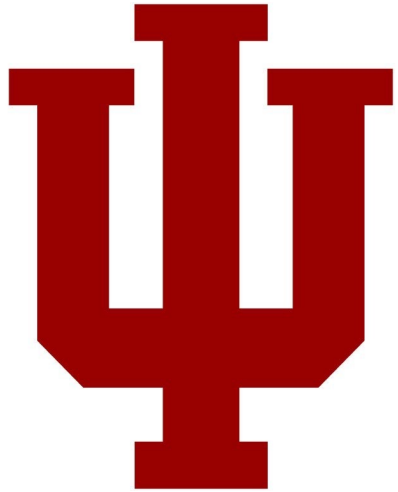
09: High-Performance Buses

Engr 315: Hardware / Software Codesign

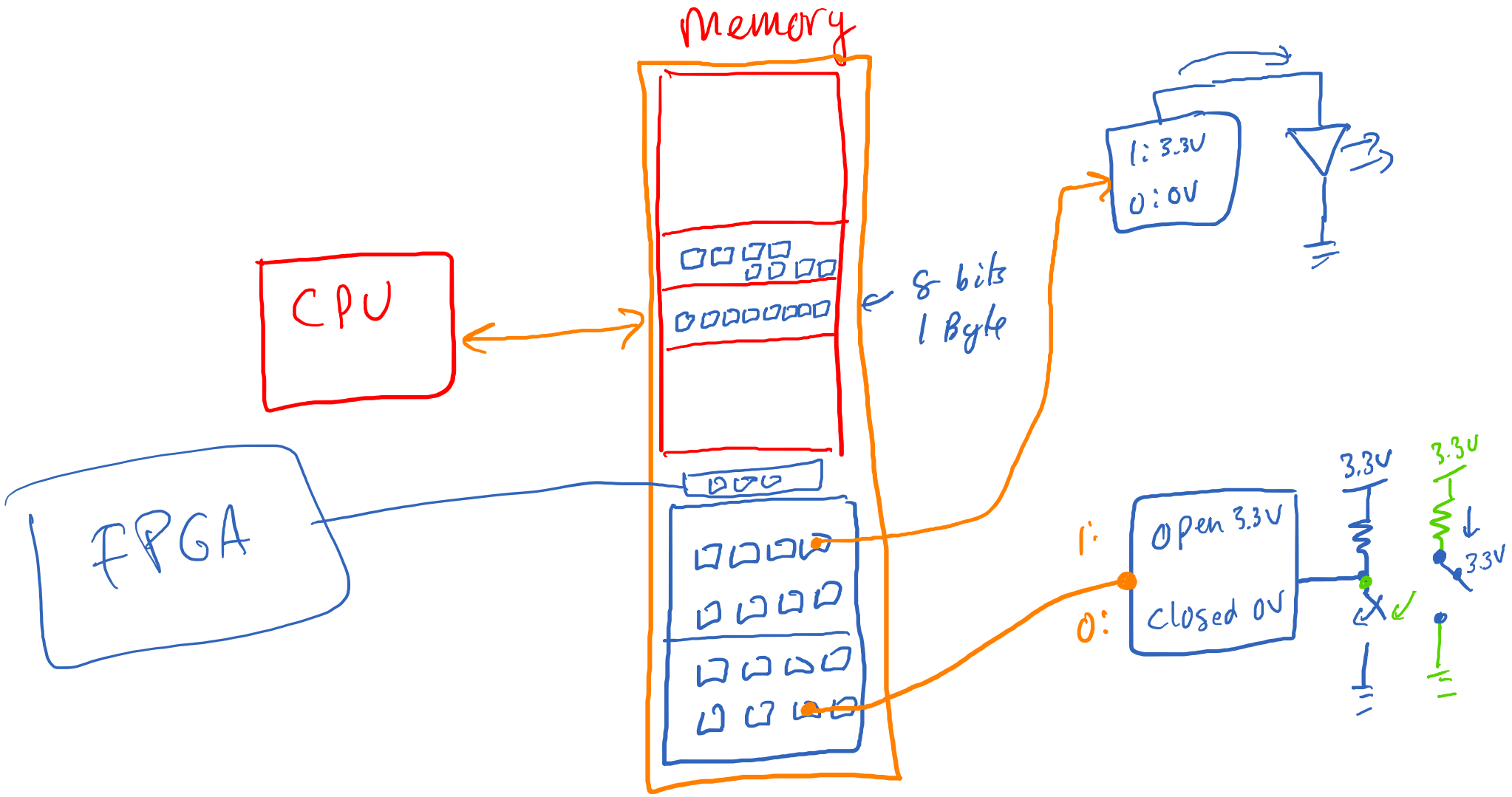
Andrew Lukefahr
Indiana University



Announcements

- P4: Due Next Wednesday. → AG + b different from project.
- P5: Out soon.

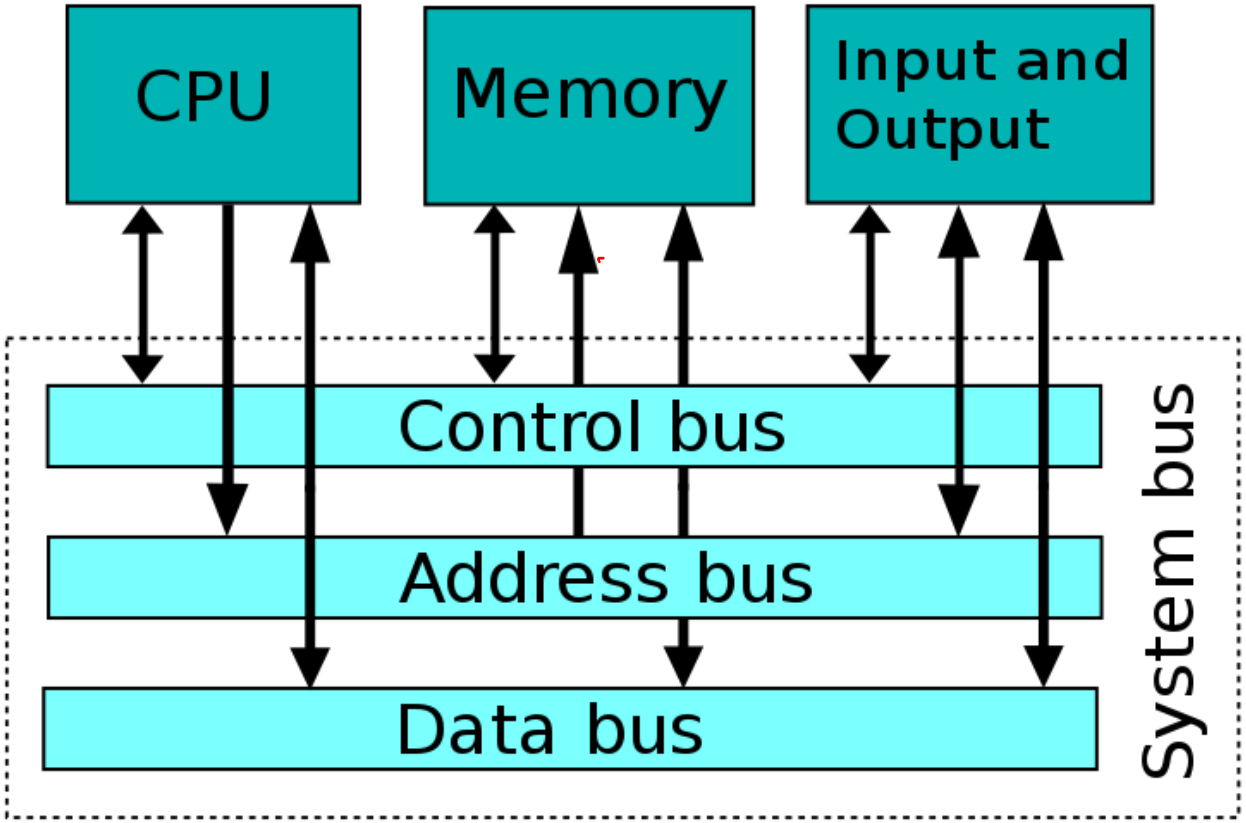
Review: Memory-Mapped I/O



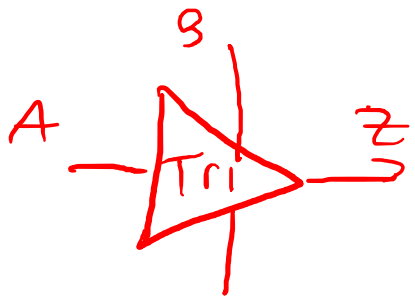
Use `volatile` for MMIO addresses!

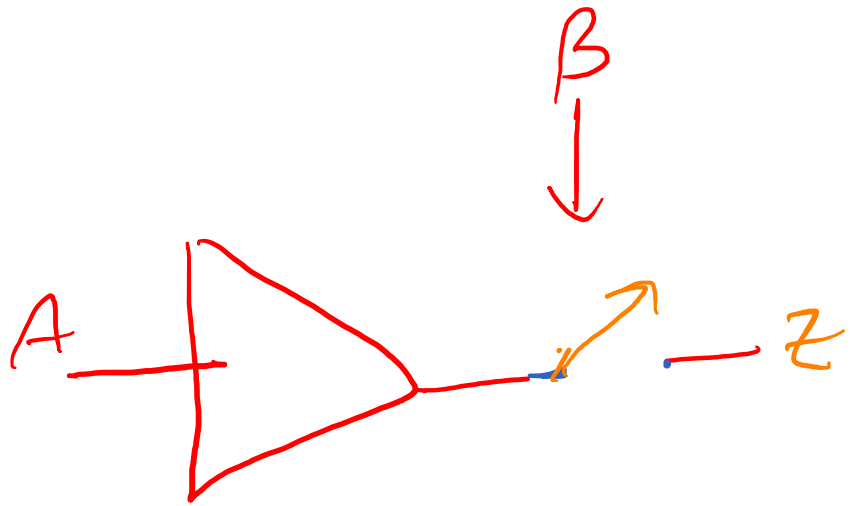
```
#define SW_ADDR 0xfffe
volatile uint32_t * SW_REG = (uint32_t * SW_ADDR);

int quit = (*SW_REG);
while(!quit)
{
     //more code
     quit = (*SW_REG);
}
```

The System Bus



Tri-State Buffer





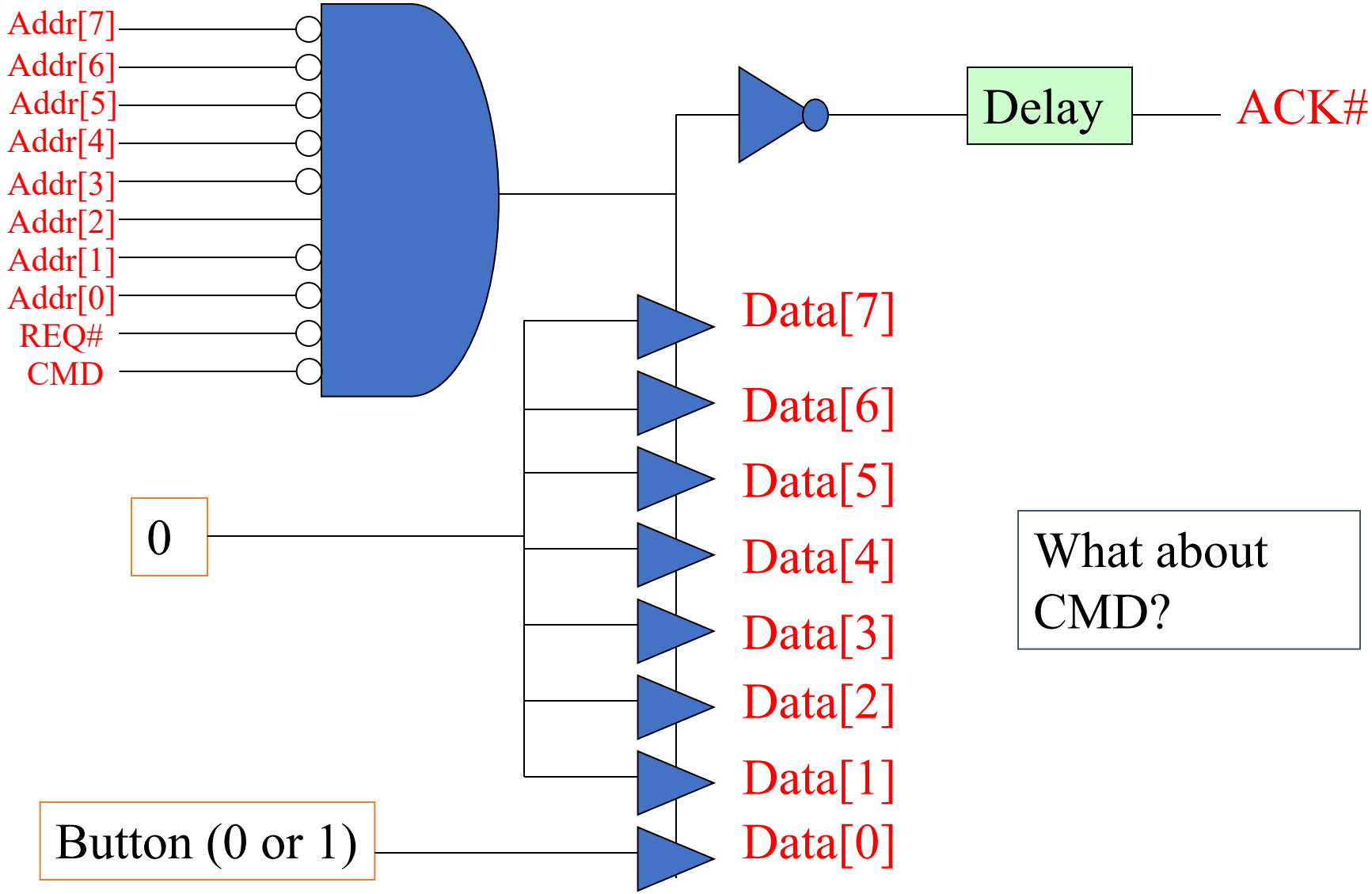
A	B	Z
0	0	High Z
0	1	0
1	0	High Z
1	1	1

A	B	Z
0	0	High-Impedance (High-Z)
0	1	0
1	0	High Z
1	1	1

Tri-State: can be 0, 1, Z

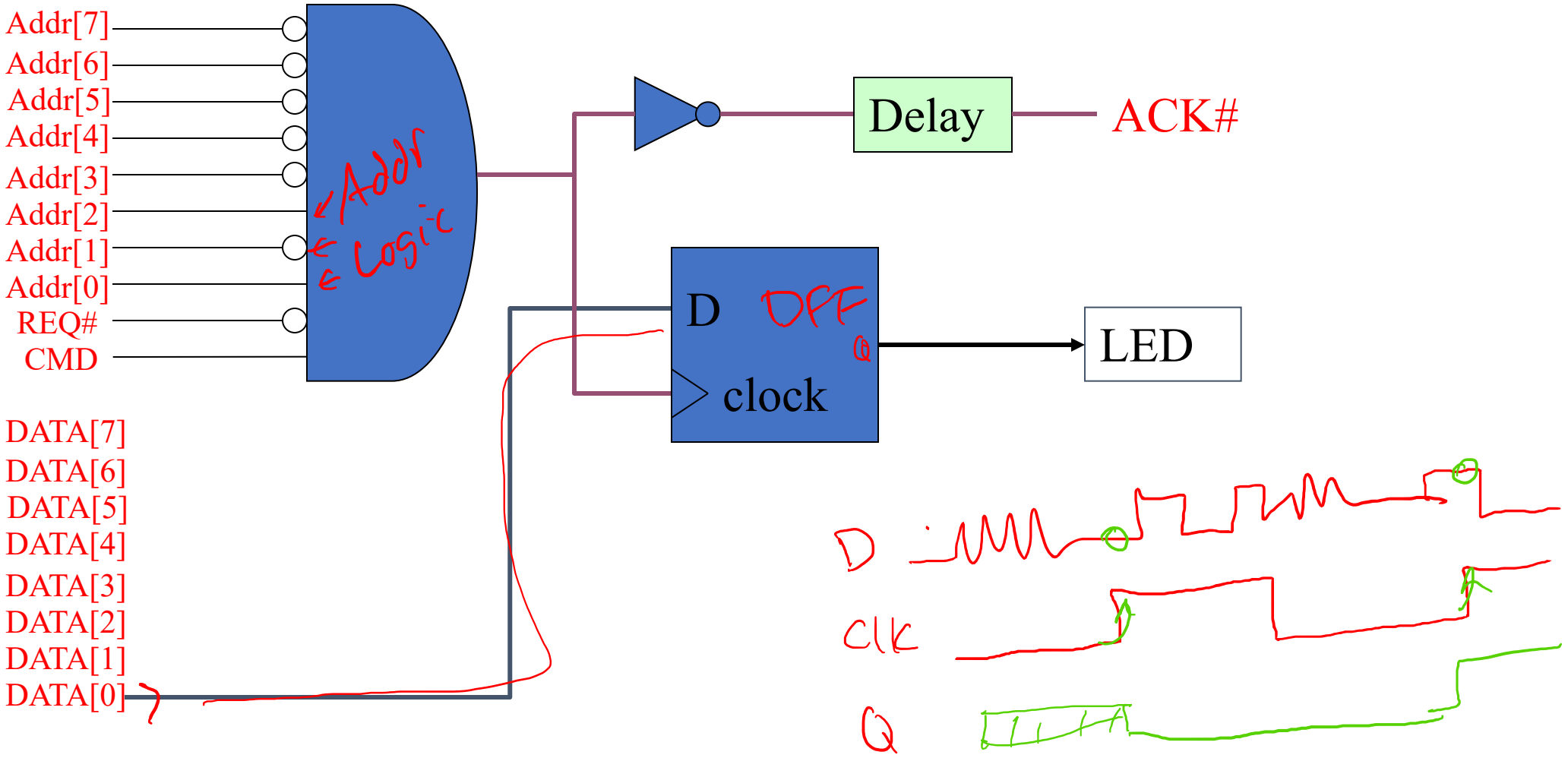
The push-button

(if Addr=0x04 read 0 or 1 depending on button)



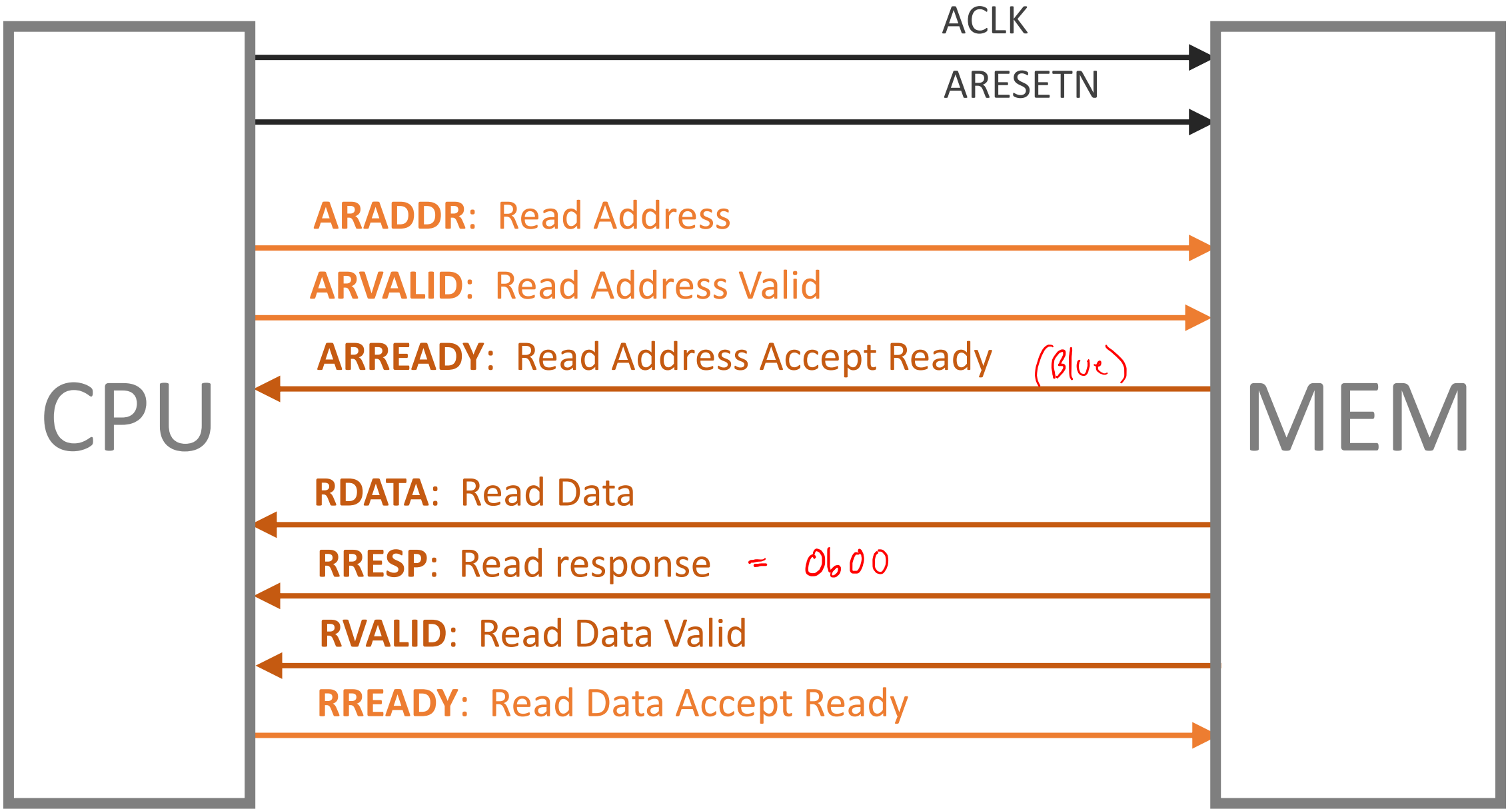
The LED

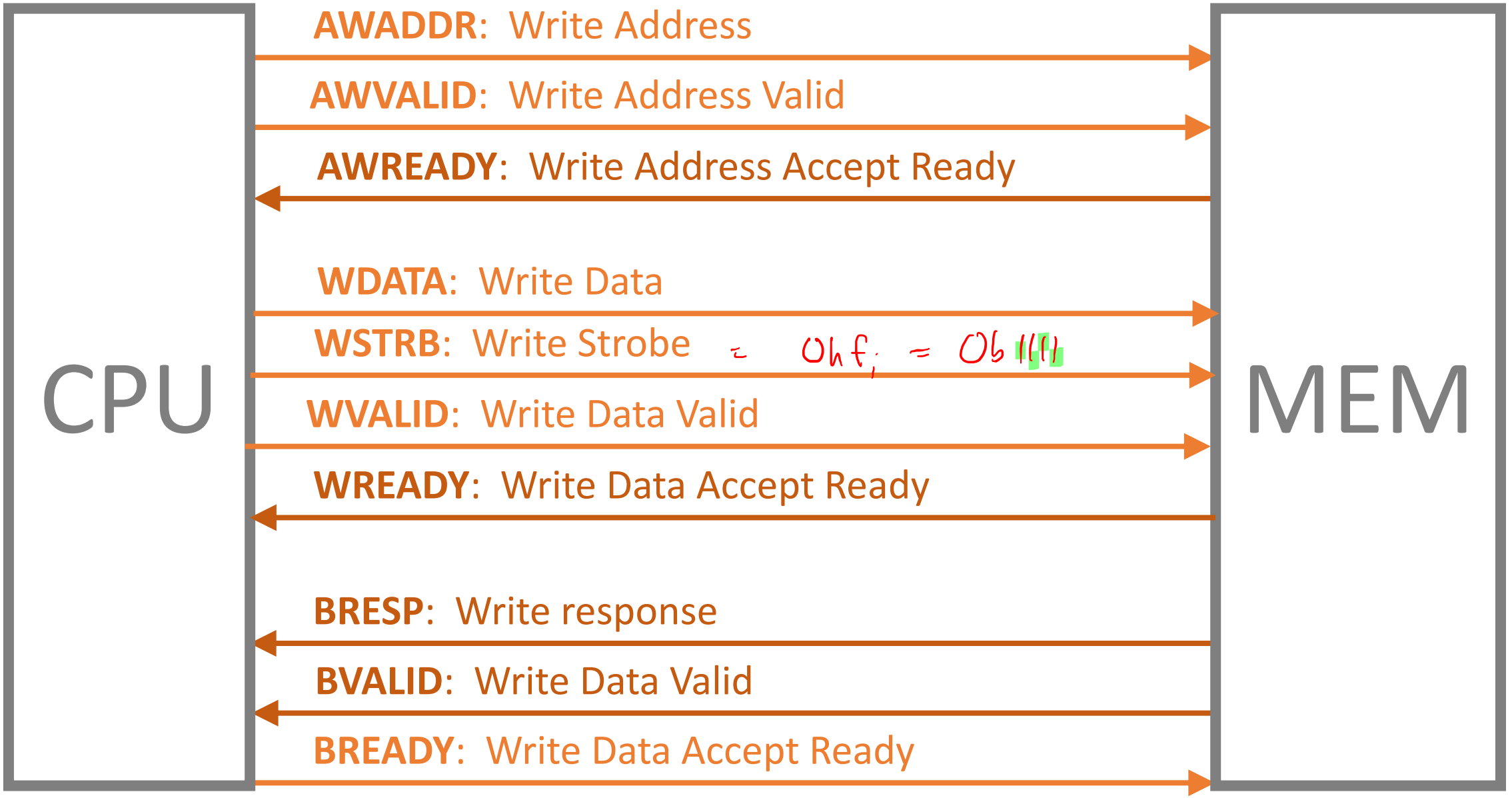
(1 bit reg written by LSB of address 0x05)



ARM AXI Bus

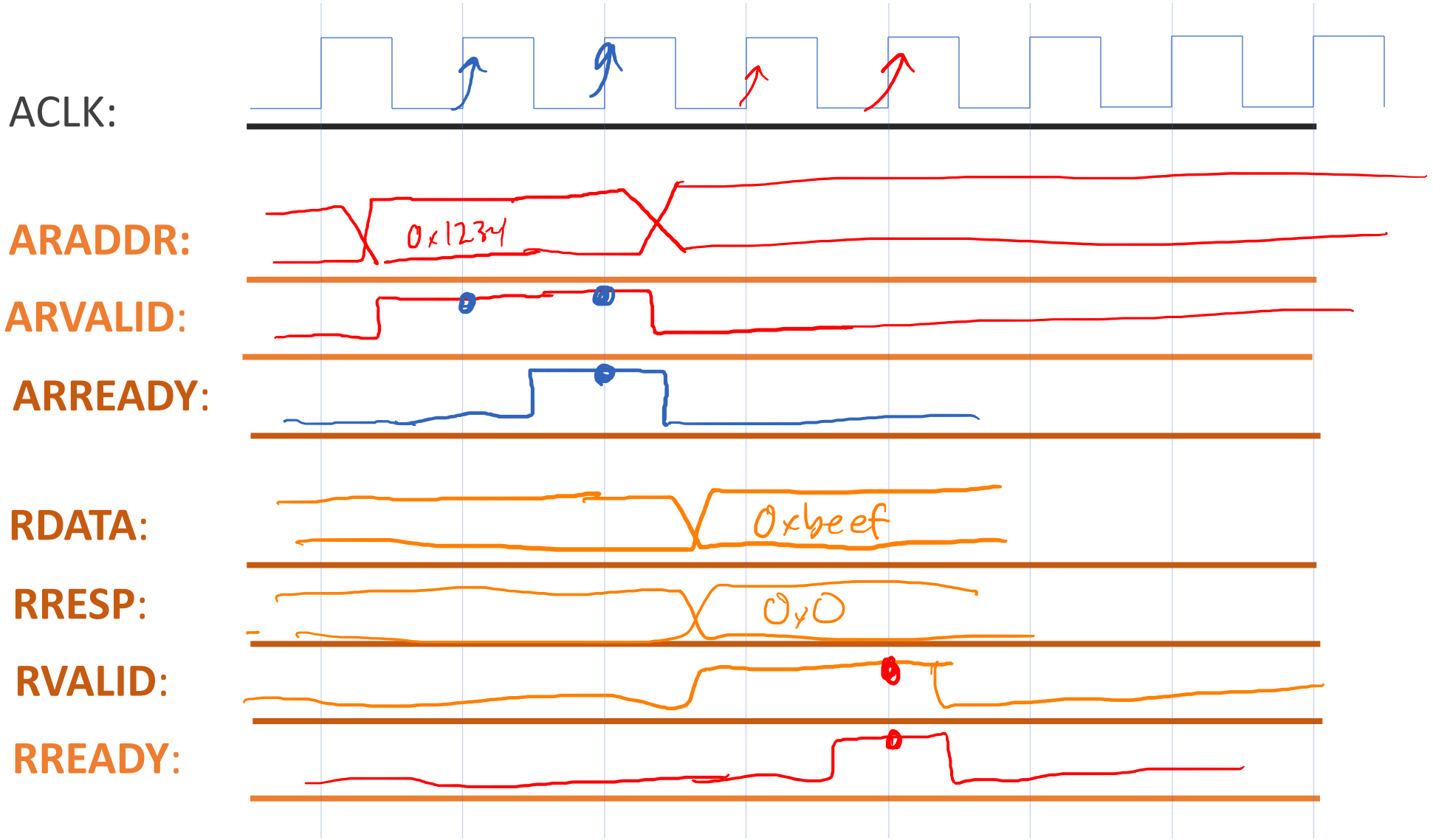
- "Advanced eXtensible Interface" Bus Version 4, "AXI4"
- Three Variants
  - AXI4: Fast but complicated; Memory-mapped
  - AXI4 Lite: Slow but simple; Memory-mapped
  - AXI4 Stream: Fast and simple; Not memory-mapped


ACLK
ARESETN
CPU
MEM
ARADDR: Read Address
ARVALID: Read Address Valid
ARREADY: Read Address Accept Ready
(Blue)
RDATA: Read Data
RRESP: Read response = 0b00
RVALID: Read Data Valid
RREADY: Read Data Accept Ready



ACLK and ARESETN not shown

How long does a read(load) take?



where is store (write)

High-Performance Bus Ideas

- Make single transaction faster

AXI Handshake Speedup

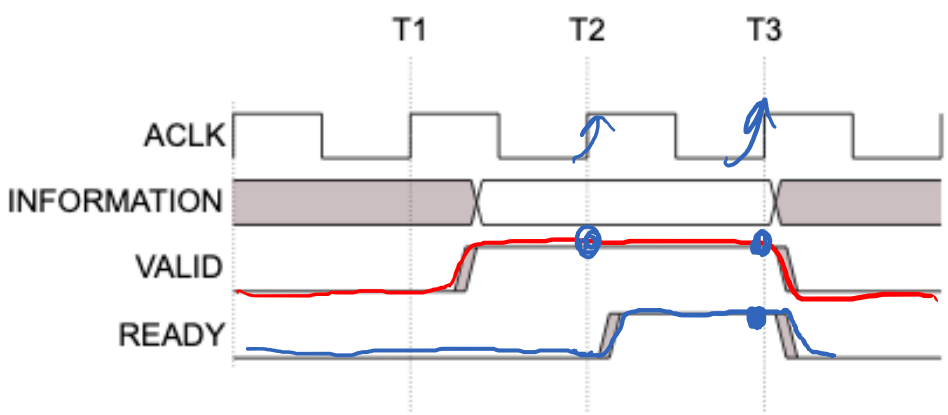


Figure A3-2 VALID before READY handshake

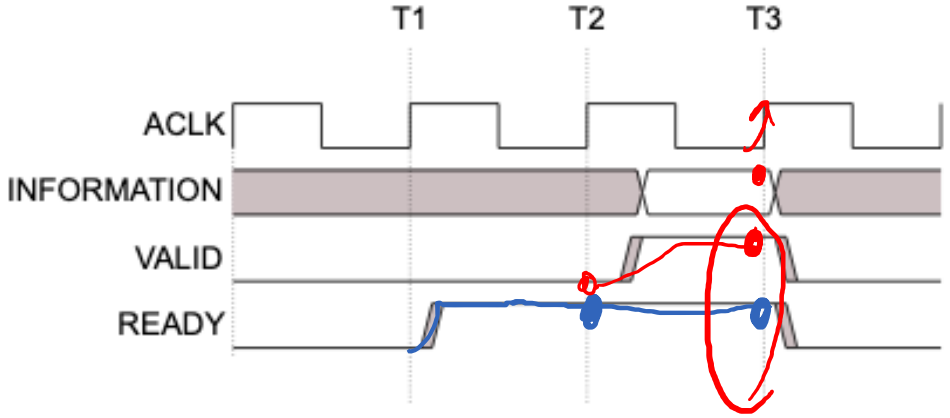


Figure A3-3 READY before VALID handshake

- Both are valid
- Figure A3-3 is faster

What happens here?

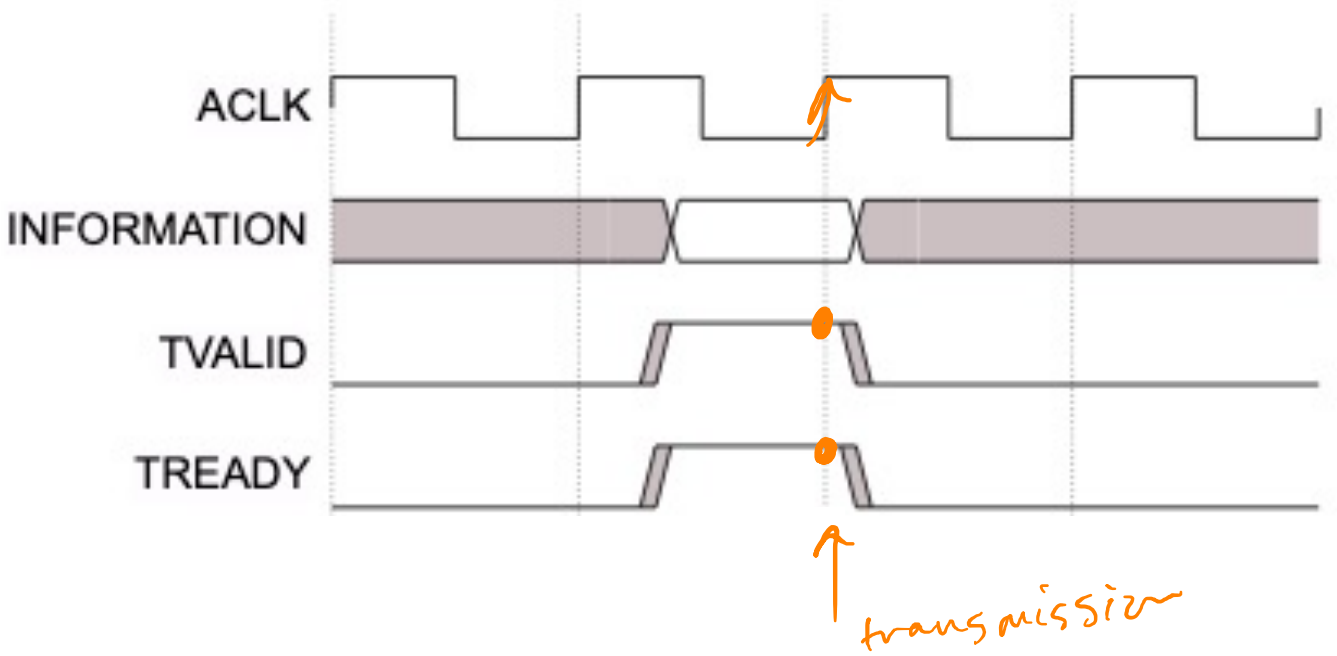
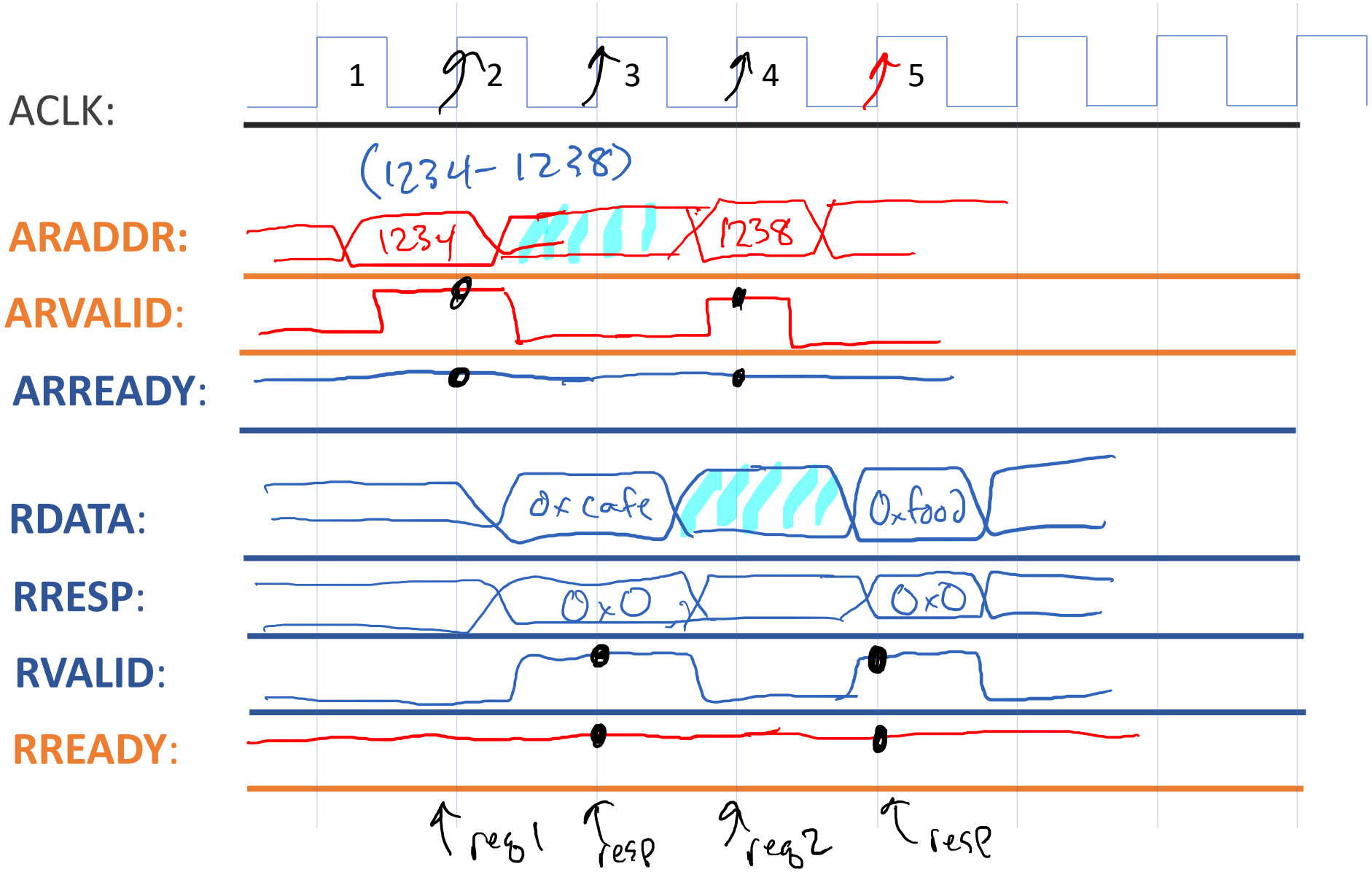


Figure 2-3 TVALID with TREADY handshake

What can we do to make this faster?

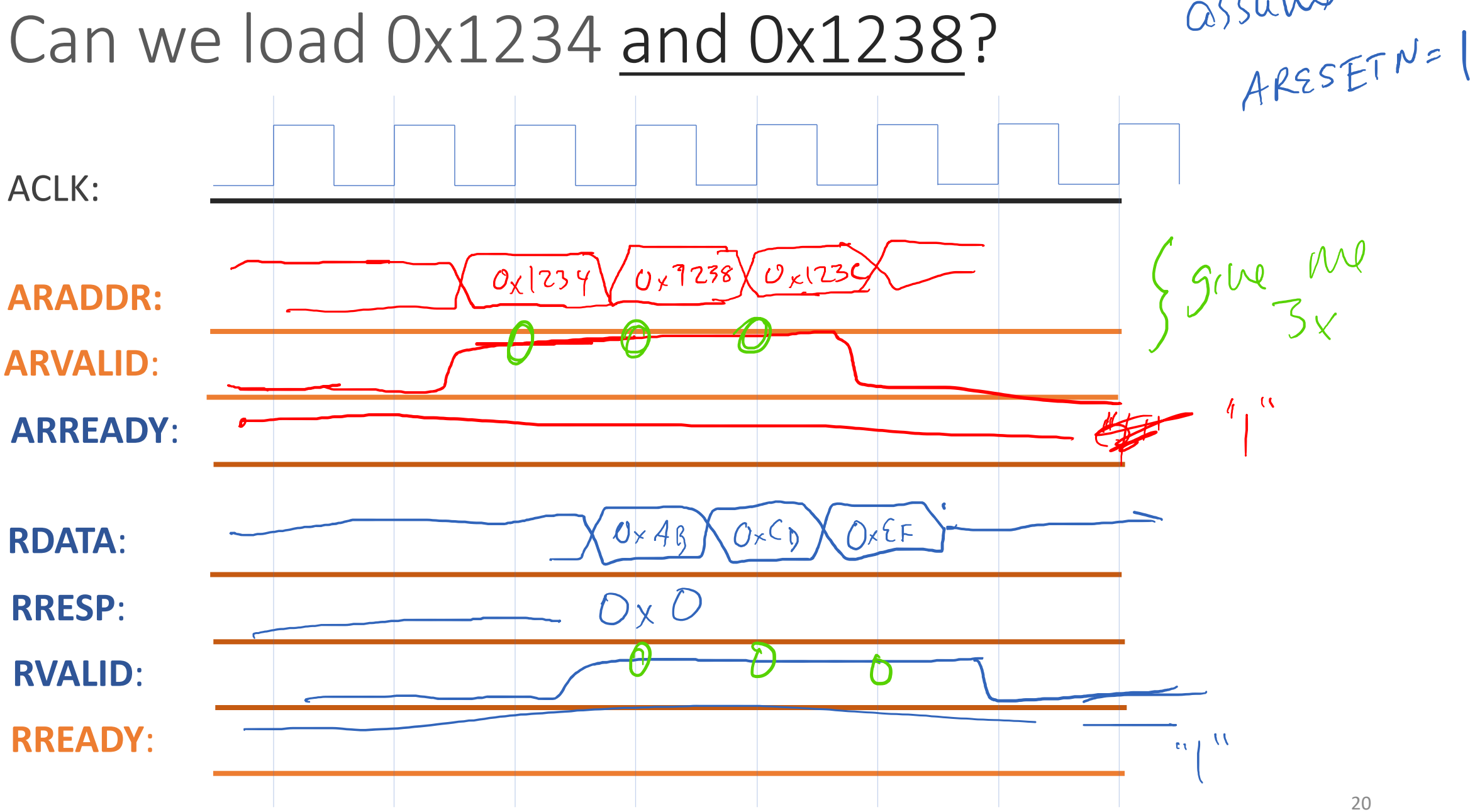


High-Performance Bus Ideas

- Make single transaction faster

- Overlap multiple transactions

Can we load 0x1234 and 0x1238?



Burst Transactions

- When a device is transmitting data repeatedly without going through all the steps required to transmit each piece of data in a separate transaction

Burst Transaction

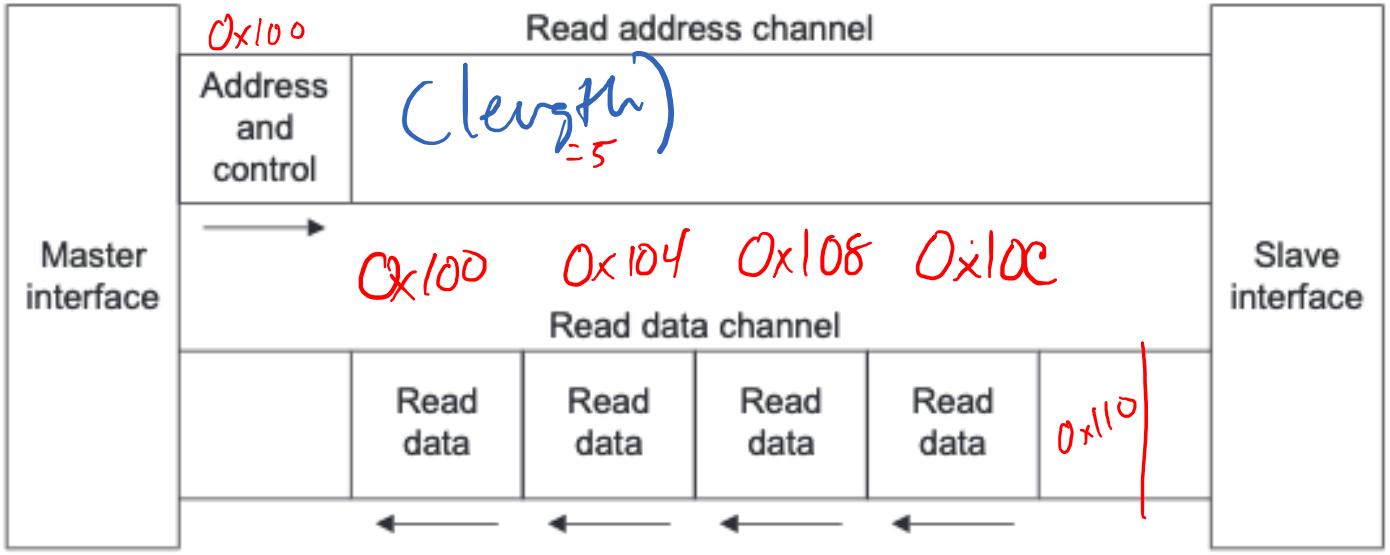
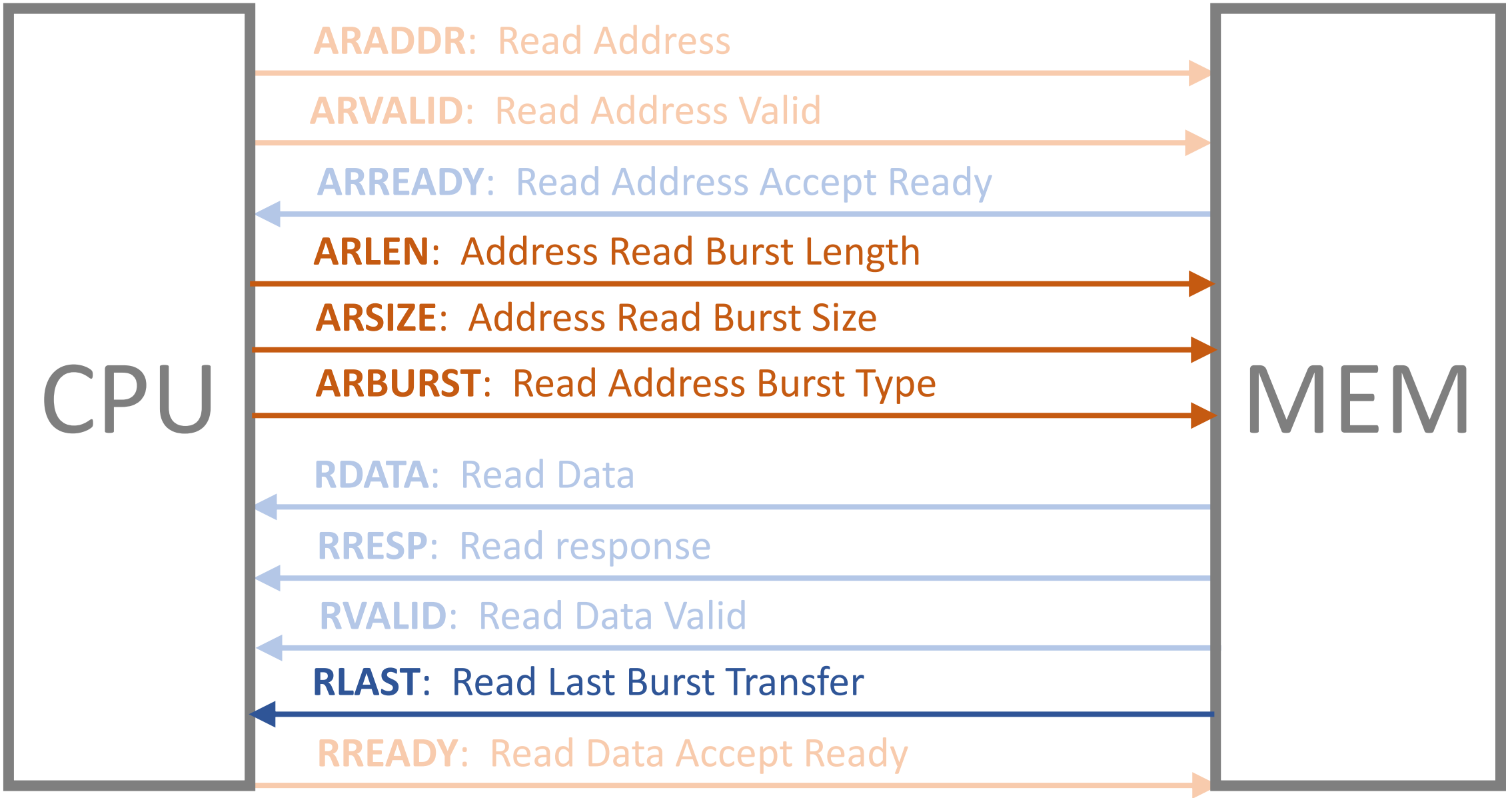


Figure 1-1 Channel architecture of reads


CPU
MEM
ARADDR: Read Address
ARVALID: Read Address Valid
ARREADY: Read Address Accept Ready
ARLEN: Address Read Burst Length
ARSIZE: Address Read Burst Size
ARBURST: Read Address Burst Type
RDATA: Read Data
RRESP: Read response
RVALID: Read Data Valid
RLAST: Read Last Burst Transfer
RREADY: Read Data Accept Ready

What do the new signals do?

ARLEN: Address Read Burst Length

How many bursts should occur? (+1)

Burst_Length = ARLEN[7:0] + 1

ARLEN = 0 => Burst Length = 1

ARLEN = 1 => Burst = 2

What do the new signals do?

ARLEN: Address Read Burst Length

How many bursts should occur? (+1)

ARSIZE: Address Read Burst Size

How many bytes should be in each burst?

2^n Bytes / Transfer

Table A3-2 Burst size encoding

AxSIZE[2:0]	Bytes in transfer
0b000	1
0b001	2
0b010	4
0b011	8
0b100	16
0b101	32
0b110	64
0b111	128

INCR: 0x100, 4 → [0x100], [0x104], [0x108], [0x10C]

Fixed: 0x100, 4 → [0x100], [0x100], [0x100], [0x100]

What do the new signals do?

ARLEN: Address Read Burst Length

How many bursts should occur? (+1)

ARSIZE: Address Read Burst Size

How many bytes should be in each burst?

ARBURST: Read Address Burst Type

Are the addresses incrementing, or repeating?

Table A3-3 Burst type encoding

AxBURST[1:0]	Burst type
0b00	FIXED
0b01	INCR
0b10	WRAP
0b11	Reserved

FIXED: The address is the same for every transfer (Next Address = Address)

INCR: The address for each transfer is an increment of previous transfer

(Next Address = Address + 0x4)

What do the new signals do?

ARLEN: Address Read Burst Length

How many bursts should occur? (+1)

ARSIZE: Address Read Burst Size

How many bytes should be in each burst?

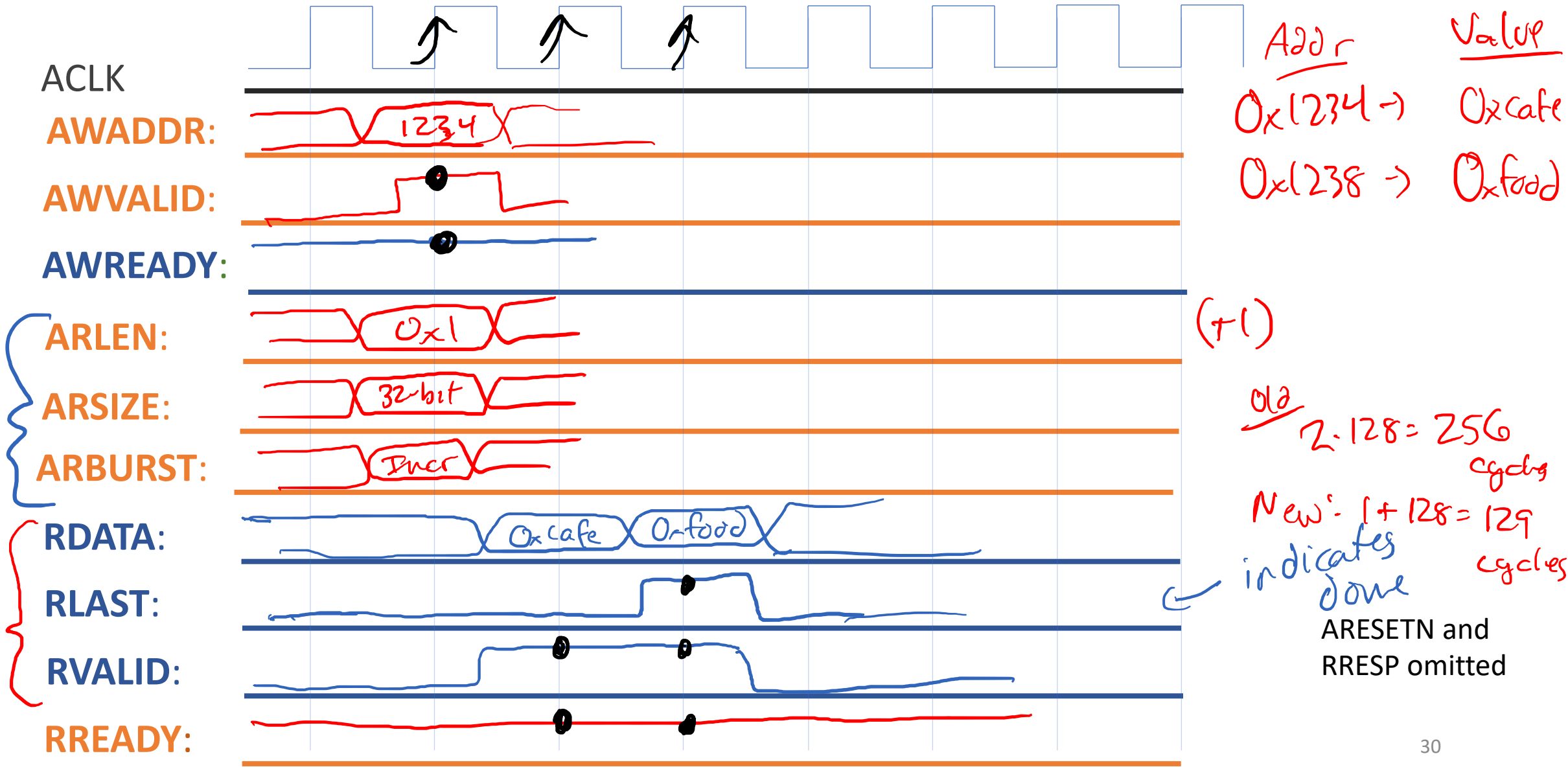
ARBURST: Read Address Burst Type

Are the addresses incrementing, or repeating?

RLAST: Read Last Burst Transfer

Are we done yet?

Reading 0x1234 and 0x1238



Read Burst Example

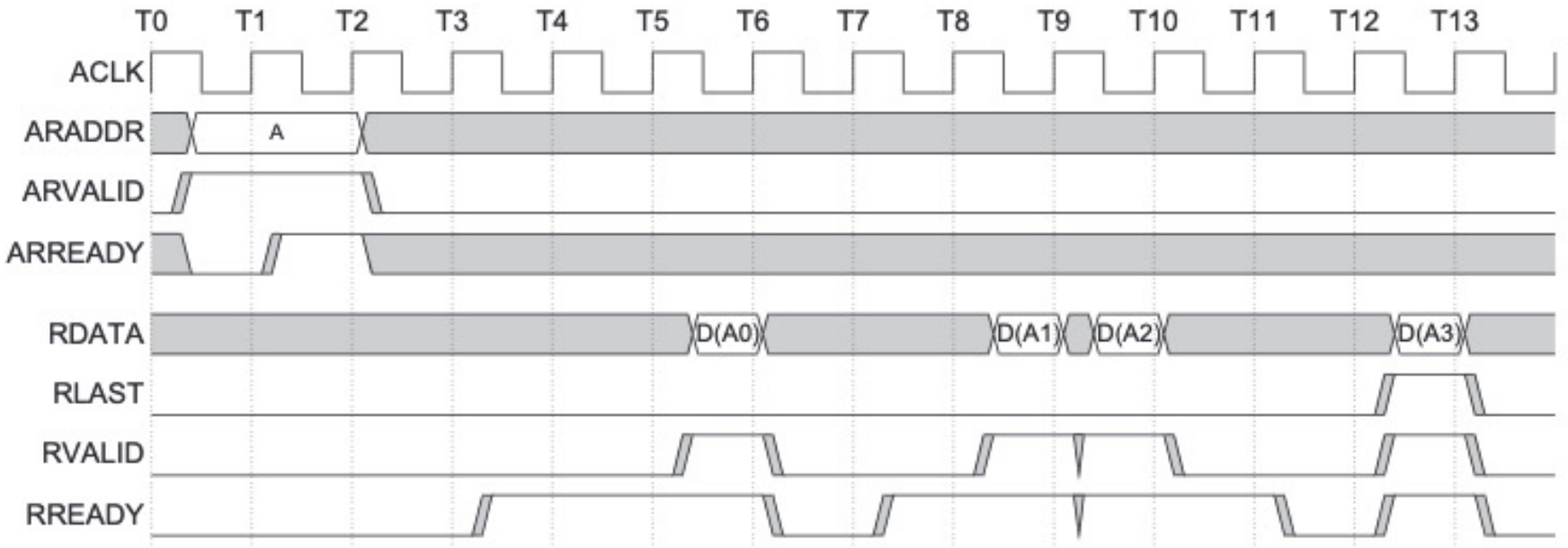


Figure 1-4 Read burst

Note

The master also drives a set of control signals showing the length and type of the burst, but these signals are omitted from the figure for clarity.

Overlapping Read Burst Transactions

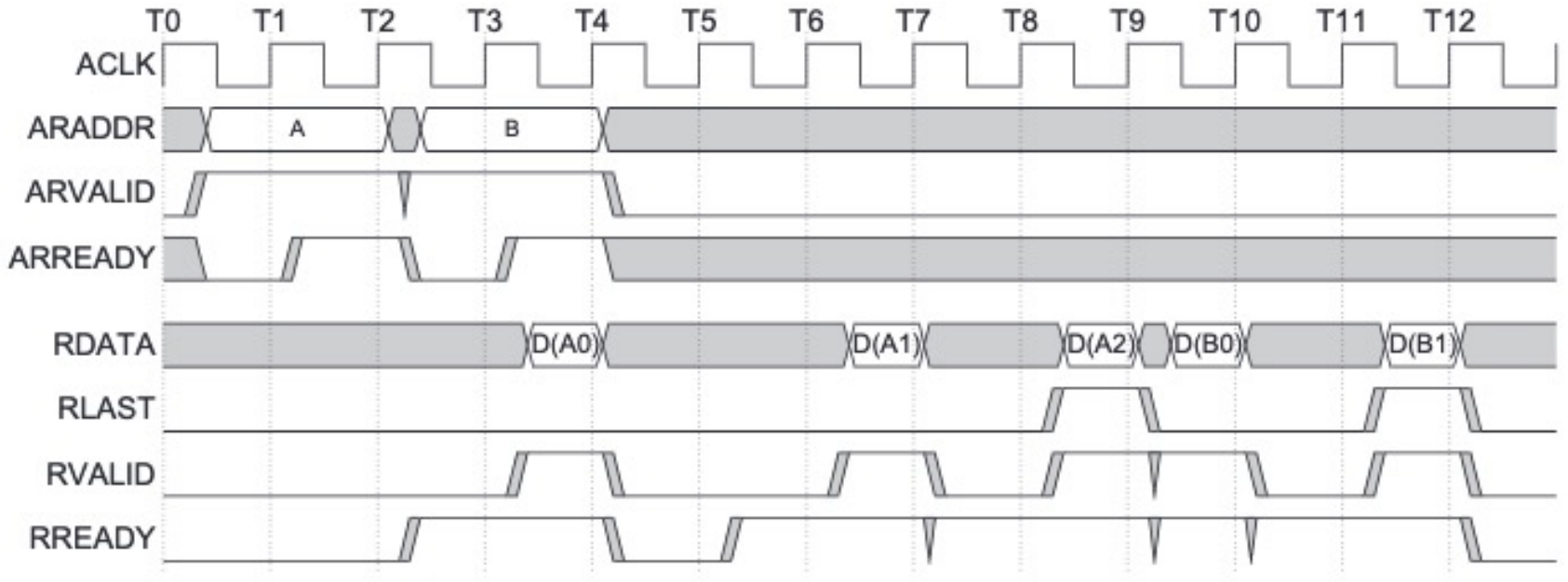


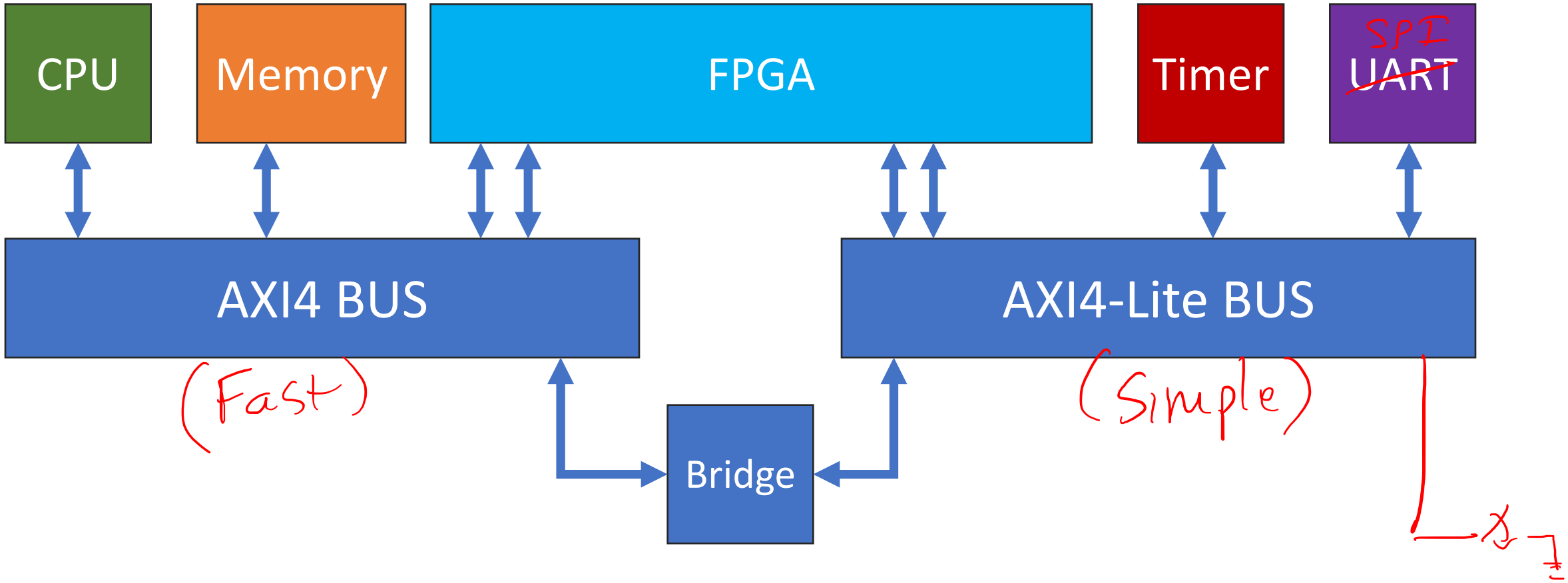
Figure 1-5 Overlapping read bursts

System’s Architecture

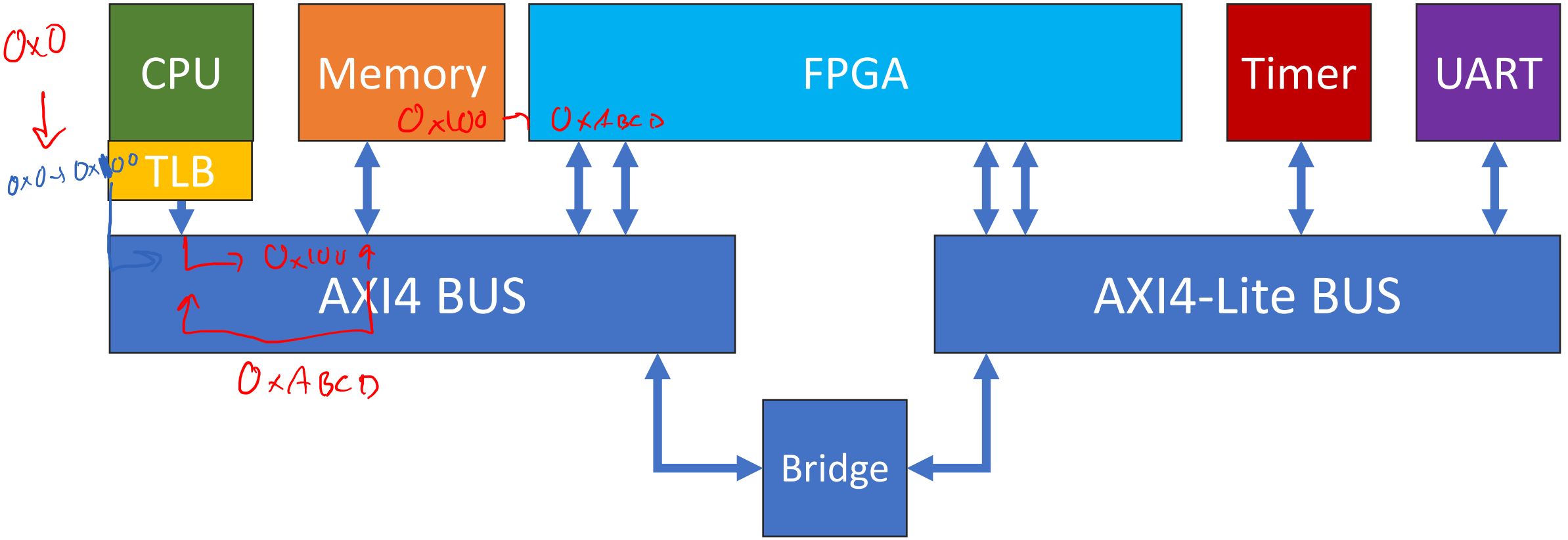
AXI4 BUS

AXI4-Lite BUS

System Architecture



Next Time: TLBs (CPU Memory Games!)



References

- Zynq Book, Chapter 19 “AXI Interfacing”
- Practical Introduction to Hardware/Software Codesign
  - Chapter 10
- AMBA AXI Protocol v1.0
  - http://mazsola.iit.uni-miskolc.hu/~drdani/docs_arm/AMBAaxi.pdf

09: High-Performance Buses

Engr 315: Hardware / Software Codesign

Andrew Lukefahr
Indiana University