Supplement of Lab 4.1

Author Jiahui HAO

College of Computer Science and Technology

Apr. 13rd, 2021

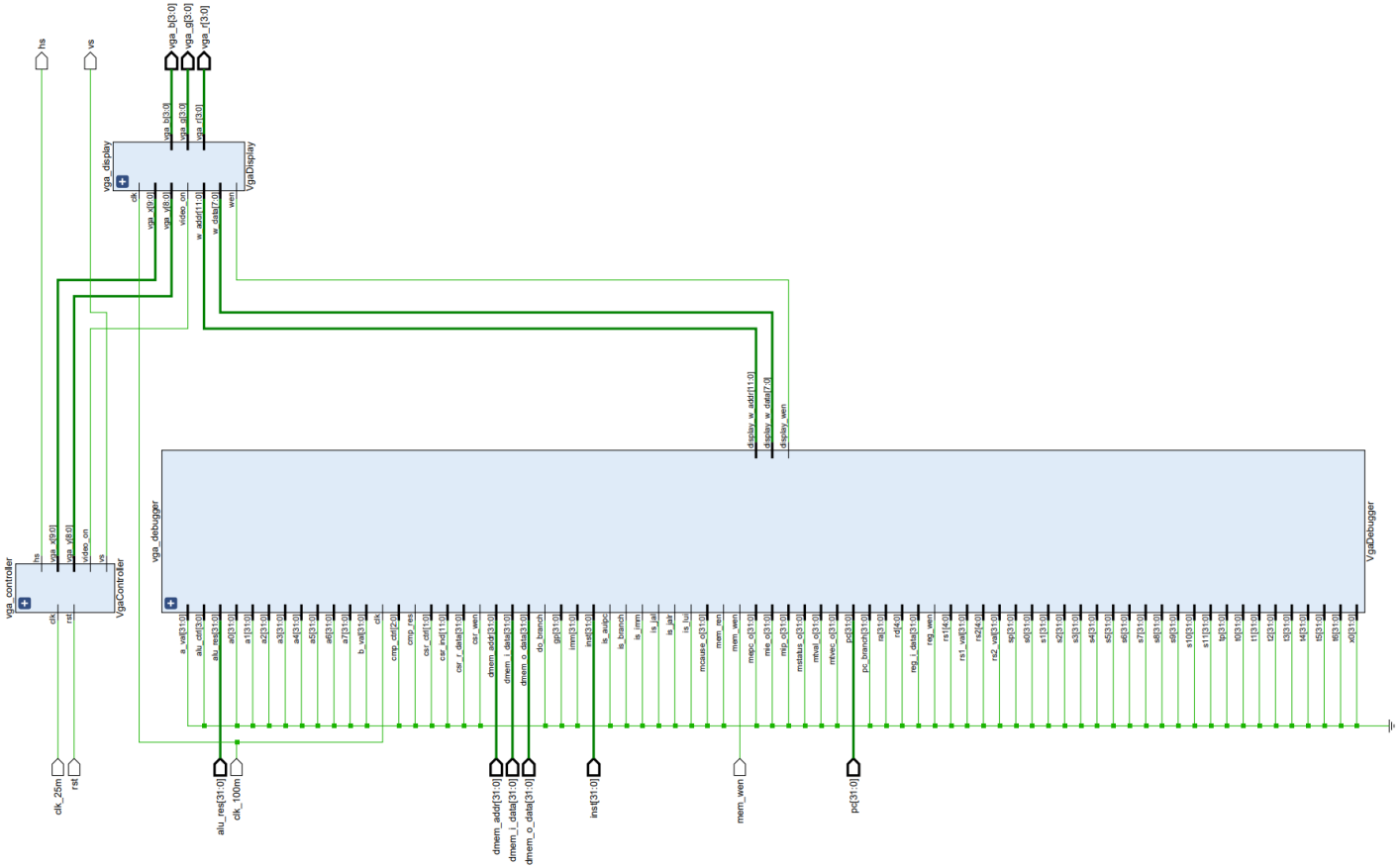
01 VGA Module

Project Location

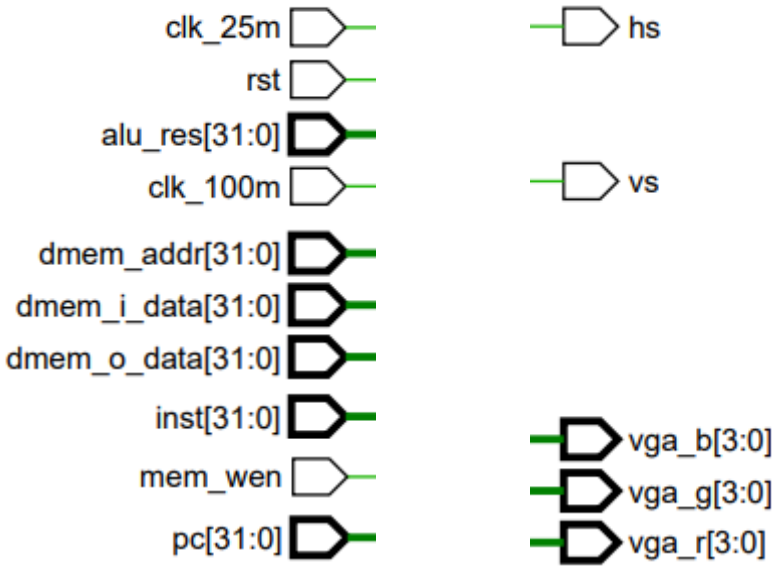
- VGA IP module's project is located at:

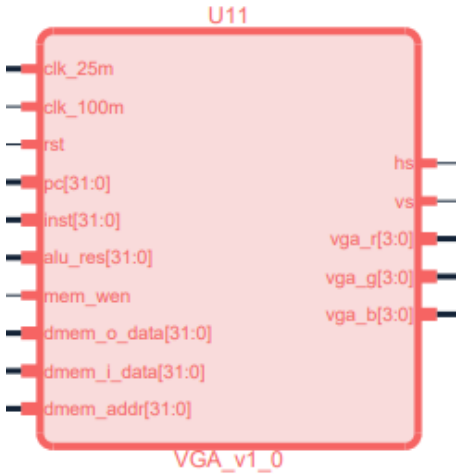
  `OExp04-IP2SOC\IP\Supplementary\VGA\VGA\VGA.xpr`

- *schematic.pdf* can be found at the same location



Current Ports






```
// file: ./VGA.srcs/sources_1/new/VGA.v
// line 23-40
module VGA(
    input wire clk_25m,
    input wire clk_100m,
    input wire rst,
    input wire [31:0] pc,
    input wire [31:0] inst,
    input wire [31:0] alu_res,
    input wire mem_wen,
    input wire [31:0] dmem_o_data,
    input wire [31:0] dmem_i_data,
    input wire [31:0] dmem_addr,

    output wire hs,
    output wire vs,
    output wire [3:0] vga_r,
    output wire [3:0] vga_g,
    output wire [3:0] vga_b
    );
```

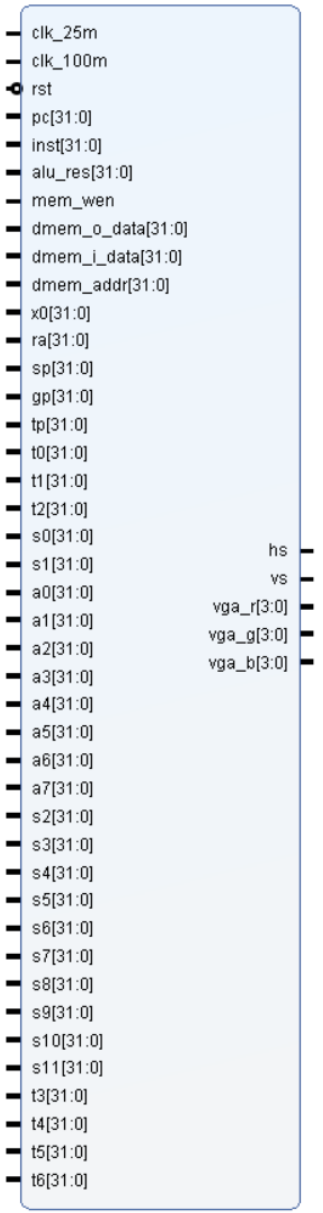
……

Add Register Ports

```
// file: ./VGA.srcs/sources_1/new/VGA.v
// line 106-137
VgaDebugger vga_debugger(
……
.x0(),
.ra(),
.sp(),
.gp(),
.tp(),
.t0(),
……
.s11(),
.t3(),
.t4(),
.t5(),
.t6(),
……
```

- Add register ports at VGA module.
- Use RTL language to connect the additional ports.
- Package IP

  Tool -> Create and Package New IP -> ……
- (Option) Customization GUI



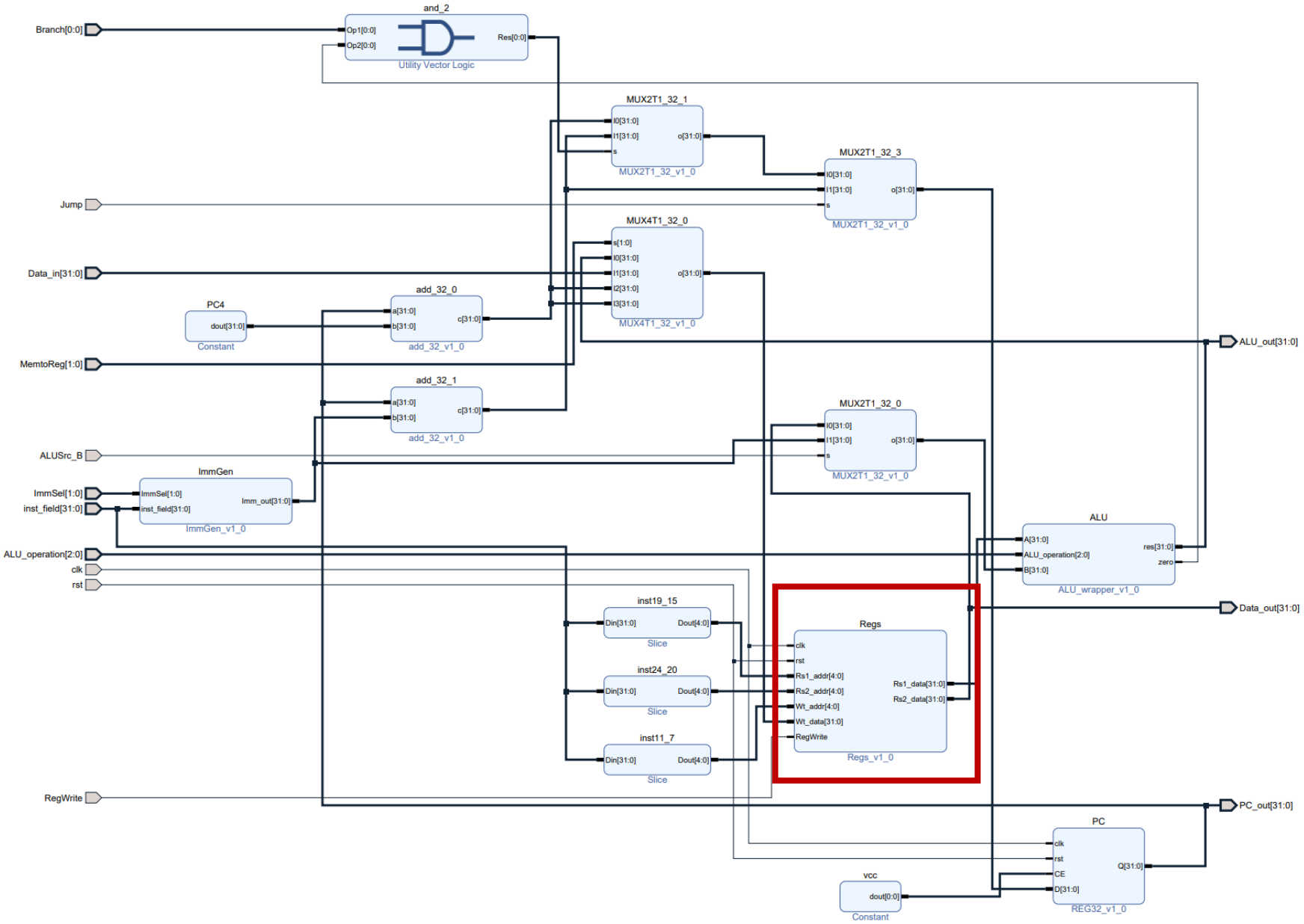
Update Top Module

- Delete symbol in the top module .bd file
- Open IP Catalog
- Remove old VGA IP
- Add new VGA IP
- Insert symbol in the .bd file

02

SCPU Module

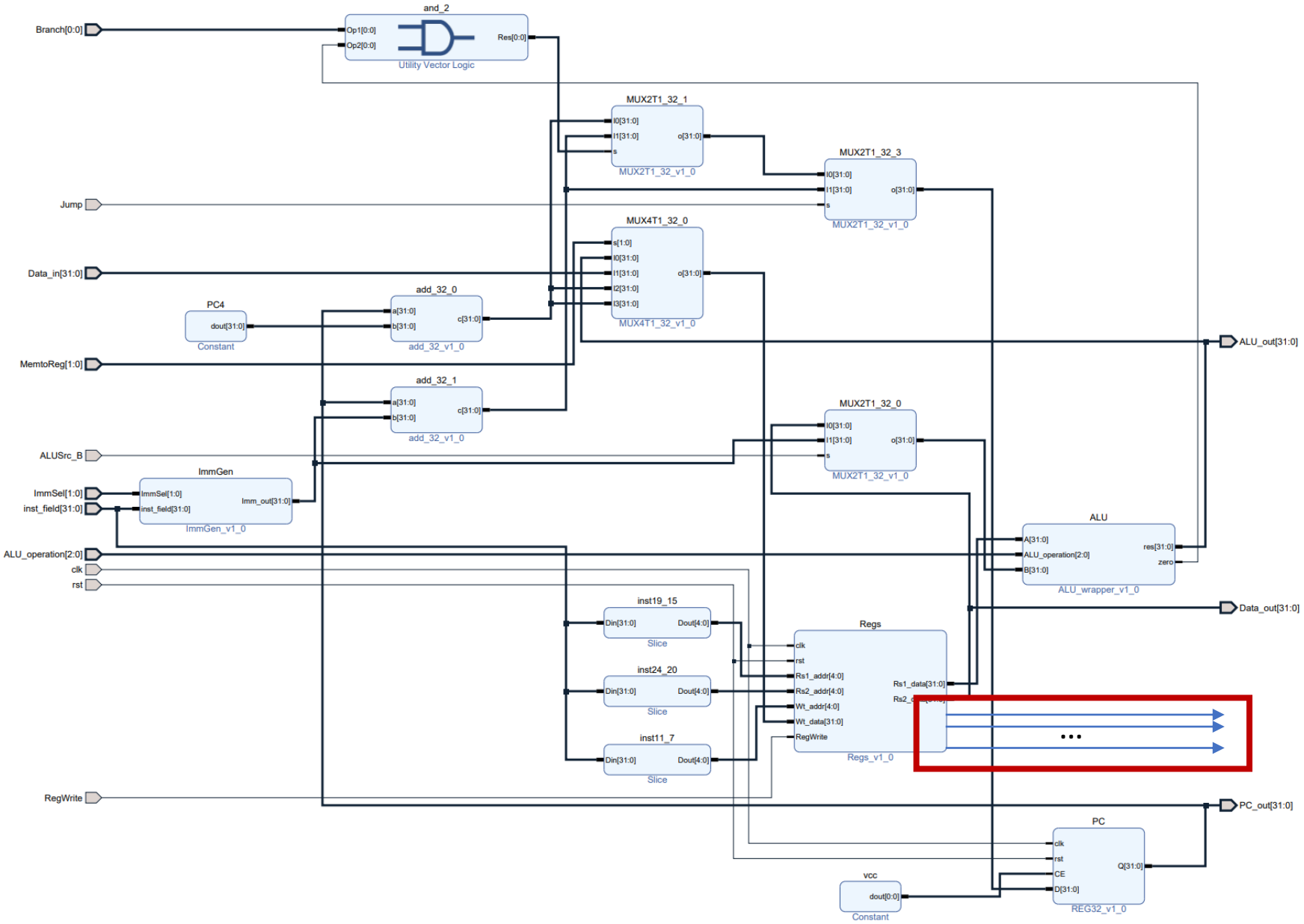
DataPath


and_2
Branch[0:0]
Op1[0:0]
Op2[0:0]
Res[0:0]
Utility Vector Logic
MUX2T1_32_1
I0[31:0]
I1[31:0]
s
o[31:0]
MUX2T1_32_v1_0
MUX2T1_32_3
Jump
MUX4T1_32_0
s[1:0]
I2[31:0]
I3[31:0]
Data_in[31:0]
PC4
dout[31:0]
Constant
add_32_0
a[31:0]
b[31:0]
c[31:0]
add_32_v1_0
ALU_out[31:0]
MemtoReg[1:0]
add_32_1
MUX2T1_32_0
ALUSrc_B
ImmGen
ImmSel[1:0]
inst_field[31:0]
Imm_out[31:0]
ImmGen_v1_0
ALU
A[31:0]
ALU_operation[2:0]
B[31:0]
res[31:0]
zero
ALU_wrapper_v1_0
ALU_operation[2:0]
clk
rst
inst19_15
Din[31:0]
Dout[4:0]
Slice
Regs
inst24_20
inst11_7
Rs1_addr[4:0]
Rs2_addr[4:0]
Wt_addr[4:0]
Wt_data[31:0]
RegWrite
Rs1_data[31:0]
Rs2_data[31:0]
Regs_v1_0
Data_out[31:0]
RegWrite
PC_out[31:0]
PC
CE
D[31:0]
Q[31:0]
REG32_v1_0
vcc
dout[0:0]
Constant

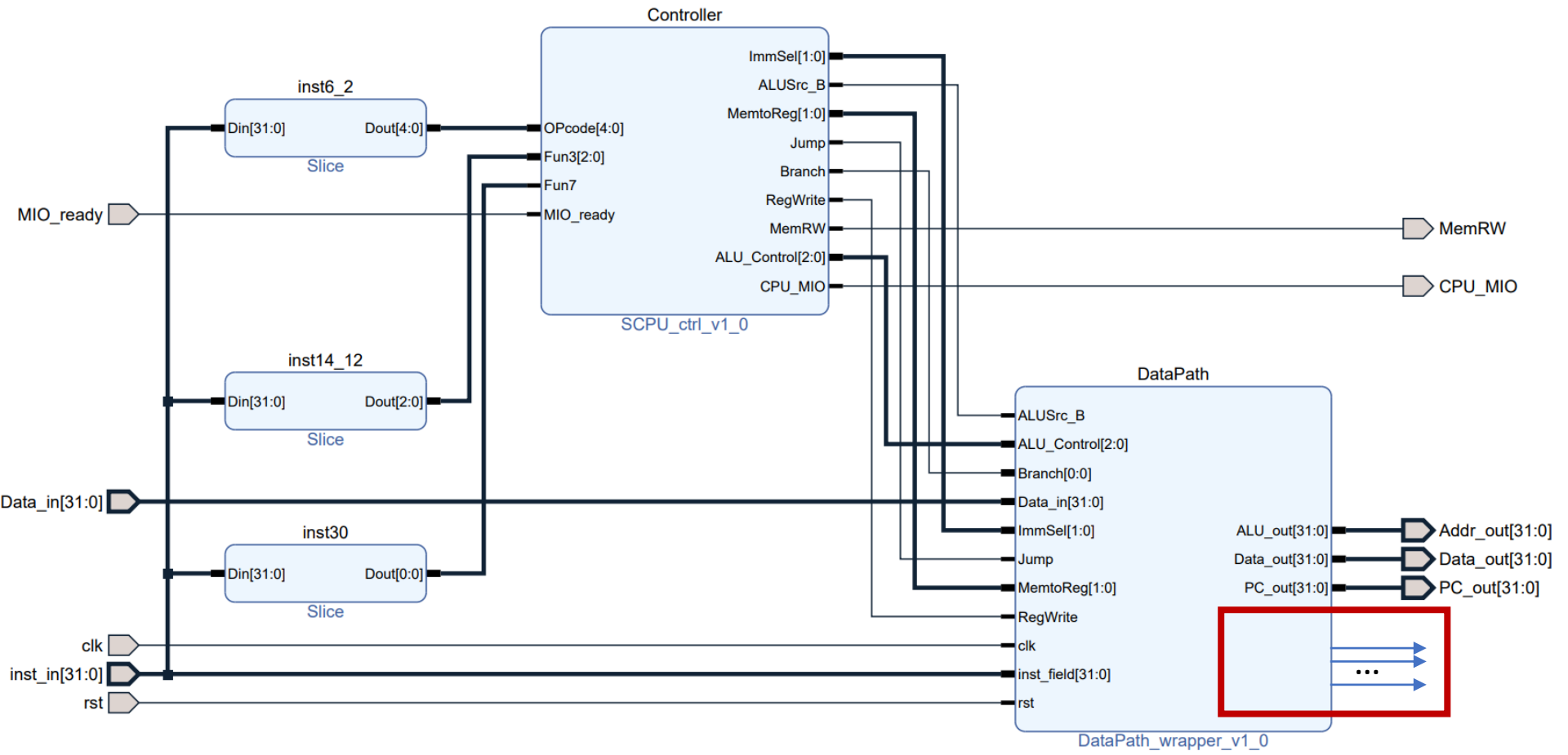
Redesign RegFile Module

- Add 32 output ports for register file
- Package as a new IP
- Update IP in the top module .bd file
- Same as the process for VGA

New DataPath

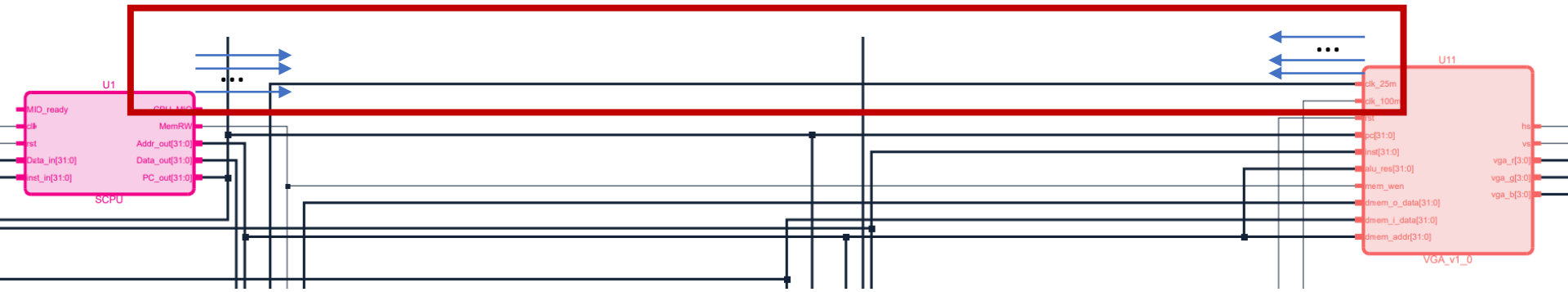


Redesign SCPU



03 Top Module

Redesign Top



04 TA Test Code

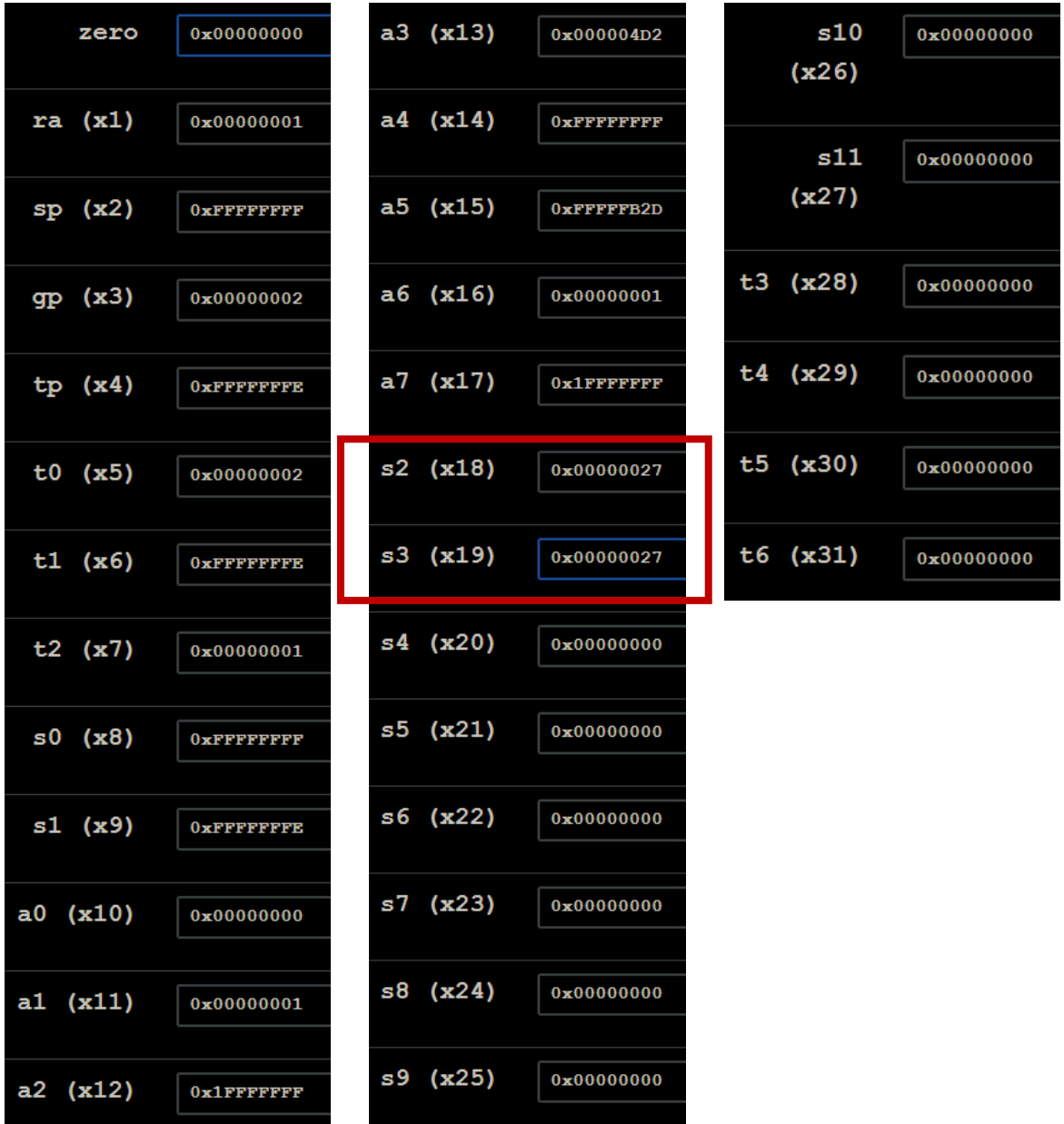
Instruction

```
jal x0, start

dummy:
addi x0, x0, 0
addi x0, x0, 0
addi x0, x0, 0
addi x0, x0, 0
addi x0, x0, 0
jal x0, dummy

start:
addi x1, x0, 1
addi x2, x0, -1
add x3, x1, x1
add x4, x2, x2
sub x5, x1, x2
sub x6, x2, x1
and x7, x2, x1
or x8, x2, x1
xor x9, x2, x1
slt x10, x1, x1
slt x11, x2, x1
srl x12, x2, x1
srl x12, x12, x1
srl x12, x12, x1
andi x13, x2, 1234
ori x14, x2, 1234
xori x15, x2, 1234
slti x16, x2, -5
slti x16, x2, 5
srli x17, x2, 3
lw x18, 0(x1)
addi x18, x0, 0x27
sw x18, 0(x1)
lw x19, 0(x1)
beq x18, x19, dummy
error:
addi x0, x0, 1
addi x1, x0, 1
addi x2, x0, 1
addi x3, x0, 1
addi x4, x0, 1
jal x0, error
```

Change 0x27 to your student ID's last 2 digits

- Use https://venus.cs61c.org/ to get your own I_mem.coe file.
- Initialize ROM with I_mem.coe.
- Contact TA and show me the result.

Result


zero 0x00000000
ra (x1) 0x00000001
sp (x2) 0xFFFFFFFF
gp (x3) 0x00000002
tp (x4) 0xFFFFFFFE
t0 (x5) 0x00000002
t1 (x6) 0xFFFFFFFE
t2 (x7) 0x00000001
s0 (x8) 0xFFFFFFFF
s1 (x9) 0xFFFFFFFE
a0 (x10) 0x00000000
a1 (x11) 0x00000001
a2 (x12) 0x1FFFFFFF
a3 (x13) 0x000004D2
a4 (x14) 0xFFFFFFFF
a5 (x15) 0xFFFFFB2D
a6 (x16) 0x00000001
a7 (x17) 0x1FFFFFFF
s2 (x18) 0x00000027
s3 (x19) 0x00000027
s4 (x20) 0x00000000
s5 (x21) 0x00000000
s6 (x22) 0x00000000
s7 (x23) 0x00000000
s8 (x24) 0x00000000
s9 (x25) 0x00000000
s10 (x26) 0x00000000
s11 (x27) 0x00000000
t3 (x28) 0x00000000
t4 (x29) 0x00000000
t5 (x30) 0x00000000
t6 (x31) 0x00000000

Other Signals

`rs1 rs1_val`	Source register 1's address and value
`rs2 rs2_val`	Source register 2's address and value
`rd`	Destination register's address
`rd_i_data`	Input data for destination register
`reg_wen`	Write enable for register file
`is_imm`	Is I-type instruction
`is_auipc` `is_lui`	Is AUIPC instruction Is LUI instruction
`imm`	Immediate number
`a_val b_val`	Two sources number of ALU
`alu_ctrl`	ALU mode
`cmp_ctrl`	?compare mode?
`alu_res`	ALU computing result
`cmp_res`	Compare result
`mem_wen mem_ren`	RAM write enable and read enable