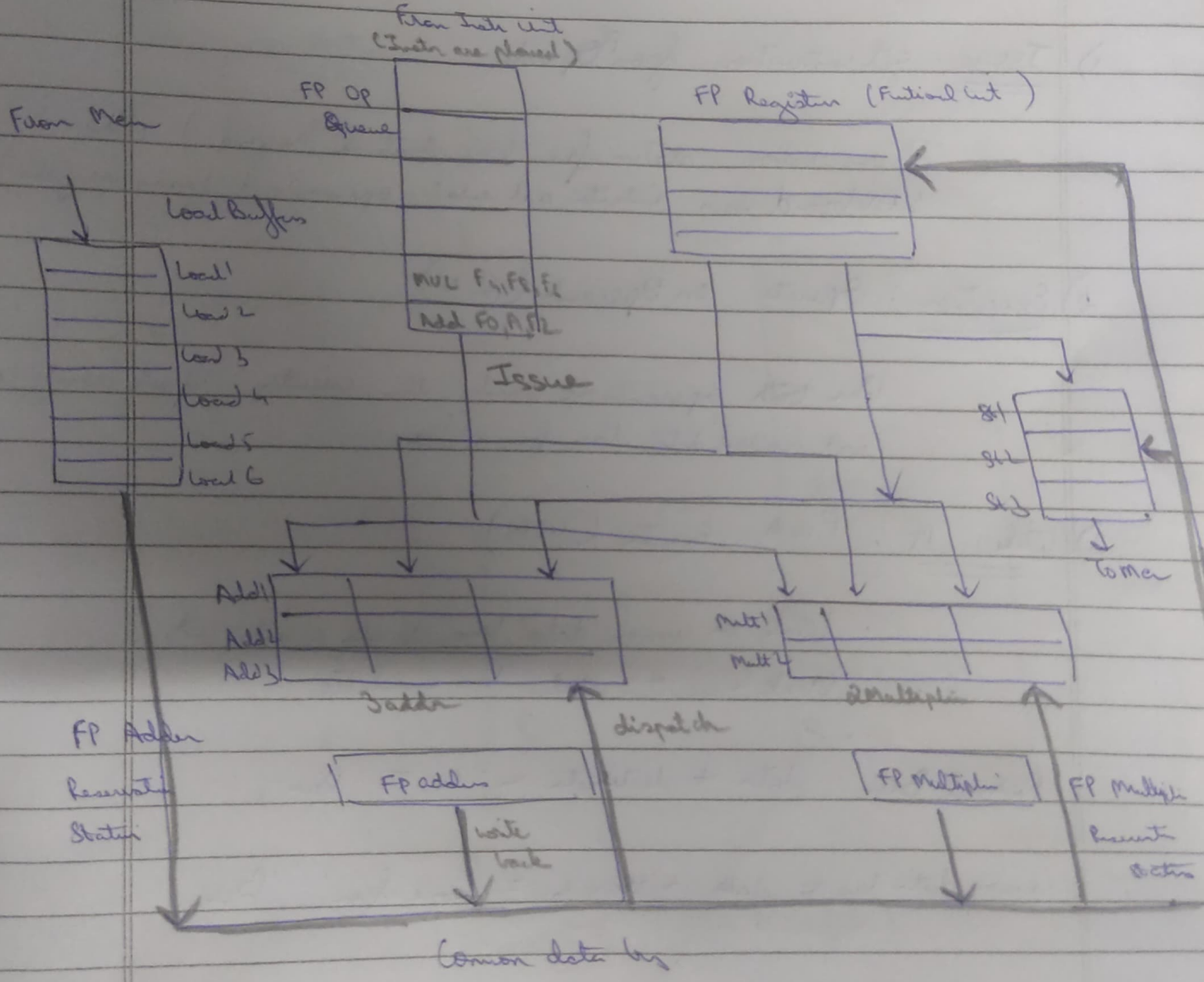
Tomasulo's Algorithm

→ dynamic scheduling technique
→ It removes WAR dependency through register renaming
→ It tracks when operands are available to satisfy the data dependency
→ Hardware approach



→ Reservation Station Components:

- Op – operation to perform in the unit
- Qj, Qk – Reservation stations producing source registers (value to be written)
- Vj, Vk – value of source operands

- R_j, R_k - Flags indicating when V_j, V_k are ready
- Busy - Indicates reservation station or FU is busy

Three Stages of Tomasulo's Algo :-

1) Issue : get instruction from FP Op Queue

If reservation station free (no structural hazard), the scoreboard issues instruction and sends operands (renames registers)

2) Execution : Operate on Operands (EX)

When both operands are ready, then execute; if not ready, then watch Common Data Bus for result

3) Write result : Finish Execution (WB)

- Write on Common Data Bus to all awaiting units; Mark reservation station available

Normal Bus : data + destination = "Go To" Bus;

Common Data Bus = data + source = "Come from" Bus;

Example: Steps of Execution using Tomasulo's Algo

3 loads, 3 adders, 2 multiplies. The following instructions are executed in a pipelined way.

LD F6, 34(R2)
LD F2, 45(R3)
MUL F0, F2, F4
SUB F8, F6, F2
DIV F10, F0, F6
ADD F6, F8, F2

Load - 2 cycles
Add - 2 cycles
MUL - 10 cycles
Div - 40 cycles

Instruction Status

Instruction Status	Issue	Execute	Write Back
LD F6, 34(R2)	1	2-3	4
LD F2, 45(R3)	2	3-4	5
MULT F0, F2, F4	3	6-15 (10 cycles)	16
SUB F8, F6, F2	4	6-7	8
DIV F10, F0, F6	5	17-56 (40 cycles)	57
ADD F6, F8, F2	6	9-10	11

Write Backs the result

	Busy	Address
Load 1	~~Yes~~	~~34+R2~~ (At 4 cycle)
Load 2	~~Yes~~	~~45+R3~~ (At 5th cycle)
Load 3		

ACHIEVER
Date: / /

Reservation Station Status

TIME	NAME	BUSY	OP	VJ	VK	QJ	QK
	ADD1	~~Yes~~	~~SUB~~	~~M(A1)~~	~~M(A2)~~		~~Load2~~
	~~ADD2~~						
	ADD2	~~Yes~~	~~ADD~~	~~M(A1)~~	M(A1)	~~ADD1~~	
	ADD3						
	MULT1	~~Yes~~	~~MULT~~	~~M(A1)~~	~~R(R3)~~	~~Load2~~	
	MULT2	~~Yes~~	~~DIV~~	~~M(A1)x R~~	~~M(A1)~~	~~MULT1~~	

Register Result Status

Clock	1	2	3	4	5	6	7	8	9	10	11	...	55
F0		F2		F4		F6		F8		F10		F30	
FU	MULT1	Load2				Load1		ADD1		MULT2			
		M(A2)				M(A1)		(M1)-(M2)					
						Add2							
	M(A2)xR					(M1-M2)+M(A2)				M1-M2 / M(A2)			