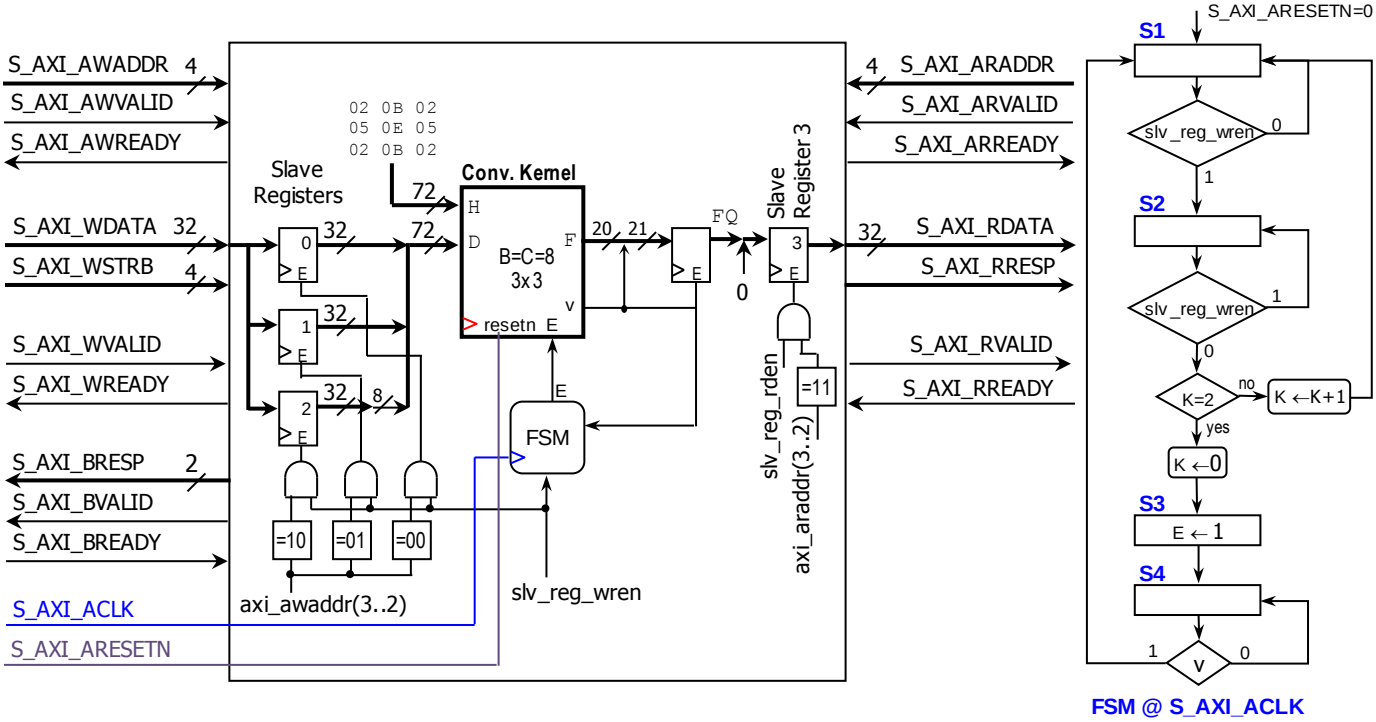
Homework 2

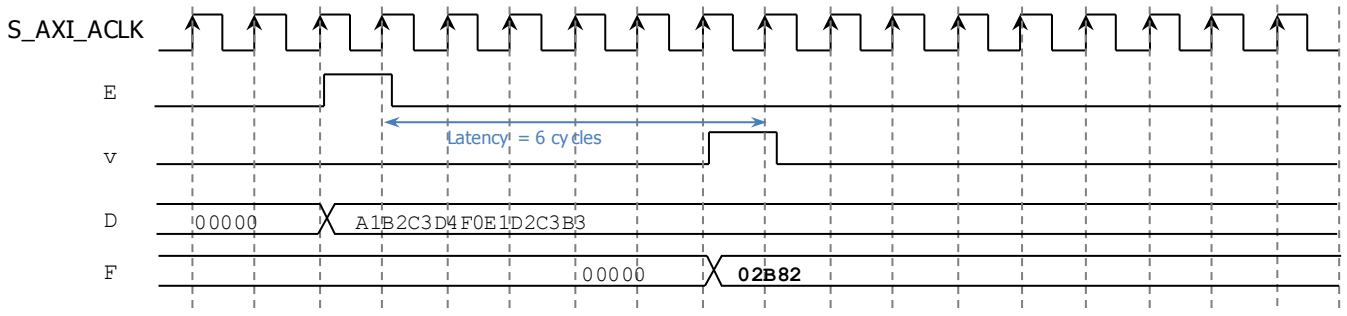
(Due date: May 25th)

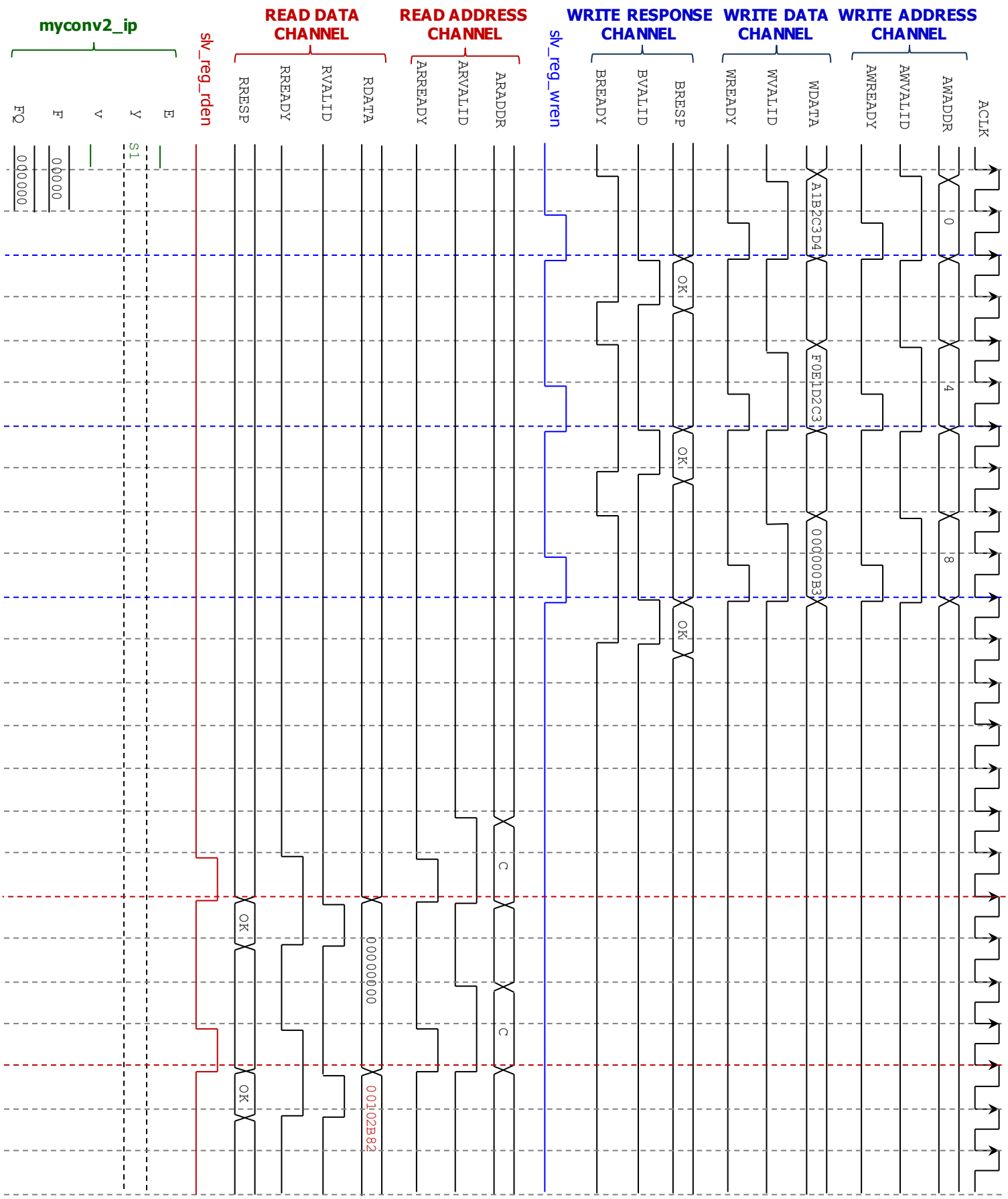
Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (30 PTS)

- AXI4-Lite interface for Pipelined 2D convolution kernel (N=3, B=C=8):
  - ✓ The I/O timing diagram of the pipelined 2D convolutional kernel is shown below.
    - Input data: `0xA1B2C3D4F0E1D2C3B3`. This data is captured when E is asserted.
    - Output data: `0x2B282`. It appears after the processing delay (6 clock cycles) with v=1.
  - ✓ Complete the timing diagram for the AXI4-Lite Interface: Given the AXI signals for the 5 Channels, complete the signals associated with the Pipelined 2D Convolution Kernel block (`E, v, y, F, FQ` signals) on the next page.

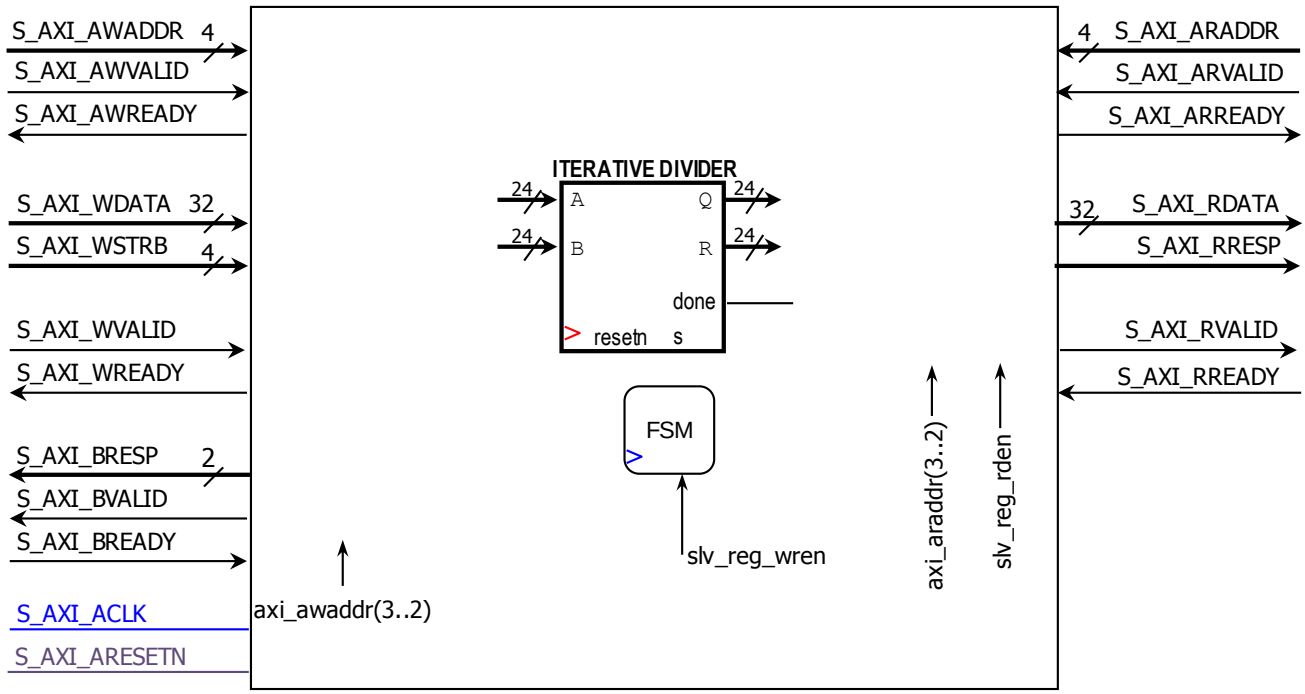





myconv2_ip
FQ
F
v
y
E
S1
sl_reg_rden
READ DATA CHANNEL
RRESP
RREADY
RVALID
RDATA
READ ADDRESS CHANNEL
ARREADY
ARVALID
ARADDR
sl_reg_wren
WRITE RESPONSE CHANNEL
BREADY
BVALID
BRESP
WRITE DATA CHANNEL
WREADY
WVALID
WDATA
WRITE ADDRESS CHANNEL
AWREADY
AWVALID
AWADDR
ACLK
OK
A1B2C3D4
F0E1D2C3
00000B3
C
00000000
00102B82
0
4
8

PROBLEM 2 (35 PTS)

- AXI4-Lite interface for Iterative Divider (N=24, M=24):
  - ✓ Sketch the AXI4-Lite Interface. This includes the Slave Registers, their control signals, as well as the extra glue logic (registers, FSM, etc.) to connect the Iterative Divider to the Slave Register signals.
    - Slave Registers: Use as many as you need, indicating their number. The latched addresses depicted (axi_awaddr[3..2], axi_araddr[3..2]) support up to 4 registers. If for example, you need more registers (say up to 8), you would need axi_awaddr[4..2], axi_araddr[4..2].
    - The start signal s should not be generated via software, rather it should be issued by an FSM once the input data has been received. Sketch the FSM diagram (in ASM form) as well.



PROBLEM 3 (35 PTS)

- Calculate the result of the following operations. The operands are signed (2C) fixed-point numbers. The result must be a signed fixed-point number. For the division, use x=5 fractional bits.

1.101001 + 1.0001	1001.1101 - 1.010101	0.01001 + 01.11011
0.10011 × 10.101	1.011 × 1.0011	01.01110 ÷ 1.011

- Represent these numbers in Fixed Point Arithmetic (signed numbers). Use the FX format [16 8]. (5 pts)
  - ✓ -32.1875
  - ✓ 123.3125