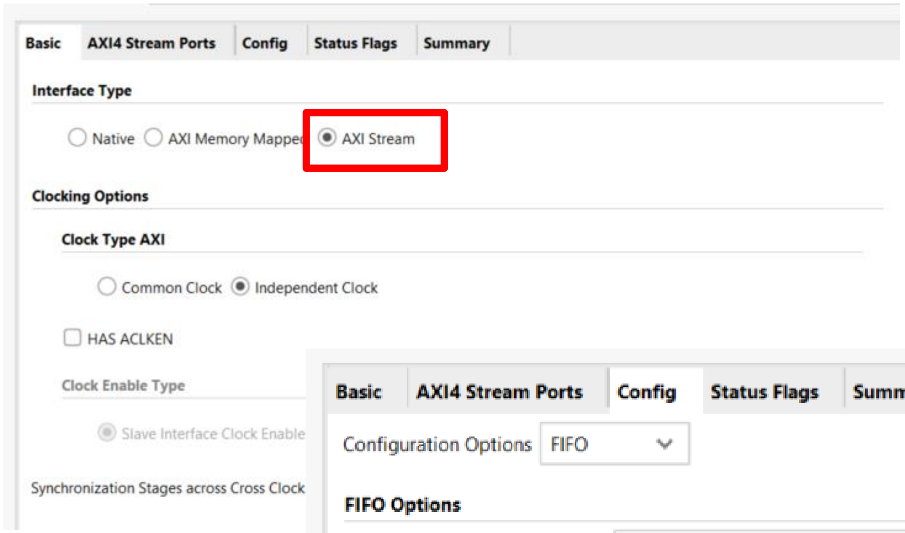
- 10) Pokrenite kreiranje Bitstream
  - Generate BitStreame
- 11) File -> Export -> Export Hardware …
- I to bi trebalo biti to

Neobavezno dodavaje FIFO Generator komponente



FIFO

- Postavke FIFO Generator komponente



Basic | AXI4 Stream Ports | Config | Status Flags | Summary

Interface Type

Native [ ] AXI Memory Mapped [ ] [✓ AXI Stream]

Clocking Options

Clock Type AXI

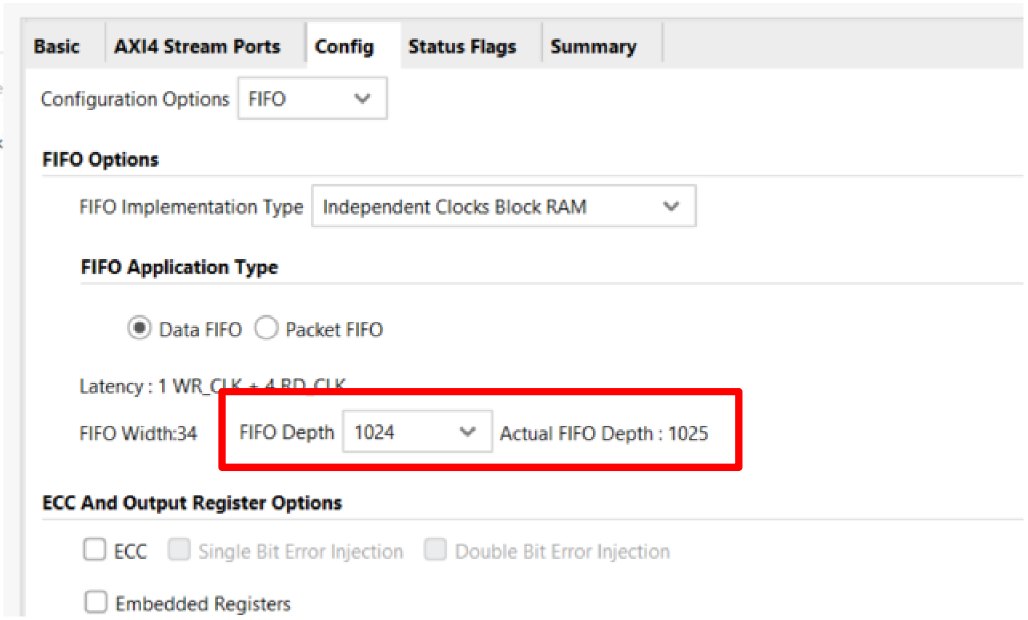
Common Clock [ ] [✓ Independent Clock]

[ ] HAS ACLKEN

Clock Enable Type

[✓ Slave Interface Clock Enable]

Synchronization Stages across Cross Clock



Basic | AXI4 Stream Ports | Config | Status Flags | Summary

Configuration Options: FIFO

FIFO Options

FIFO Implementation Type: Independent Clocks Block RAM

FIFO Application Type

[✓ Data FIFO] Packet FIFO [ ]

Latency : 1 WR_CLK + 4 RD_CLK

FIFO Width:34 FIFO Depth: 1024 Actual FIFO Depth : 1025

ECC And Output Register Options

[ ] ECC [ ] Single Bit Error Injection [ ] Double Bit Error Injection

[ ] Embedded Registers