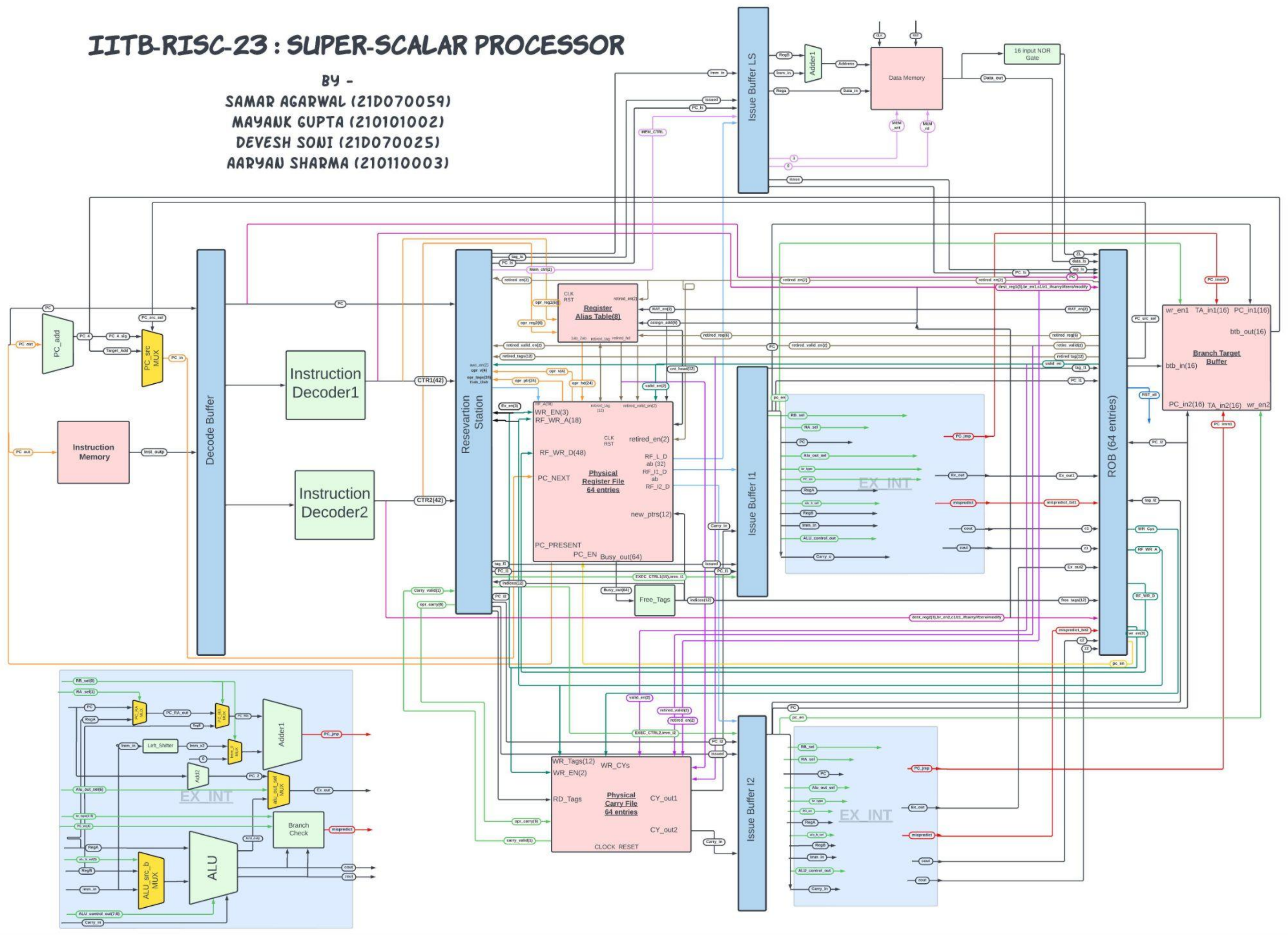


EE-309 Project Report

IIT-BOMBAY-RISC-SUPERSCALER-23

PIPELINED IMPLEMENTATION

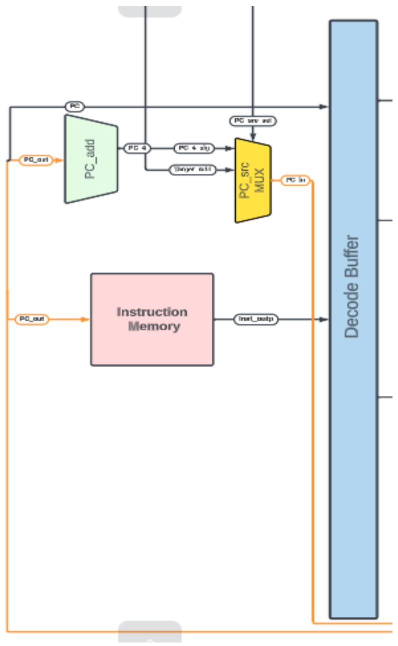
DESIGN-REPORT

EE309 | 04/12/2023

Samar Agarwal	21d070059
Aaryan Sharma	210110003
Mayank Gupta	210101002
Devesh Soni	21d070025

Data Path Stages

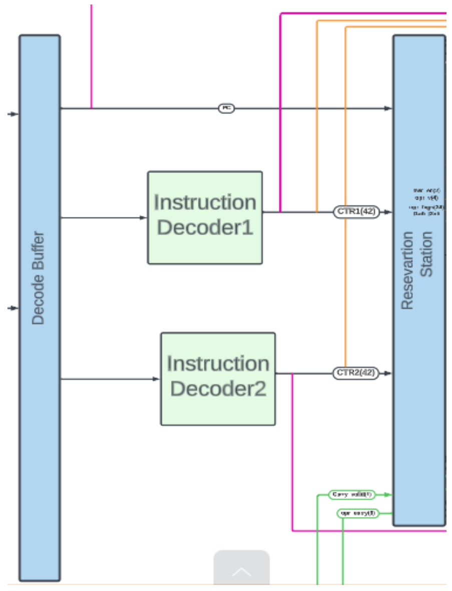
Instruction Fetch



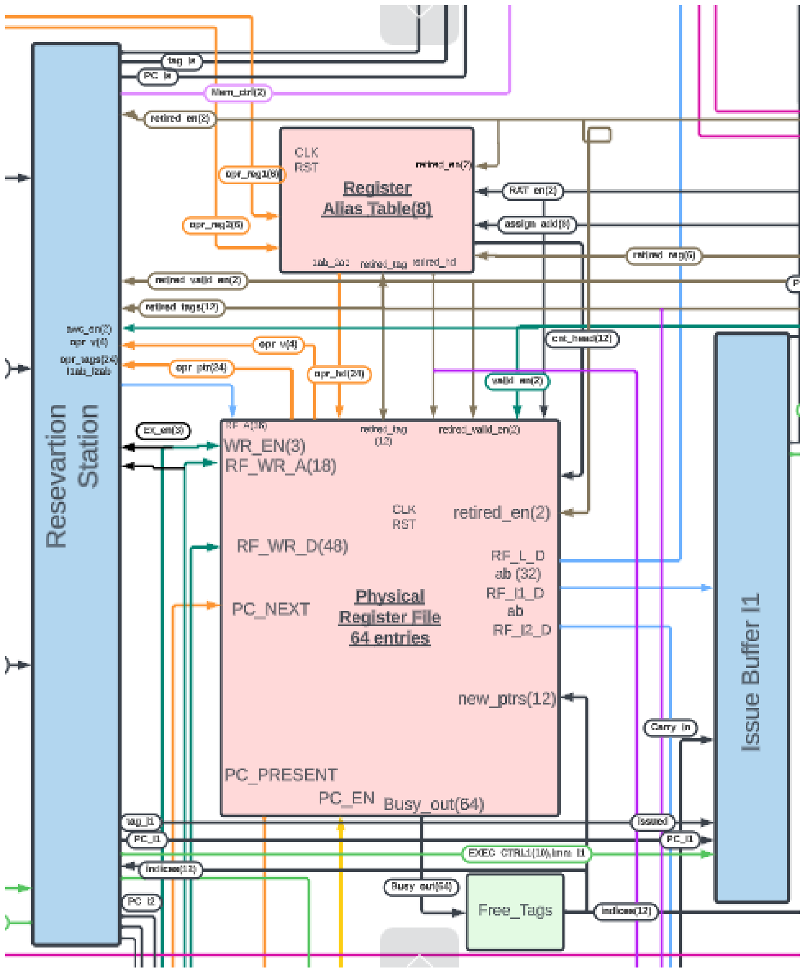
- ○ Program Counter: Pointing to the instruction in the instruction memory to fetch it stored in R0 of the register file
- ○ PS: orange lines indicate to and fro from pc.
- ○ Instruction Memory: Storing all the two-byte instructions written by the programmer. Byte addressable , total size = $2^{16} \times 2^{3} = 64MB$
- ○ PC_src_MUX(2x1): Deciding whether to fetch the next instruction according to pc or jump to another instruction based on the control signal(Pc_src) coming from the branch Identifier.
- ○ PC_add: It increments the PC value by 4(2-way fetch each instruction is 16-bit) for the next instruction.
- ○ It stores bits for PC , and two fetched instructions . Thus , it stores total 48 bits .

Instruction Decode

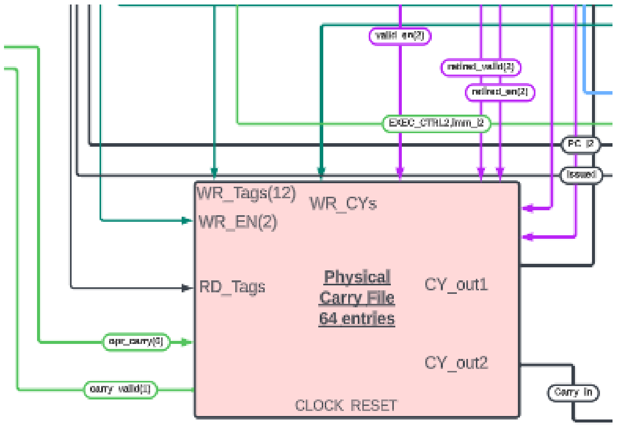
- Instruction Decoder (sign-extender used inside it): Decodes the instruction into control bits for all the stages for that particular instruction. We have used 2 instruction decoders (one for each instruction).
- Sign Extender: It extends the 6-bit or 9-bit immediate data to 16-bit.
- Control_Bits: Instruction decoders pass control bits (42) for each instruction to the reservation station.



Register Read



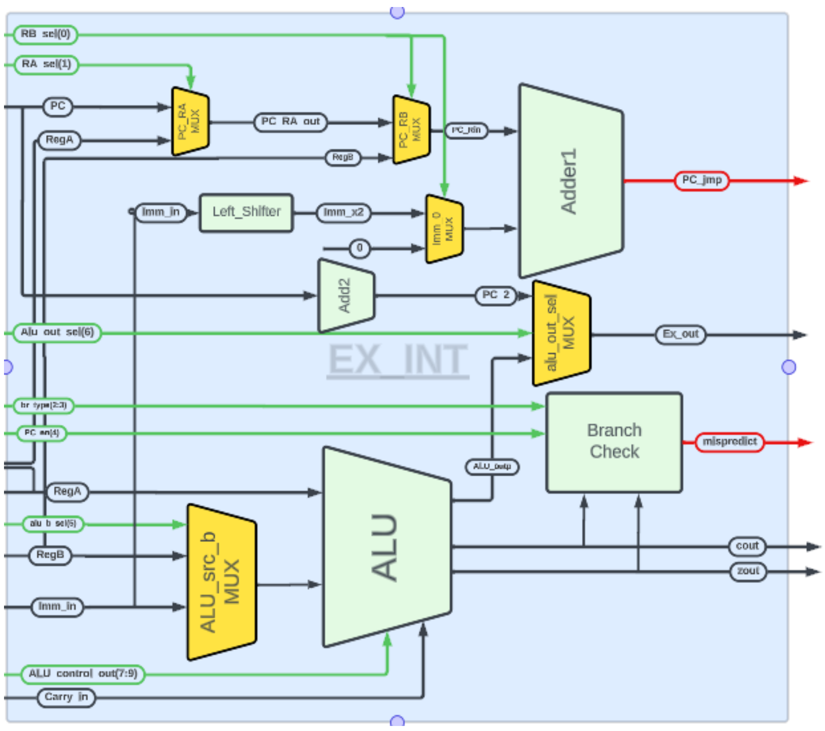
- Physical Register File: Stores registers for ARF and RRF (size = 64). Each register is represented by three quantities:
  - Busy bit - Register is filled or empty
  - Pointer - points to most updated value as per dispatch
  - Valid - register contains garbage value or most updated value
- Register Alias Table (RAT): stores header for each ARF register and updates header at time of dispatch (size = 64). The initial 8 entries store header bits for 8 ARF registers.
- Free Tags: Priority encoder returns 2 indices for 2 empty entries of PRF.
- Physical Carry File (PCF): resolves carry dependencies for instructions which modify or use carry by maintaining a pointer, similar to Rename Registers. Carry generated is stored in the carry registers with the same tags as that of destination tags of instructions



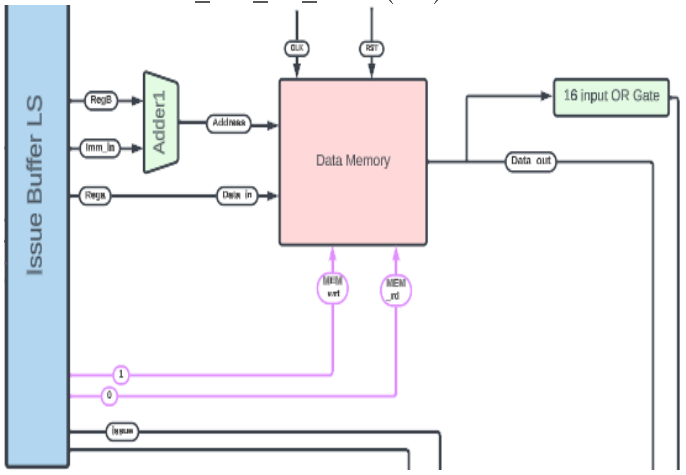
- Color Coding:
  - Dark Green: execution
  - Black/Orange/Light Green: Dispatch
  - Blue: Data reading for scheduled instructions
  - Brown: Completion /Retiring

Execution Stage

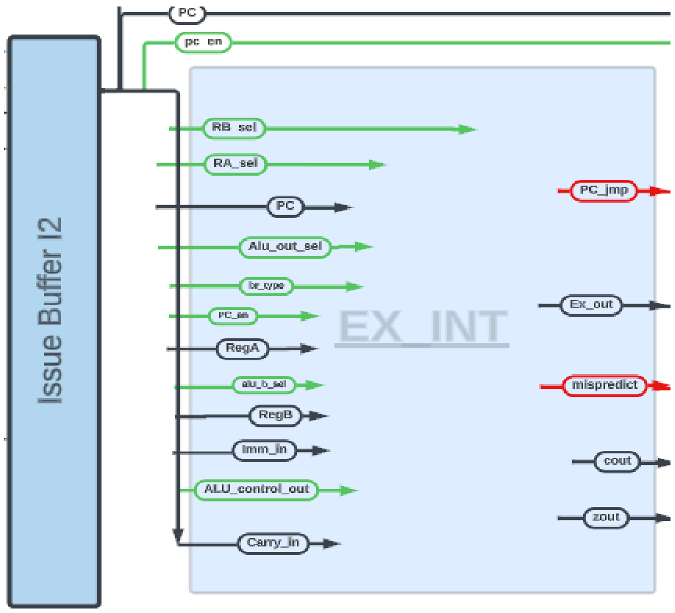
- ALU: Perform ADD, NAND, ADD_CARRY, NAND_CARRY, ADD_COMP, ADD_CarryComplement and SUB operations.
- Left_Shifter: Left shifts the sign-extended immediate value by two, giving us Imm*2.
- Adder1: Used to calculate the Jump Address.
- ALU_src_b_MUX(2x1): to select the input b of ALU between Register B, its complement, and extended immediate Value.
- PC_RA_MUX(2x1): to select the input between regA and PC for the input of PC_RB_MUX (Jump Address)
- PC_RB_MUX(2x1): to select the input between PC_RA_Mux output and RegB for Adder1 input A.



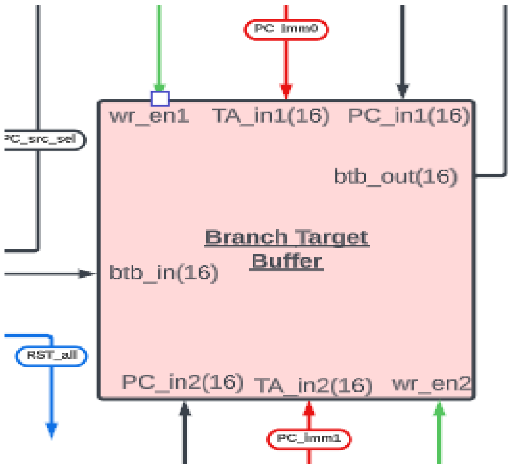
- Imm_0_MUX(2x1): to select between Imm_x2 and 0 for Adder1 input B.
- Brancher: Deals with the instructions with conditional or unconditional branches and is governed by Cflag, Zflag,br_type, and pc_en controls. Br_type tells the type of branch instruction and pc_en is zero if the instruction is not jump type. It detects the branch hazards, introduces bubbles and updates the pc to pc_jmp
- Z_flag: Flip flop used to store the z flag. | C_flag: Flip flop used to store C flag.
- alu_out_sel_MUX(2x1): Selects either we want ALU_output or PC+2



- For superscalar, we have three execution pipelines:
    - Load Store pipeline :
        - Issue_Buffer_LS: for storing issued load-store instructions and their PC values.
        - Adder1: Simply adds RegB values with sign extended Immediate bits
        - Data Memory: Stores data value for each instruction
        - 16 input NOR Gate: Kind of Zero flag, returns 1 if all bits are zero



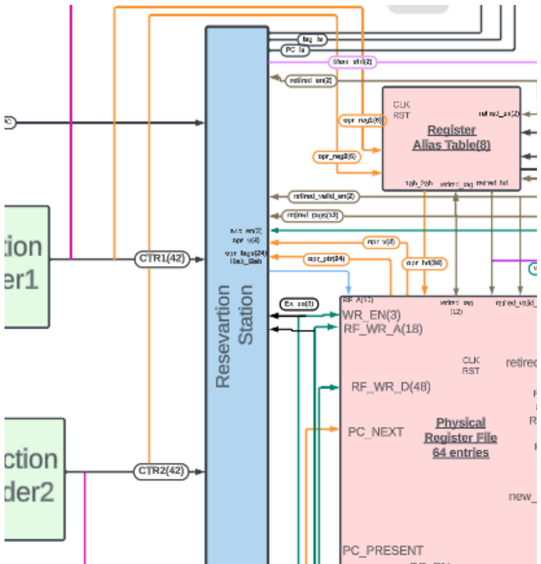
- Integer pipeline :
  - Issue_Buffer_Integer: Stores control bits, integer instructions, and their PC values
  - There are two such pipelines and it gives various parts of control bits as output



Branch Target Buffer (BTB)

- 16 entry buffer, which stores branches with their target addresses.
- It is updated on execution and used by ROB when misspeculation is predicted .
- Uses 16 bit comparator to compare PC values.

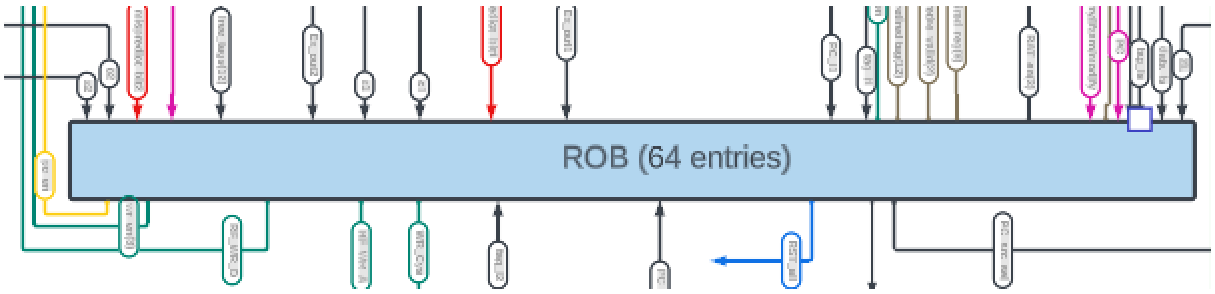
Reservation Station



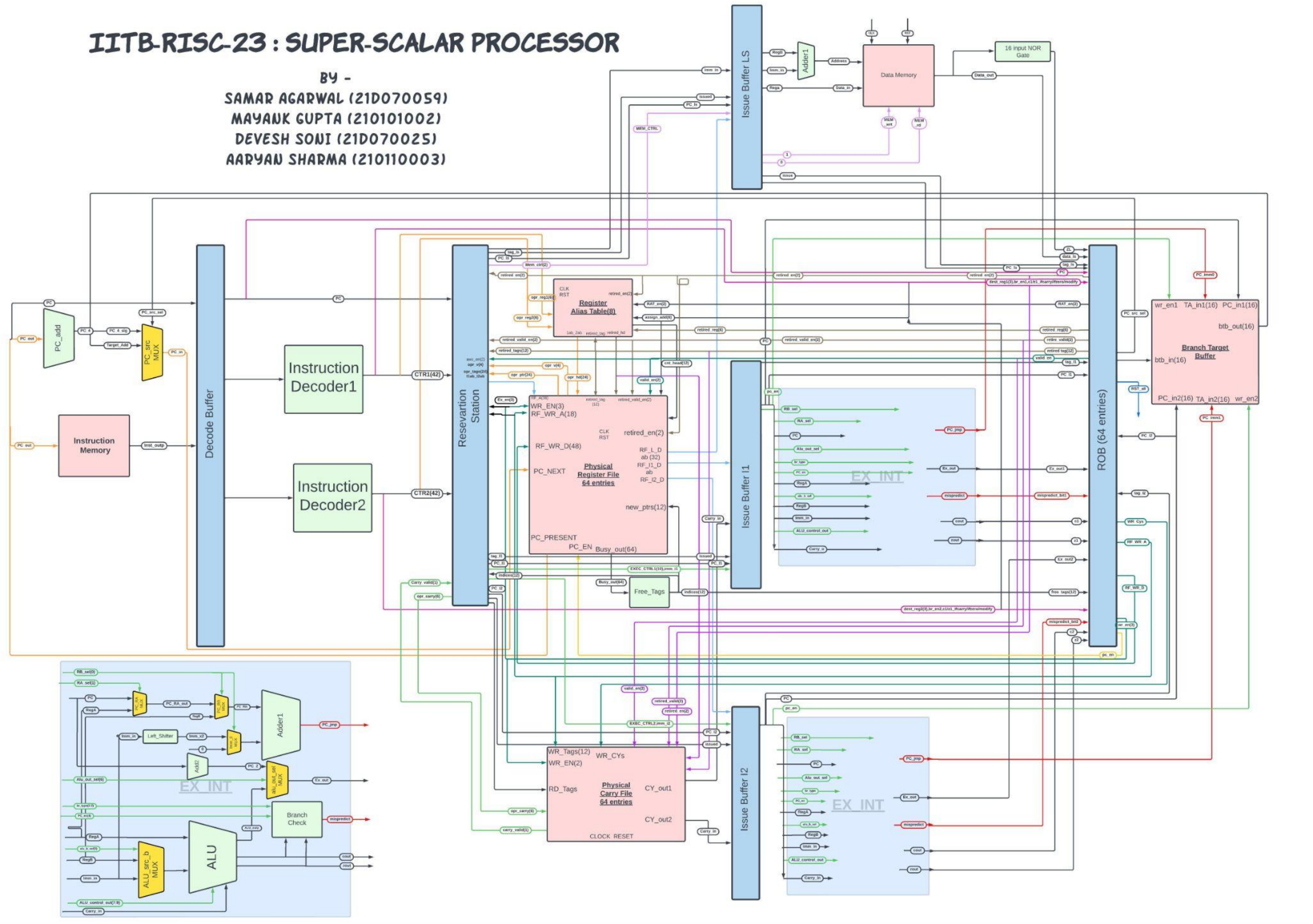
- Reservation Station with 64 entries
- Total Load Store Order Buffer inside Reservation Station with 16 entries
- Dispatch- Adding two entries from ID stage to the Reservation Buffer
- Scheduling- Scheduling Instructions to the 3 pipelines based on their ready bits
- Execution and Completion - Waking Up (Valid=1)
- Without Data Capture
- Has Busy,PC,Opr1,Opr2,V1,V2,Vc,Control_Bits

Reorder Buffer (ROB)

- The reorder buffer primarily takes care of 3 sub-processes:
  - Dispatch
  - Execute
  - Completion (Retire)
- In dispatch, an instruction is entered into the ROB. The register to which the data is supposed to be stored is entered in the ROB along with its PC and tags. A few control bits are also stored for the branch and if carry set instructions.
- The ROB here does the work of the bookkeeper.
- After execution, the data is stored in the tag stored in the ROB in the PRF. The valid bit is made 1 after storing the data for instructions that don't have the if carry condition. For the if carry cases, the valid bit is made 1 if carry is 1 during retirement.
- In retirement, the instructions are retired in order, 2 at a time. The head keeps track of the executed instructions while the tail keeps track of the instructions to be added.
- Also, the busy bit is made 0 in the PRF, and the head is changed in RAT.
- Waking up is done at execution for non-if-carry set instructions, while for if-carry set instructions waking up is done at completion.



DATAPATH


IITB-RISC-23 : SUPER-SCALAR PROCESSOR
BY -
SAMAR AGARWAL (21D070059)
MAYANK GUPTA (210101002)
DEVESH SONI (21D070025)
AARYAN SHARMA (210110003)
Issue Buffer LS
Adder1
Data Memory
16 input NOR Gate
Decode Buffer
Instruction Memory
PC_add
PC_src MUX
Instruction Decoder1
Instruction Decoder2
Resevartion Station
Register Alias Table(8)
Physical Register File 64 entries
Free_Tags
Issue Buffer I1
EX_INT
ROB (64 entries)
Branch Target Buffer
Physical Carry File 64 entries
Issue Buffer I2
ALU
ALU_src_b MUX
Branch Check
Left_Shifter