Bridge of Life Education

Advanced SOC Design

2024 Spring Course Plan

Jiin Lai

Agenda

- Objectives
- Recap SOC Design Course and its continuation
- Lab Platform
- Course Logistics
- Course Materials
- Course Schedule
- Course Policy & Grading

Objectives

Educate a full-stack IC designer with

- IC design
- FPGA design
- Embedded Programming

By Hands-on Labs on SOC-level Platform

Complete SOC Design Training includes

- **One year SOC Design course**
- **One Semester special project to tapeout the application accelerator chip (selected) & Chip Validation**

Recap on SOC Design

SOC Design (1st semester)

Objective:

- System (FPGA) & Embedded Programming
- Learn Verilog and HLS Design Implementation
- Study the SOC design and Front-end design flow
- Build SOC Simulation and Emulate it in FPGA
- Implement an Workload Optimized Design and integrate into Caravel SOC

Design & Lecture

1. Introduction to HLS and tools
2. Verilog & Logic Design
3. Caravel SOC
4. Processor
5. Memory
6. Peripheral
7. Interconnect - Bus
8. SOC Bus/Interconnect
9. Embedded Programming
10. Static Timing Analysis
11. Synthesis & Optimization
12. Verification & Simulation

Design Flow

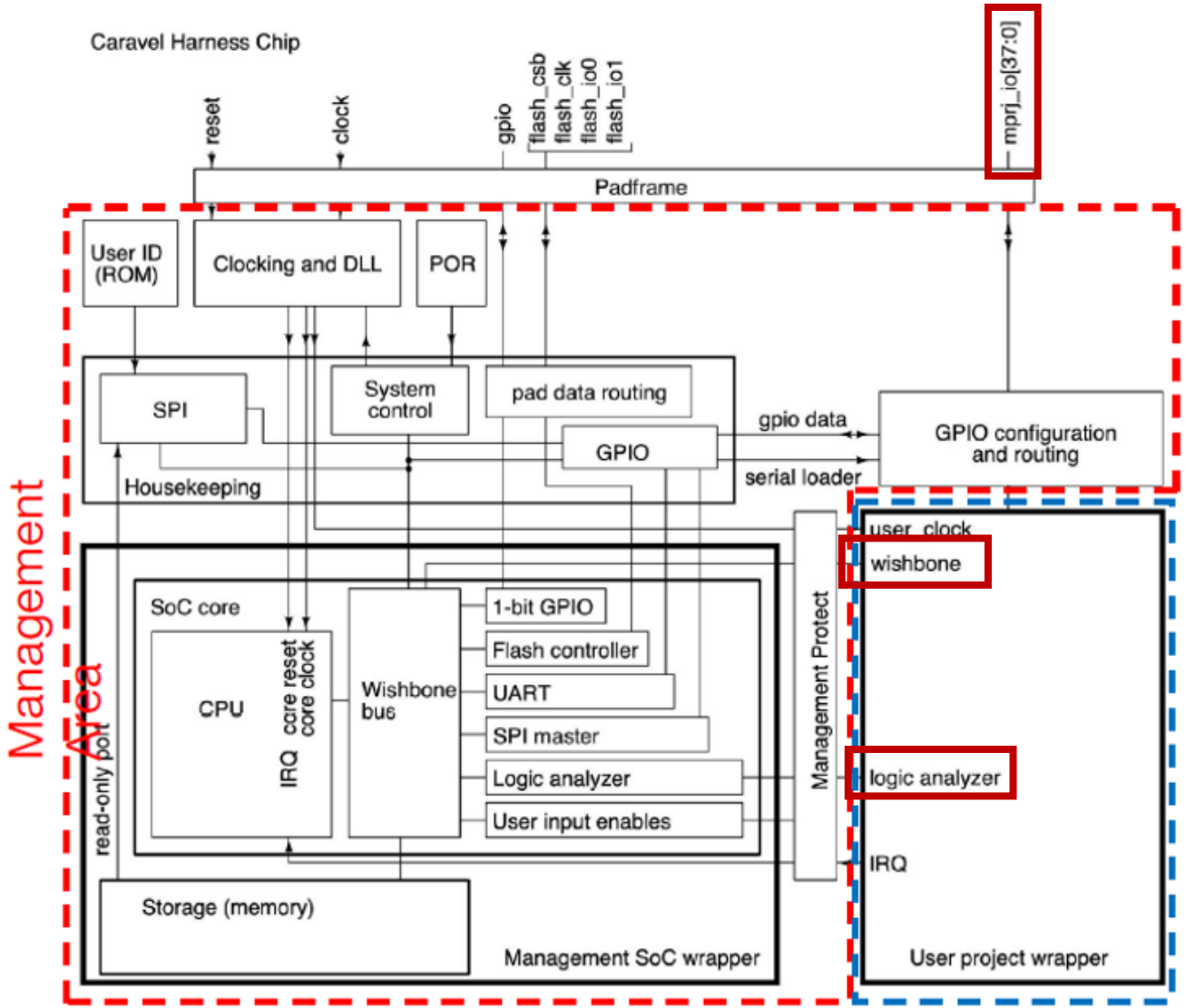
1.FPGA flow – Xilinx Vivado
2.Simulator（XSIM）
3.Synthesis (FPGA Synthesis)
4.Timing analysis
5.Verification

Laboratory

1.Vivado Tool Installation
2.Vitis HLS
3.Verilog FIR Filter (AXI master, AXI Stream)
4.Caravel SOC Simulation
5.Caravel SOC FPGA
6.SOC Design Labs
    1.Interrupt
    2.User RAM
    3.UART
    4.SDRAM
7.Workload Optimized SOC (WLOS) Baseline
8.Final Project

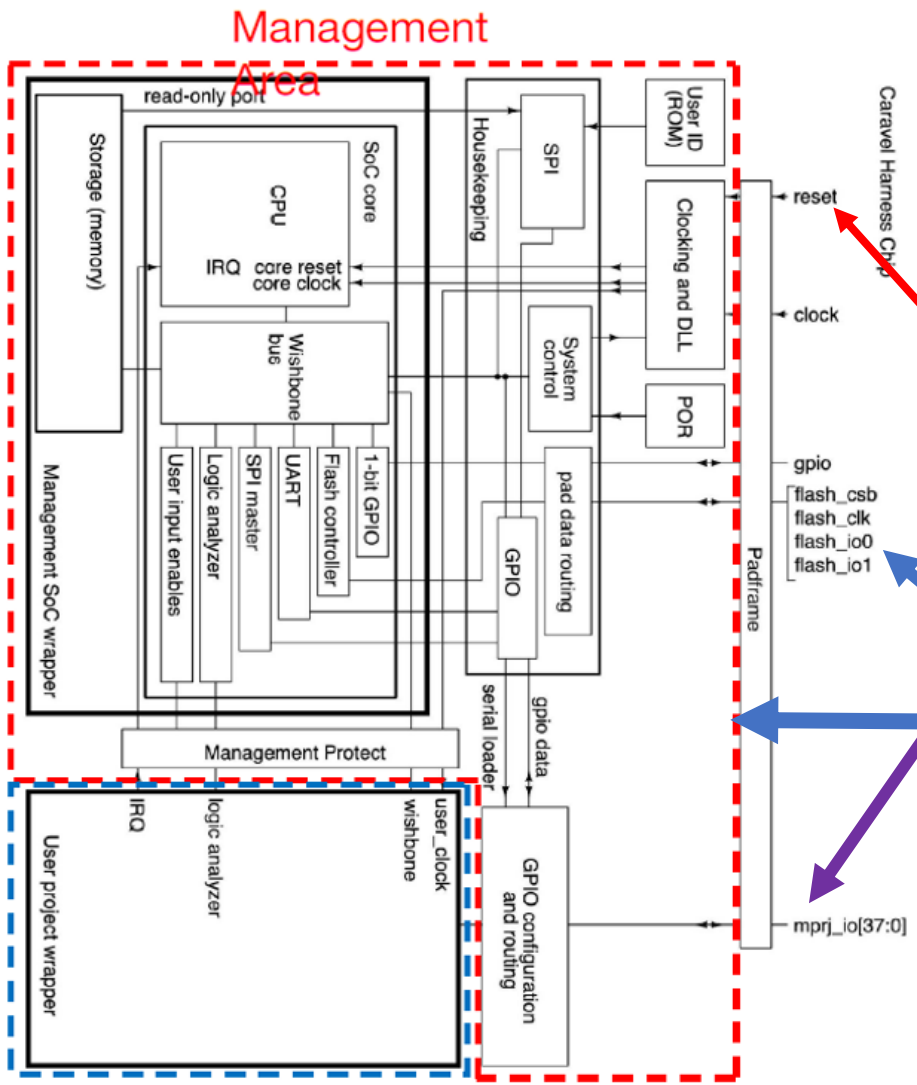
Lab Platform – CaravelFPGA©

Caravel Harness



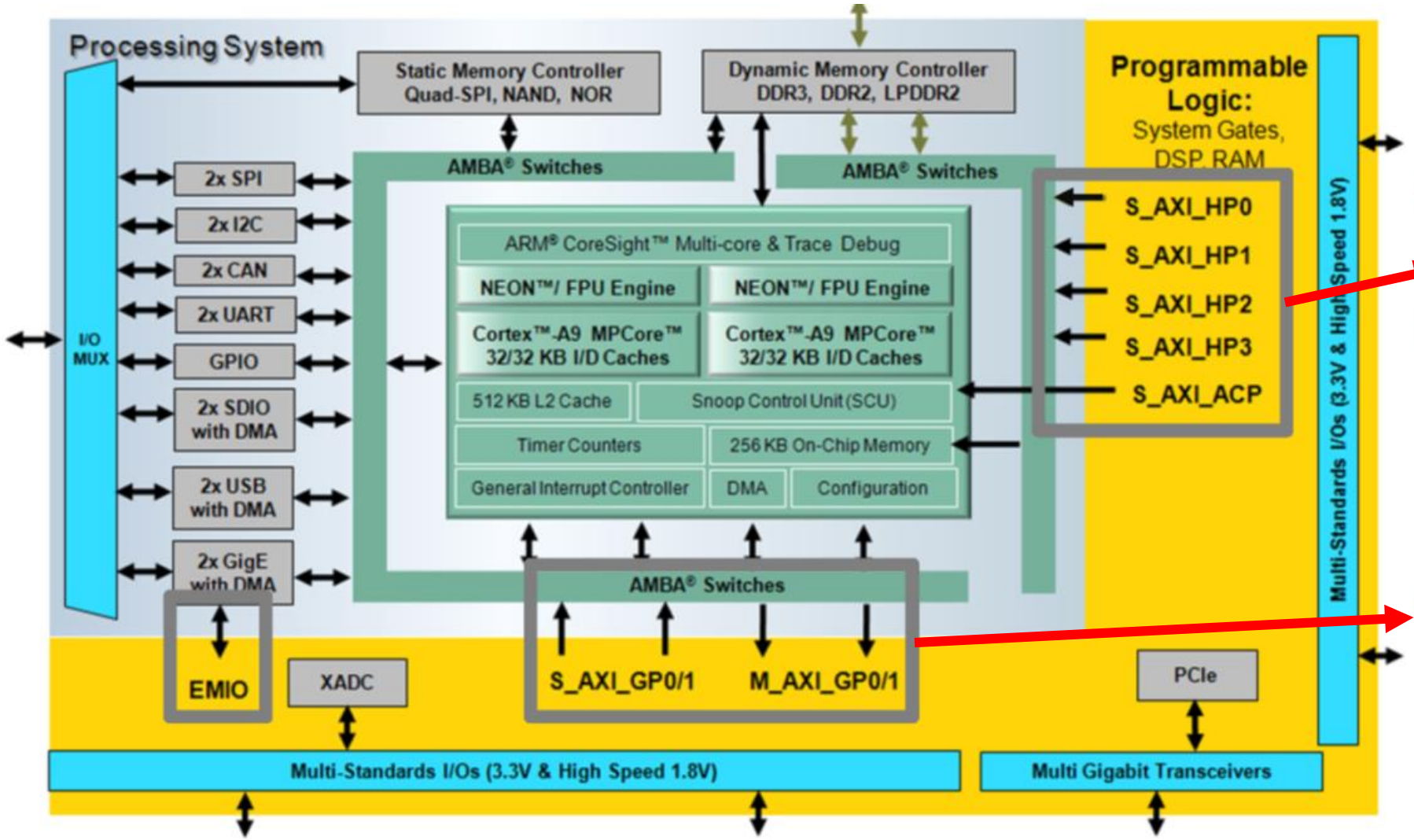
Caravel Simulation Verification System – Lab4

Verilog Testbench




```
module counter_tb;
Initial beigin
 wait(checkbits == 16'hAB40);
 wait(checkbits = ..
end

Initial begin
  RSB <= 1'b0;
  RSB <= 1'b1;
end

caravel uut(    );

Spiflash #( .FILENAME("counter_la.hex"} spiflash ( );

tbuart tburat (     );

endmodule
```

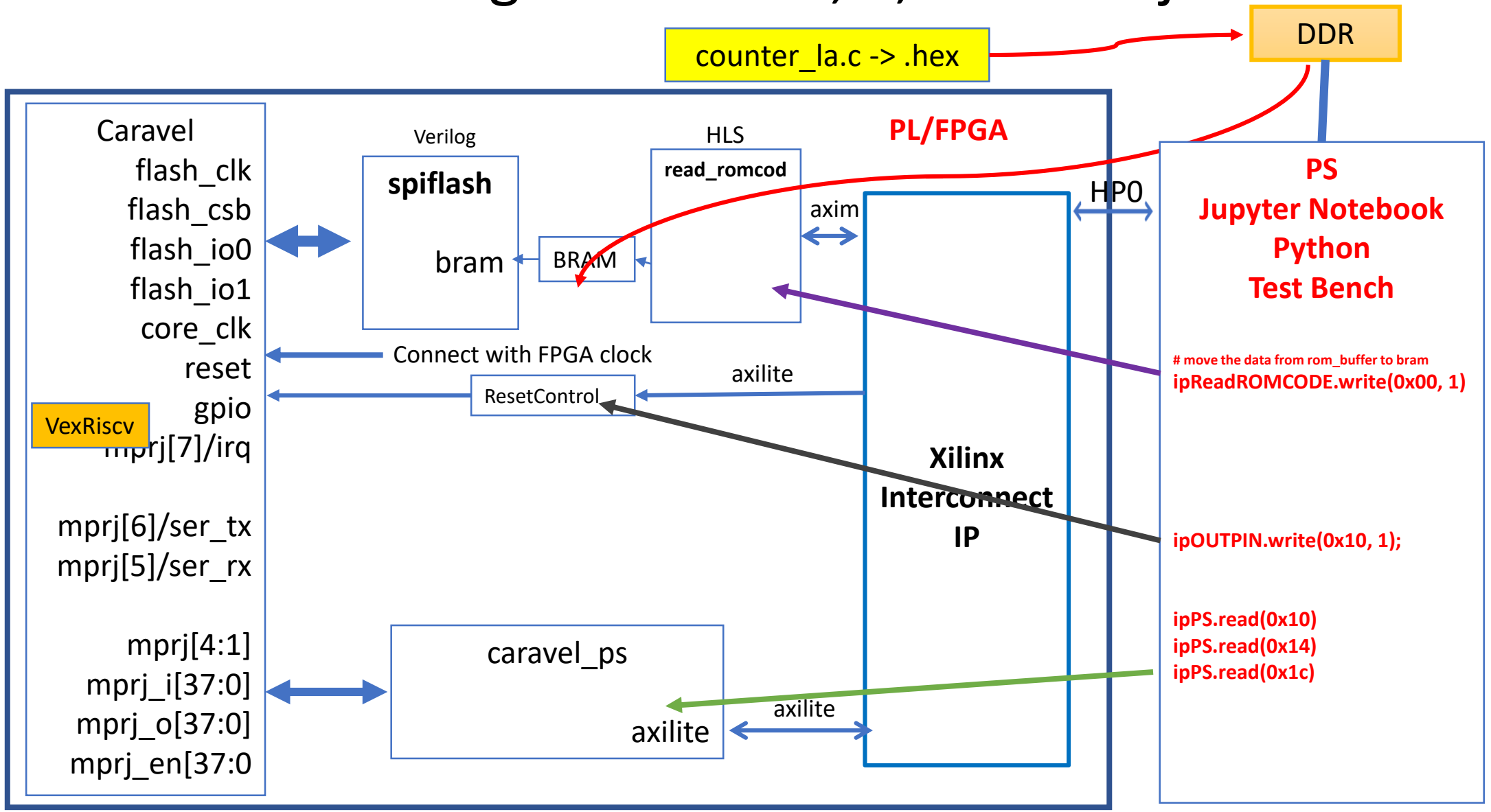
Zynq Block Diagram (PYNQ-Z2 or KV260)



- **AXI High Performance**
  - 4x Slave (64-bit)
    - 1K FIFOs
- **ACP**
  - 1x Slave (64-bit)
  - Cache access

- **AXI General Purpose ports**
  - 2x Master (32-bit)
  - 2x Slave (32-bit)

CaravelFPGA Block Diagram – Lab 5, 6, Final Project



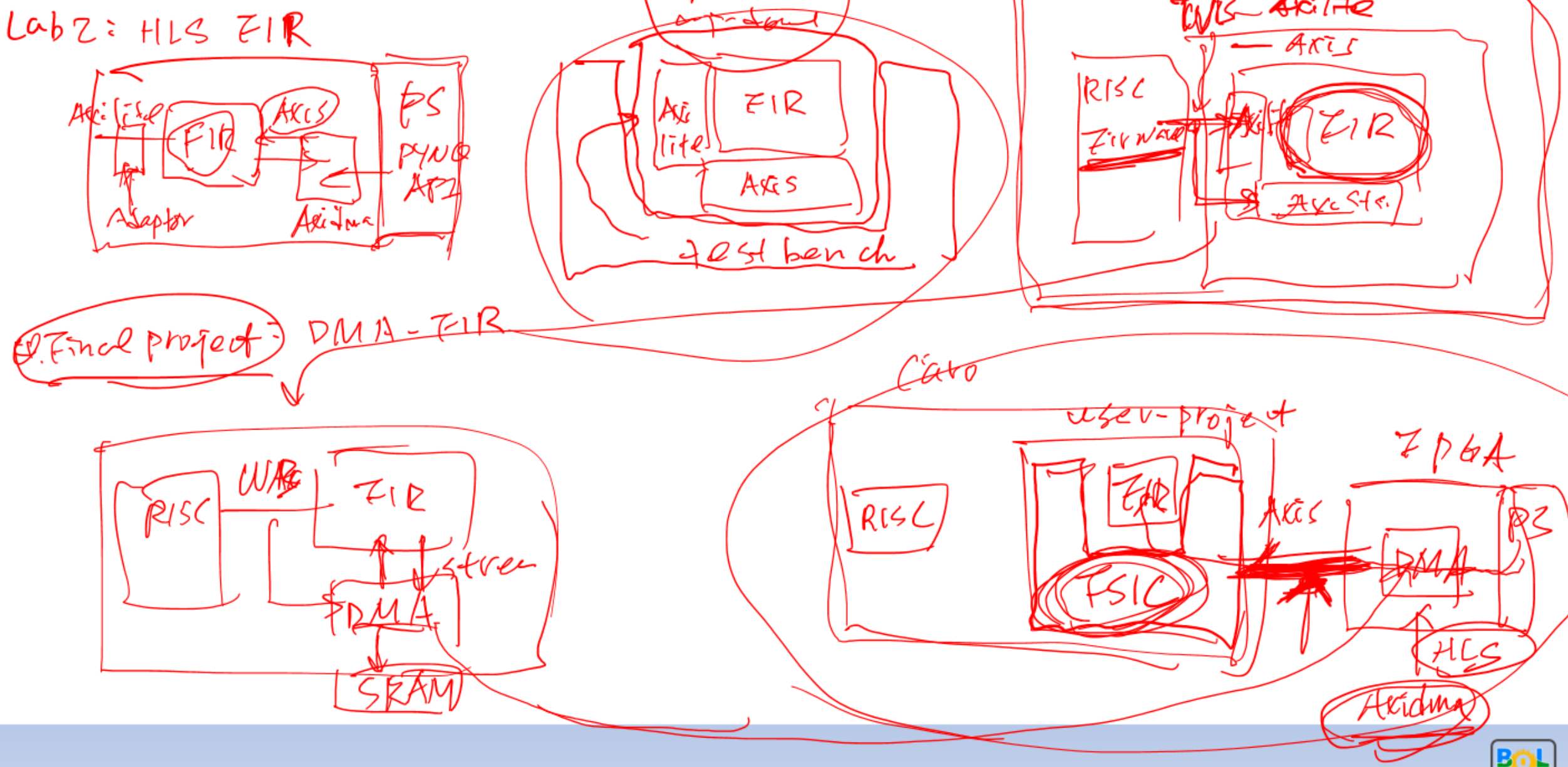
Lab-FIR Evolution

Objective

We use an FIR design (11 taps - $y[t] = \Sigma (h[i] * x[t - i])$ through a series of laboratories to exercise the flow of building
an accelerator and integrate it into a SOC system.

The purpose is to investigate the
system operation and improve its performance through hardware/ software co-design methodology.

Lab FIR Evolution

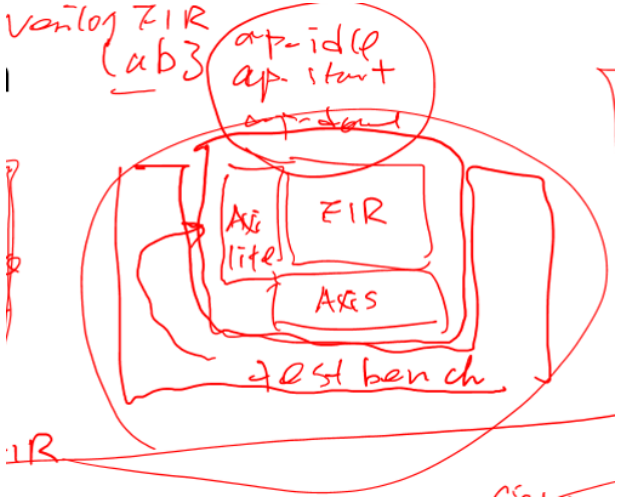


Lab#2 – HLS-FIR

- Purpose - Embedded System Concept – HW / SW Interface
- Implement FIR with HLS
- HLS compiler generates Axilite and Axi-stream Interface
- Host Python accesses FIR through PYNQ API

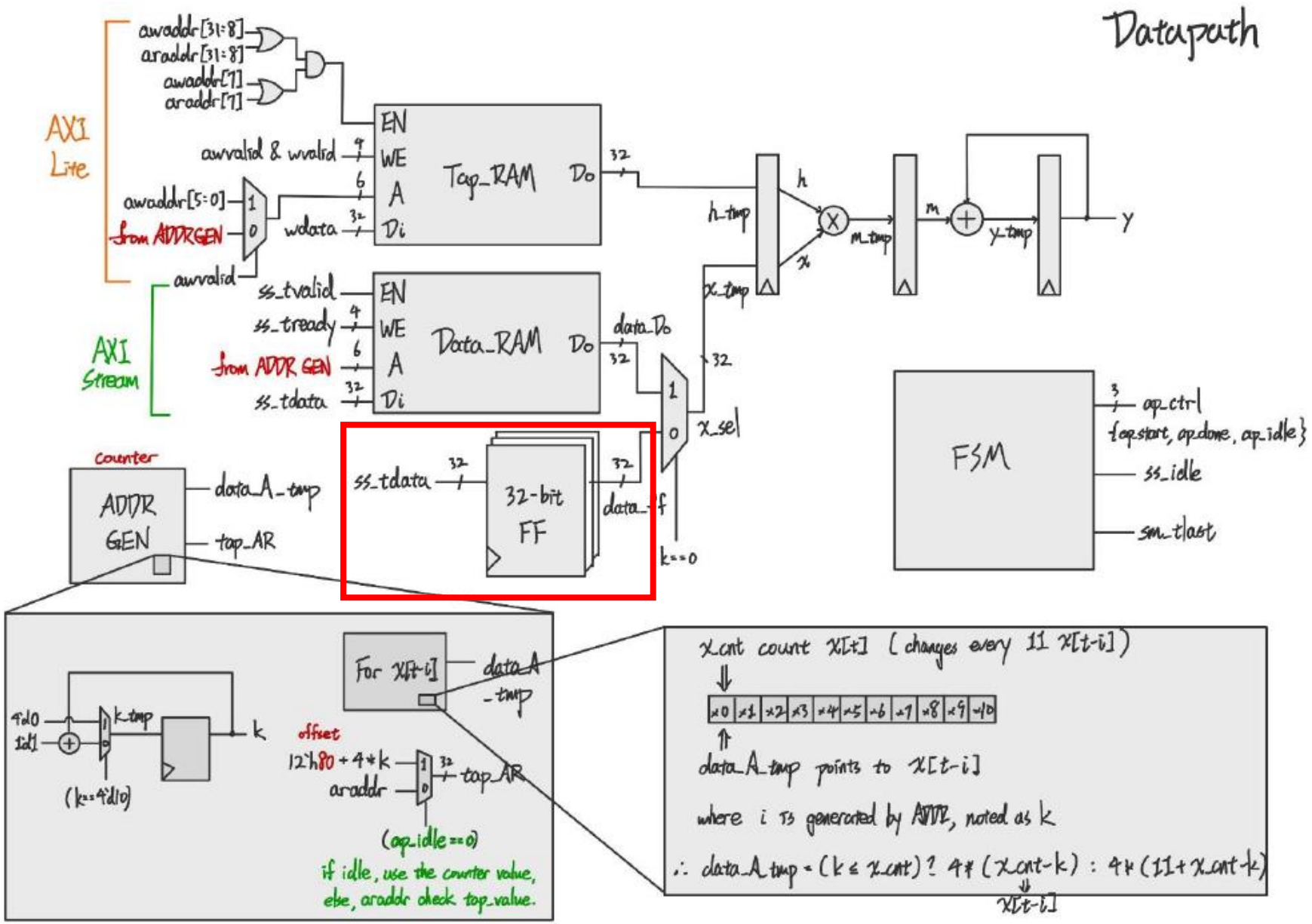
- You don’t actually design the Axilite/Axi-stream interface
- You don’t really know hardware/software communication under the hood. (PYNQ-API)
- You don’t control the data transfer

Lab3 – Verilog-FIR



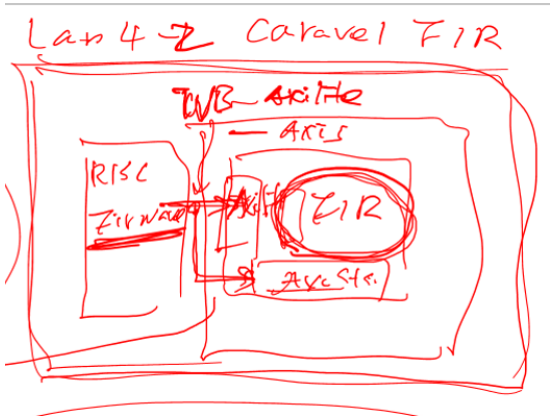
- Purpose: Verilog design & AXI Bus interface
- Design FIR with verilog
  - Design the Axilite, and Axi-stream bus interface
  - Design the data and tap storage with external synchronous SRAM (1T latency)
  - Ring FIFO for data/tap storage, and pipeline design techniques
  - Use single MAC – resource sharing and scheduling
  - Define the host communication protocol. Understand the issues for hardware and software co-design under a system operation perspective.

- Design testbench
  - Advanced testbench techniques, e.g. using fork-join, and disable to emulate parallel bus operation. Note: 3 buses runs in parallel, axilite for configuration, stream-in, and stream-out.
  - Design testbench based on system specification instead of how you implement it.
  - Validate the Block-level protocol for possible race condition.



https://drive.google.com/file/d/1ub6Xp_oDGn7PJQc_kIeb4n0XZjndhOh1/view?usp=drive_link

Lab 4-2 – Caravel-FIR

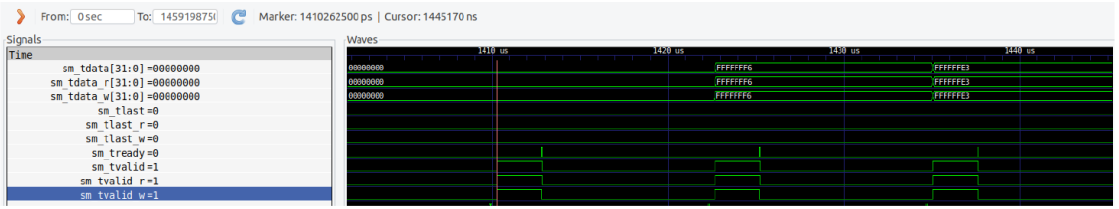


- Purpose: HW-SW Codesign
- Take Lab 3-Verilog FIR and integrate into Caravel SOC user project
- Design Wishbone to Axilite, Axis interface conversion
- Replace Testbench with RISC-V firmware code
  - Define and implement a robust block-level protocol to improvement performance. Note: both hardware and software are in your hand.
  - Implement a buffering/control scheme to improve system throughput based on the observation on RISC-V process speed, You have the choice of bram11.v or bram12.v. Note: the one extra data buffer can be very helpful to improve system performance.
- Metrics to evaluate your system: #-of-clock (latency-timer) * clock_period * gate-resource
  - #-of-clock - the latency-timer in testbench
  - clock_period – after synthesis, the longest path in static timing report
  - gate-resource - # of (LUT + FF) – assuming there is no distributed RAM, nor Block RAM in RTL use.
  - The lower the better

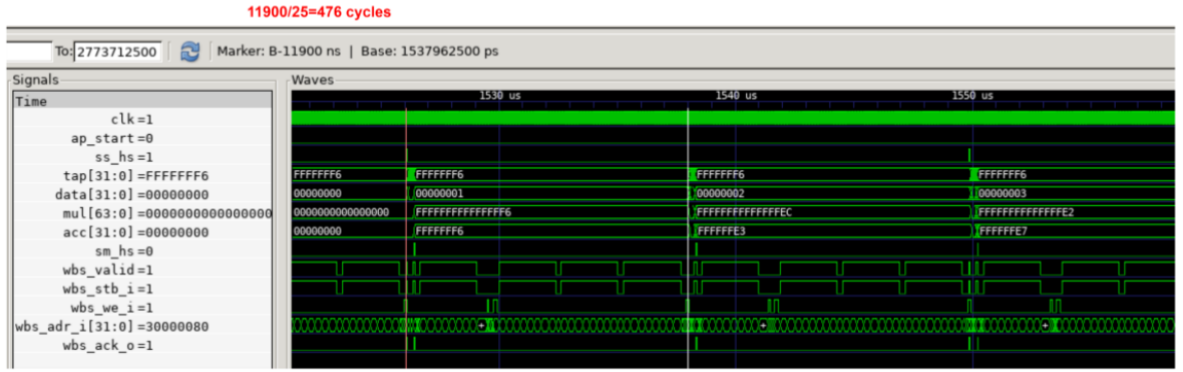
Generally, CPU feed data to FIR at a rate 400+ cycles

What is latency for firmware to feed data?

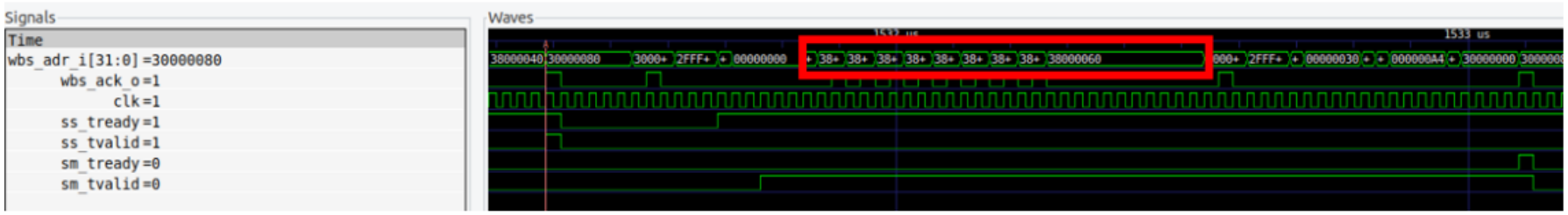
```
Latency: (1422637500 - 1410262500)ps / (25000)ps = 495 cycle
```

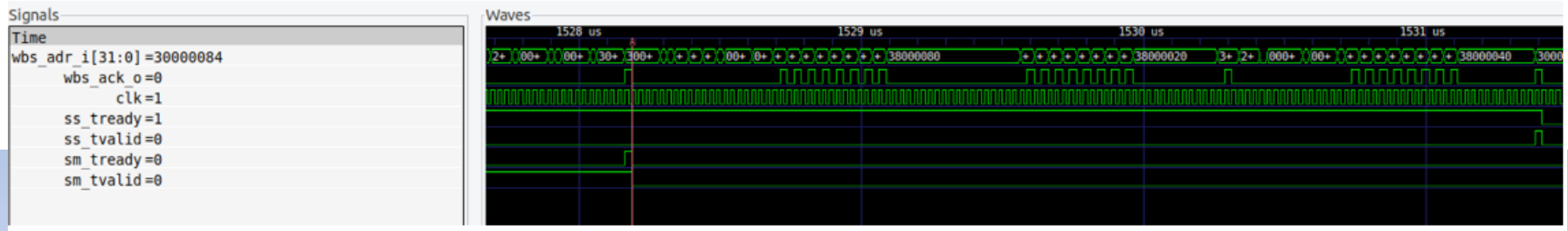


實際上的 throughput 如下圖所示，為 476 cycles per output。



CPU does a lot of code fetch (address: 3800_xxxx)





Unfold Loop by 2 to reduce branch overhead

Reorder Code (3)

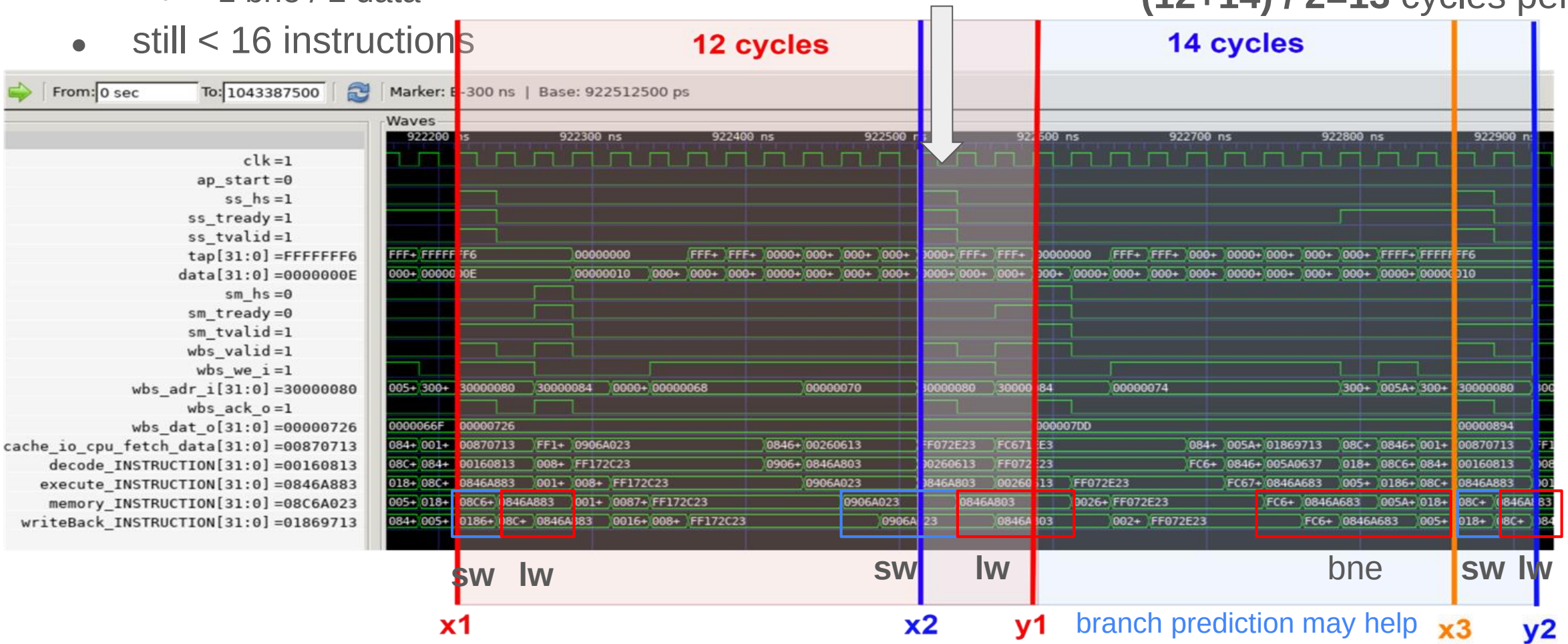
- 2-unfolded
  - 1 bne / 2 data
- still < 16 instructions

fir.c

```
*reg_fir_x = 0;
*reg_fir_x = 1;
outputsignal[0] = *reg_fir_y;
for(int i = 1; i < L / 2; i++)  {
    *reg_fir_x = 2*i;
    outputsignal[2*i-1] = *reg_fir_y;   // receive y[n] from FIR
    *reg_fir_x = 2*i+1;                 // send x[n] to FIR
    outputsignal[2*i] = *reg_fir_y; // receive y[n] from FIR
}
outputsignal[L - 1] = *reg_fir_y;
```

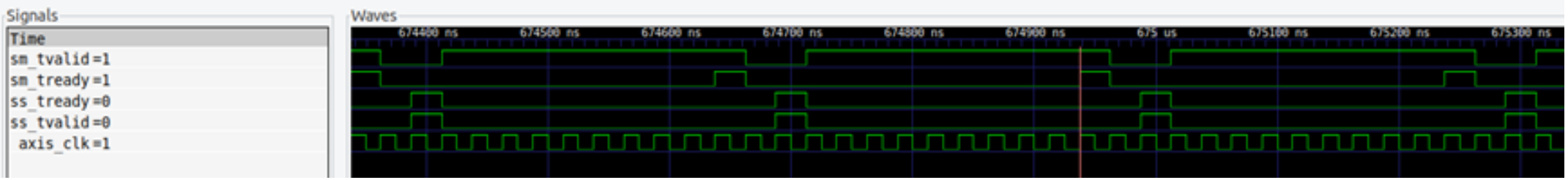
```
38000128: 08c6a023          sw  a2,128(a3) # 3
3800012c: 0846a883          lw  a7,132(a3)
38000130: 00160813          addi  a6,a2,1
38000134: 00870713          addi  a4,a4,8
38000138: ff172c23          sw  a7,-8(a4)
3800013c: 0906a023          sw  a6,128(a3)
38000140: 0846a803          lw  a6,132(a3)
38000144: 00260613          addi  a2,a2,2
38000148: ff072e23          sw  a6,-4(a4)
3800014c: fc671ee3          bne a4,t1,38000128
```

(12+14) / 2=13 cycles per data



更新一下:目前優化程度，因一開始硬體設計上並沒有加一層FF，所以每次做完y都要等到收到x才能繼續，這時極限只能做到2+14 cycles per data且在此狀況下還要花時間去設計究竟x->y還是y->x哪段要比較快來配合硬體。不過感謝台大同學的報告，從中發現加了一層FF確實可以再進一步優化。有了FF後，我們可以再運算的過程中也可以收x，所以我們現在只要思考如何讓smtready之間的距離縮短即可。所以我試著再去縮短smtready之間的cycle數，最後做出來**12cycles per data**。

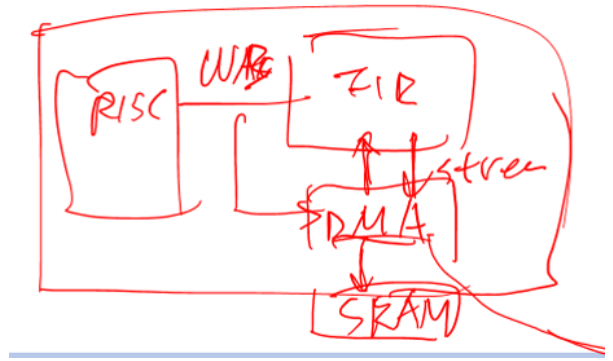
```
38000024:       0846a603                lw      a2,132(a3) # 30000084 <_erodata+0x1ffffe8c>
38000028:       00478793                addi    a5,a5,4
3800002c:       08e6a023                sw      a4,128(a3)
38000030:       fec7ae23                sw      a2,-4(a5)
38000034:       00170713                addi    a4,a4,1
38000038:       feb796e3                bne     a5,a1,38000024 <fir+0x24>
```



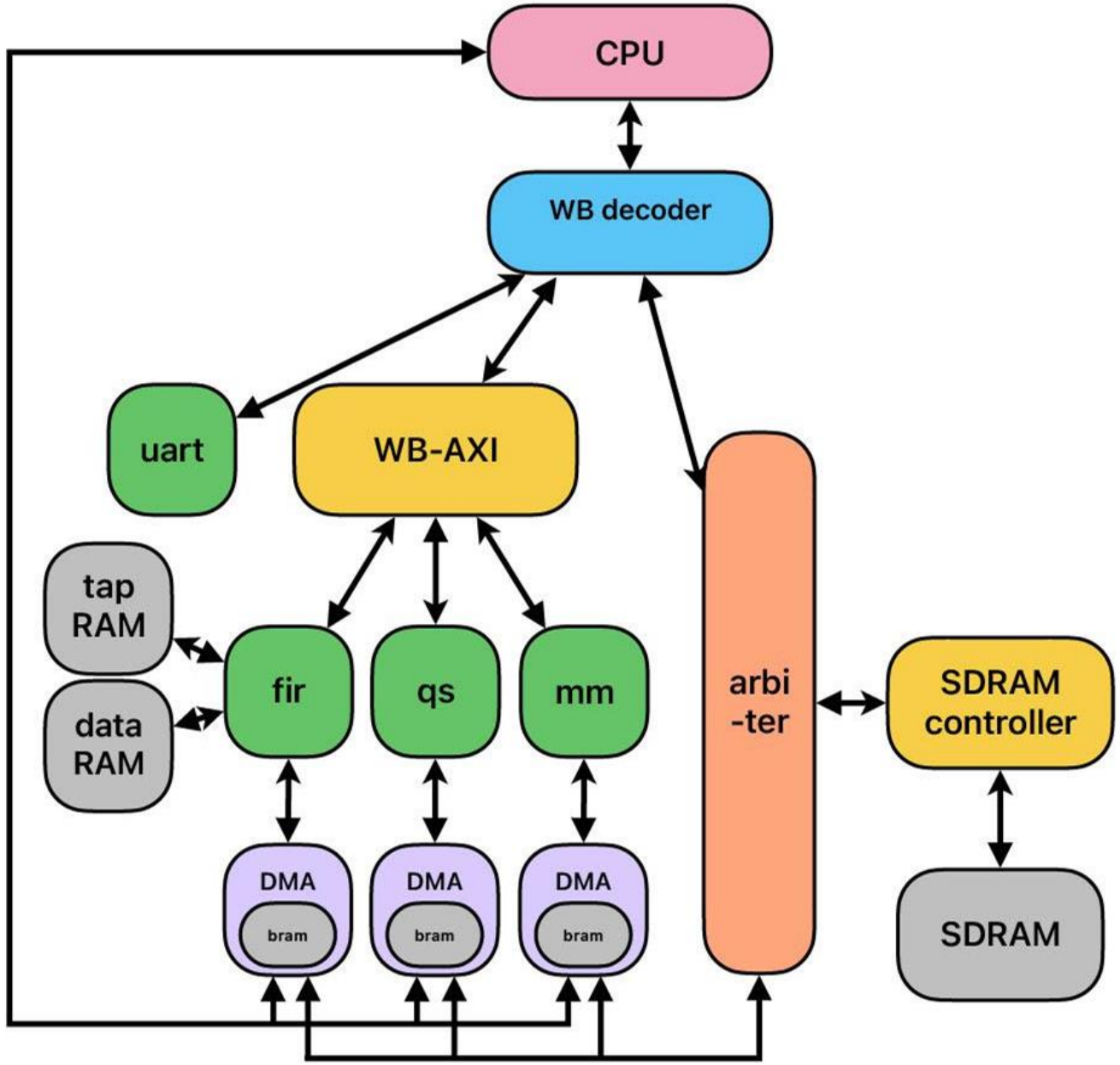
https://github.com/bol-edu/HLS-SOC-Discussions/discussions/176

WLOS - WorkLoad-Optimized SoC – DMA-FIR

- Purpose:
  - Design accelerator with DMA capability
  - Design efficient memory system
- Memory System to offload RISC-V
- Define DMA control scheme
  - Generate Interrupt when reach terminal count
- Memory system performance
  - Pipeline Memory access
  - SDRAM Bank interleave
  - Prefetch Scheme

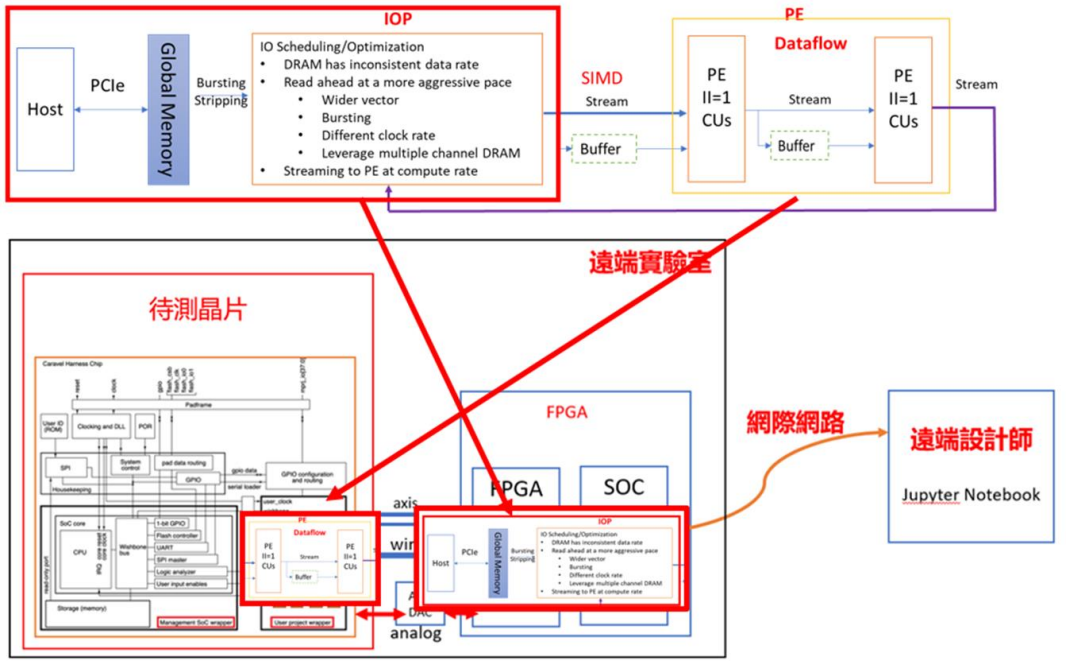


Block Diagram



SOC Validation – FSIC-FIR

- FIR is integrated with FSIC to interface with external FPGA to access test data (in FPGA DDR)
- Implement an HLS AxiDMA engine (similar to Xilinx IP)
- Validation program is implemented on PS/Python code
- This completes the whole SOC design process. You learn
  - FPGA design
  - Embedded Programming
  - IC Design



Advanced SOC Design

Advanced SOC Design (2nd Semester)

Objective:

1.Learn Advanced topics in IC Design, SOC chip-level design

2.Develop an Application Accelerator using Verilog/HLS (Catapult)

3.Complete IC design flow, and be ready for tape out.

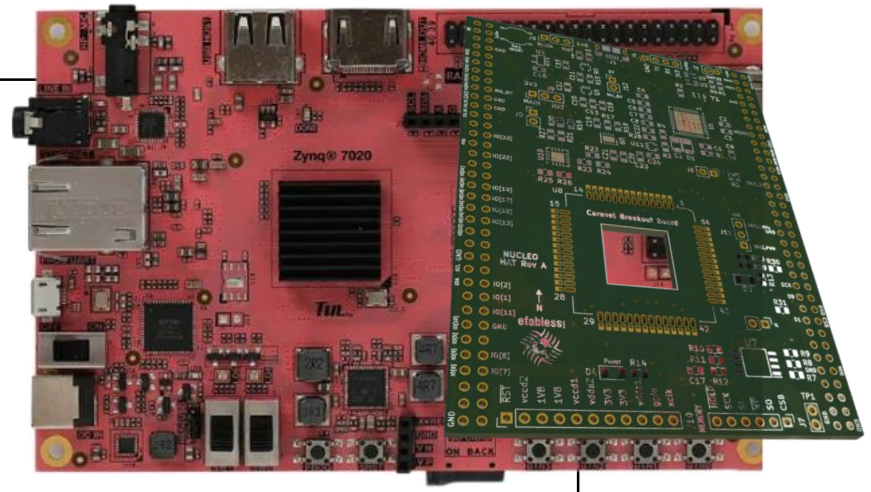
Design
1.Accelerator design with High-Level Synthesis
2.IC Design Flow
3.Low Power Design
4.Design for Test
5.SOC Chip Level Design
6.Advanced Static Timing Analysis
7.Advanced Verification (SV, UVM, Formal)
8.Synopsys front-end/backend flow, signoff

Design Flow
1.Catapult (ASIC HLS)
2.Synthesis
3.Timing Analysis
4.Floorplan
5.Placement
6.Clock tree
7.Routing
8.DFT and Testing
9.LVD, DR · ERC
10.Post-layout Timing Analysis

Laboratory
1.Algorithmic HLS Lab
2.FSIC Module Design & Verification
3.EDA frontend/backend flow
4.Application Accelerator Development
5.FSIC Integration and Simulation
6.FSIC Physical Implementation / Signoff

Lab Platform - FSIC Validation System

FSIC - IC Validation System

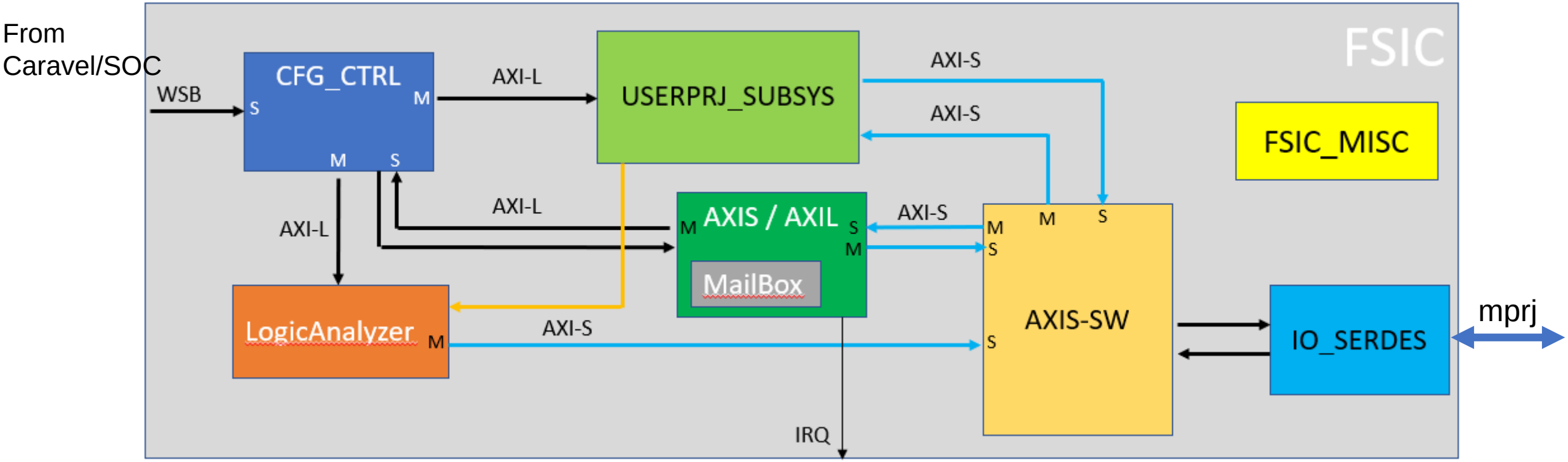


Caravel Chip DUT

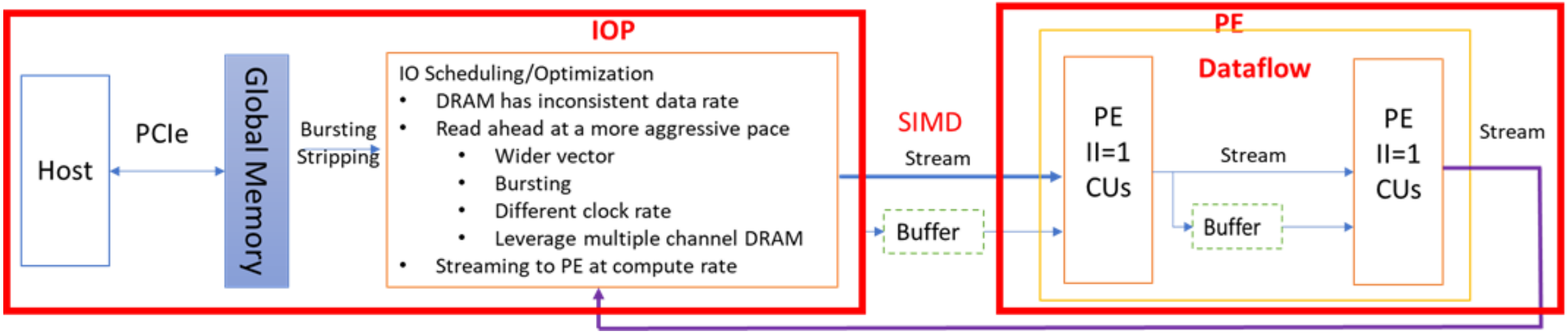
Caravel Harness Chip

reset
clock
gpio
flash_csb
flash_clk
flash_io0
flash_io1
mprj_io[37:0]
Padframe
User ID (ROM)
Clocking and DLL
POR
SPI
System control
pad data routing
GPIO
Housekeeping
gpio data
serial loader
GPIO configuration and routing
user_clock
wishbone
2.92mm x 3.52mm
logic analyzer
IRQ
Protocol Coversion
Logic Analyzer
Management Protect
SoC core
CPU
core reset
core clock
IRQ
Wishbone bus
1-bit GPIO
Flash controller
UART
SPI master
Logic analyzer
User input enables
read-only port
Storage (memory)
Management SoC wrapper
User project wrapper
axis
PYNQ FPGA
PL
Data Mover
PS
WAN
Remote Host
Jupyter Notebook

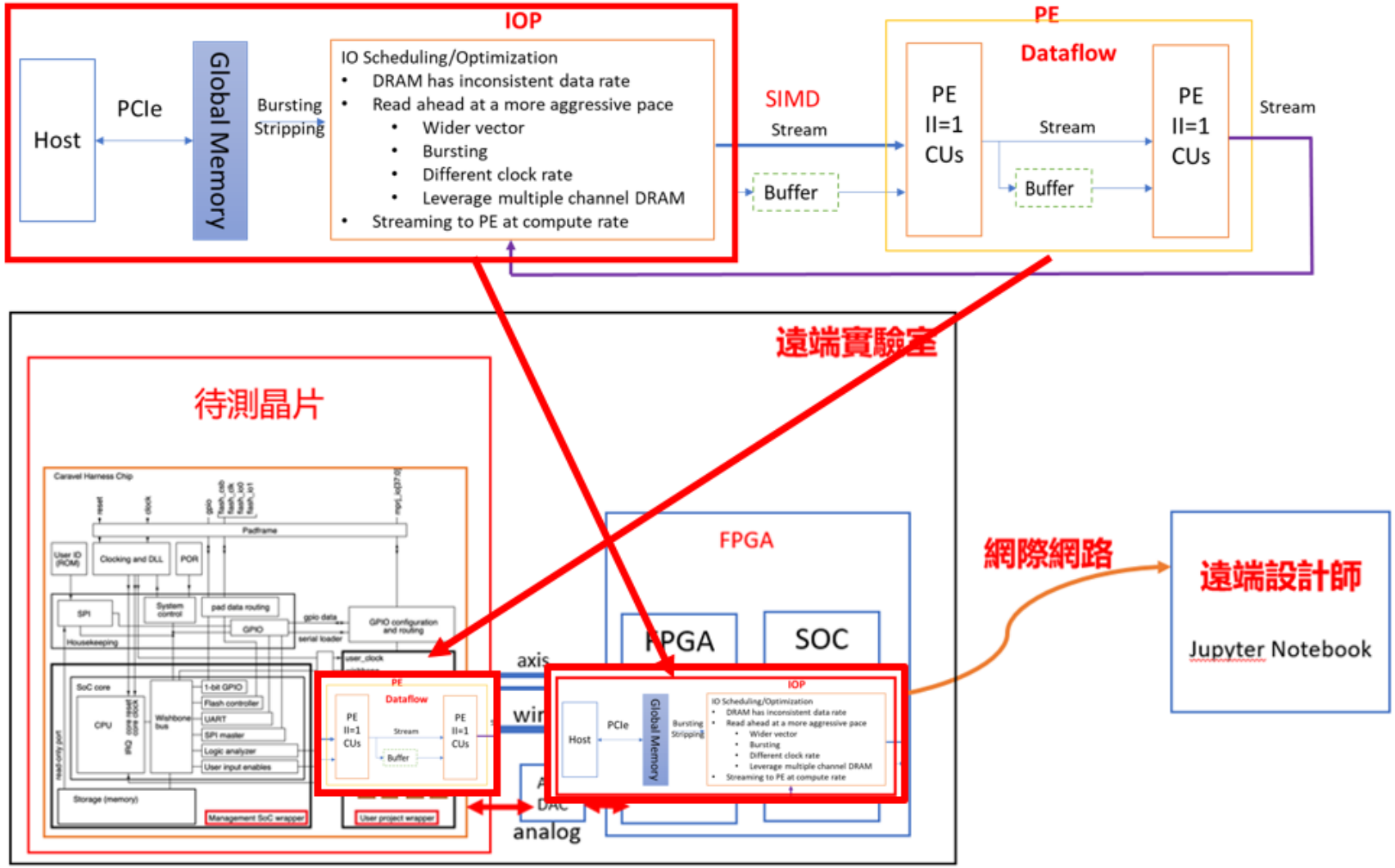
User Project Wrapper



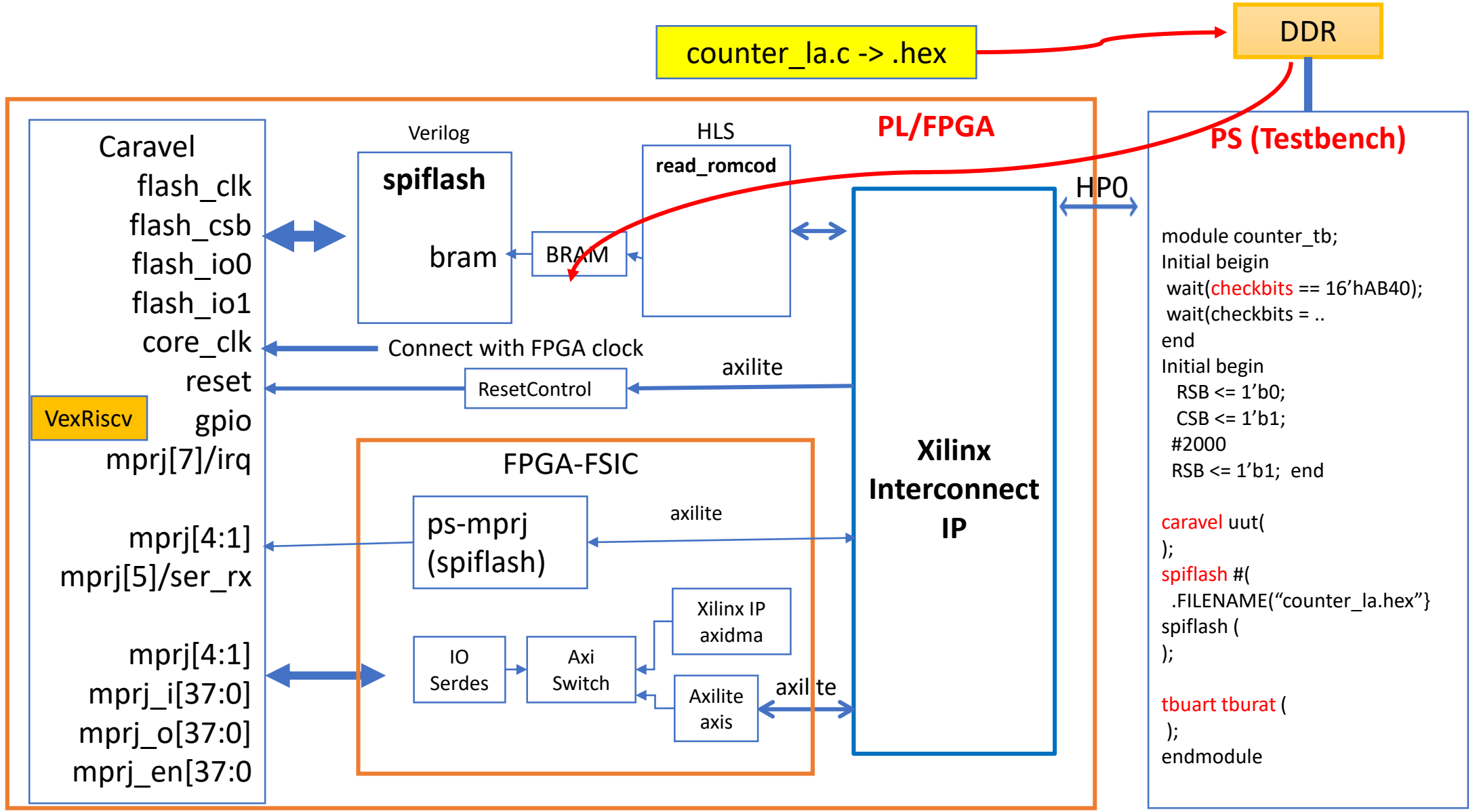
Architecture for Application Accelerator Design



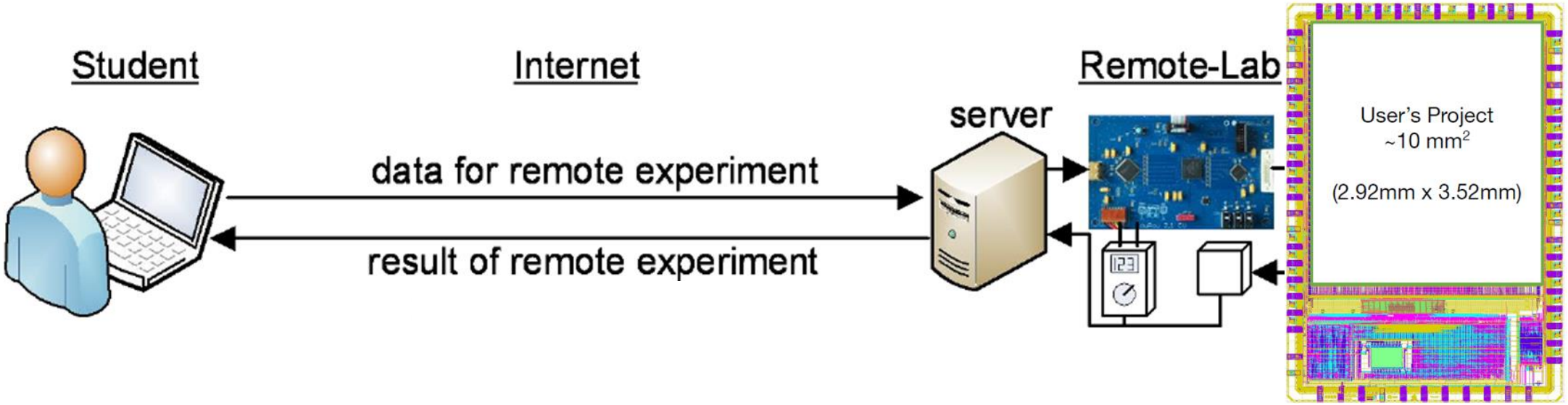
Embed Application Accelerator in FSIC



Caravel-FSIC + FPGA-FSIC



Online Lab System



Course Materials

- Google Drive (https://drive.google.com/drive/folders/1cGNs2LfW5ZdoaEpi_Nh6iANl7V4f6rtv?usp=sharing)
  - Lecture-ppt
  - Lab-note
  - Class-note
  - Lab-presentation
- Youtube https://www.youtube.com/playlist?list=PLTA_T2FLzYNDzhWnjCJR22_dvbZMmrEkf

Note: Course and Lab material shall not be put into Public area.

Collaborative Learning & Participate in Q & A

HLS and SOC Design is in its evolving development process. It is hard to teach the subject in academic environment where known and matured knowledge is taught. The only way to learn it is through collaborative learning, through a Q & A process.

Subscribe the following Slack, and Github to communicate idea.

- Github/Discussion Forum “HLS-SOC-Discussions”
    - https://github.com/bol-edu/HLS-SOC-Discussions
    - https://github.com/bol-edu/HLS-SOC-Discussions/discussions

For topics related to lecture contents, laboratory work and project implementation problems, please use **Github/Discussion** Forum

Lecture & Lab Schedule
Grade
Course Policy

Lecture & Lab Schedule

week	Date	In Class: Lecture, Presentation	Lecture	Lab	Lab Due
2nd Semester (2/19 - 6/17, 6/27)					
1	**22-Feb**	Course plan - FSIC Architecture	All	fsic-sim & fsic-plus (2w)	
2	**29-Feb**	Catapult HLS-1	All	catapul-fir	
3	**7-Mar**	Catapult HLS-2	All	catapult-hls (fir, edgedetect)	
4	**14-Mar**	Advanced HLS Topics - Memory, Best Practice, Architecture Examples	NTHU		fsic-sim
5	**21-Mar**	Chip Design Overview (chip manfuacture, design flow) snp lab introduction (video)	NYCU	snps-flow (2w) - tool study	fsic-plus
6	**28-Mar**	Design for Test (Scan, BIST, Boundary Scan, Iddq)	NTU	fsic-fpga (2w)	catapult-fir/edgedetect (2w)
7	**4-Apr**	Lab Presentation #1 - fsic-sim, fsic-plus, catapult-hls			
8	**11-Apr**	Low Power Design	NTHU	fsic-ap (4w)	snps-flow
9	**18-Apr**	SOC Component 1 - Clock/PLL, Reset, Interconnect	NYCU		
10	**25-Apr**	Midterm	ALL		fsic-fpga final project-ap proposal
11	**2-May**	Final Project Proposal - fsic-fpga lab presentation	ALL		
12	**9-May**	SOC Component 2 - IO, Power, Serdes	NTU		mid fsic-ap status report
13	**16-May**	Synopsys Tool 2 - dc, icc2	ALL	fsic-final (4w)	
14	**23-May**	Synopsys Tool 1 - startrc, icv, pt	ALL		fsic-ap
15	**30-May**	Advanced STA - Async/CDC, Source Synchronous, Time borrow, CrossTalk&Noise, Statistical Timing Analysis	NTHU		mid fsic-final status report
16	**6-Jun**	Verification -CDC, low power(UPF), UVM, System Verilog, Formal, coverage, assertion	NYCU		
17	**13-Jun**	NTU: Final Project Presentation	ALL		6/17 NTU Grade deadline
	19-Jun	NTHU, NYCU Final Project Presentation	ALL		7/5(NTHU), 6/28(NYCU)

Lecture Time Arrangement

- Each Lecture content will be lectured once, each school take turns
- Class without lecture, will have Q & A sessions and/or Quiz game
  - Code review – get one point credit if selected
  - Propose your Question one day before class – get one point credit if selected, answered in the class
  - Propose Quiz – get one point credit if adopted

Submit Question, Quiz by Tuesday noon for NTHU/NYCU, Wednesday noon for NTU

- Quiz game is team-based
  - Come to classroom for discussion with teammates.

Grading

Item	Content		Submission	Weight
Lab-fsic-sim	FSIC Design & Simulation Environment (FIR) - individual	Individual	Github & Report	6
Lab-fsic-plus	Module Improvement: CC, AA, IS, AS, LA, TestBench	Team	Github & Report (ppt)	5
Lab-catapult-fir	Catapult HLS - Lab-fir	Individual	Github & Report	5
Lab-catapult-edge	Catapult HLS - Lab-EdgeDetect	Team	Github & Report (ppt)	6
Lab-snps-flow	Synopsys frontend/backend Lab - FSIC	Team	Github & Report	15
Lab-fsic-fpga	Caravel FSIC fpga - integrate FIR + userDMA	Team	Github & Report	10
Lab-fsic-ap	HLS Application accelerator + FSIC	Team	Github & Report	12
	intermediate fsic-ap report			3
Lab-dft	Design for test (extra)	Team	Github & Report	5
Lab-lpd	Low power design flow （ extra ）- 4 subjects clock_gating, low_voltage, multi_vt, power_gating	Team	Github & Report	5
Midterm				6
Final Project	AP + FSIC + Caravel SOC integration - asic flow	Team	Github & Report & ppt	15
	intermediate final project status report			5
Synopsys Tool Study	Synopsys Tool study report	Team	Report	5
Contribution to Course	Adopted Question, Quiz, code review (each)	Individual		1
In-class Quiz	In-class quiz game	Team		3
Presentation	Selected presentation (extra credit)	Team/Ind	ppt	2
StudyJournal	Github & StudyJournal (individual)	Individual	Github & StudyJournal.md	2
Soft Skill	Ask question, offer help, sharing	Individual	Send link of evidence to TA	2
			Total	113

Study Journal

- It is By you, and For you
- Benefit
  - Thought-starter and planning
  - Note of accomplishments, keep track progress
  - Material for report
  - Space for reflection, Facilitating a breakthrough
  - Maintaining the writing habit

- Reference: https://hackmd.io/@TonyHo?utm_source=preview-mode&utm_medium=rec

Soft Skills

- Participate Q & A in
  - In-class – Question & Answer
  - For general topics of HLS and SOC (e.g., lecture contents, labortory work and project implementation problems), use Github/Discussion Forum
    - https://github.com/bol-edu/HLS-SOC-Discussions
    - https://github.com/bol-edu/HLS-SOC-Discussions/discussions
  - Submit question to Public Forum
- Offer help to solve questions
- Share your findings

At semester end, provide the link of evidence to TA for grade

About Report

- Report shall include
  - Github link
  - Report placed in Github
  - State work assignment & deliverable by each team member
    (Each team member shall contribute to the project with your best)

- Progress Report (for lab:fsic-ap, final project)
  - Current Status check against planning
  - Technical difficulties, bottleneck, and resolution
  - Remaining/Todo Item

Question / Quiz Submission Form

https://forms.gle/MSDJ5rkCubgBQm1r7