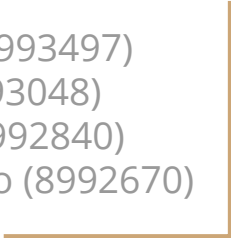




# Algoritmo de Tomasulo

Organização e Arquitetura de Computadores II



Allan Almeida Santos (8993497)  
Pedro Donini Linan (8993048)  
Pedro Duarte Bairao (8992840)  
Henrique Costabile Filho (8992670)

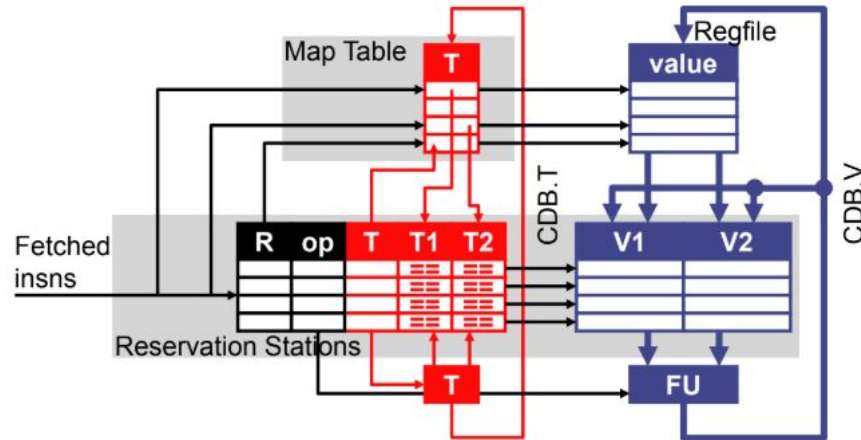
# Introdução

- **Escalonamento Dinâmico**

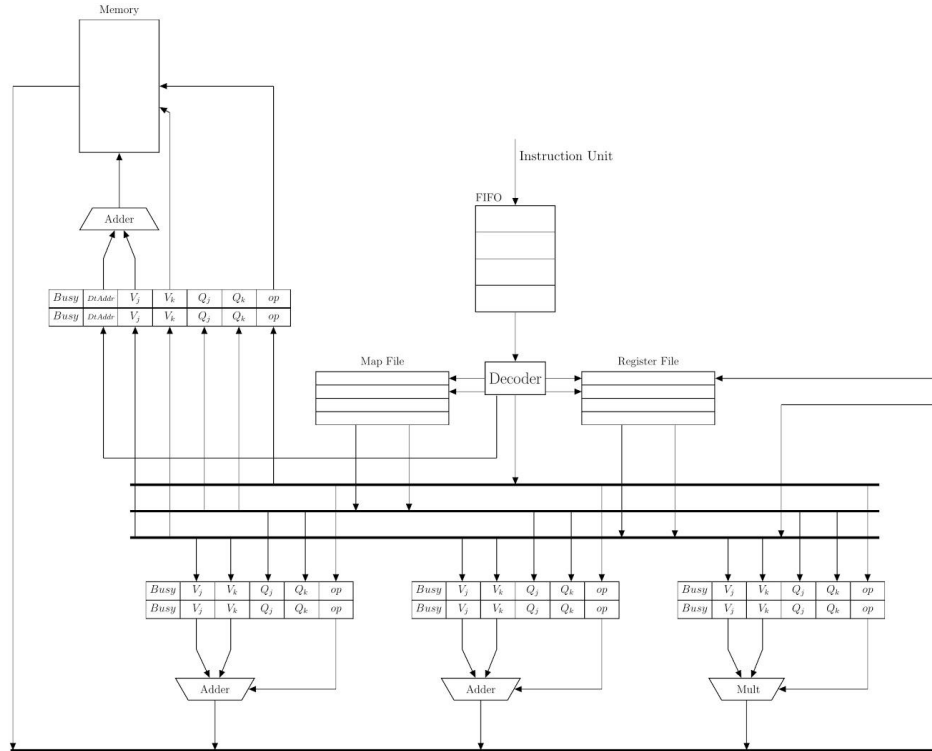
- Reservation stations (RS): Buffer de Instruções
- Common data bus (CDB): Transmissão dos resultados direto às RS

- **Renomeação de Registradores (Tags)**

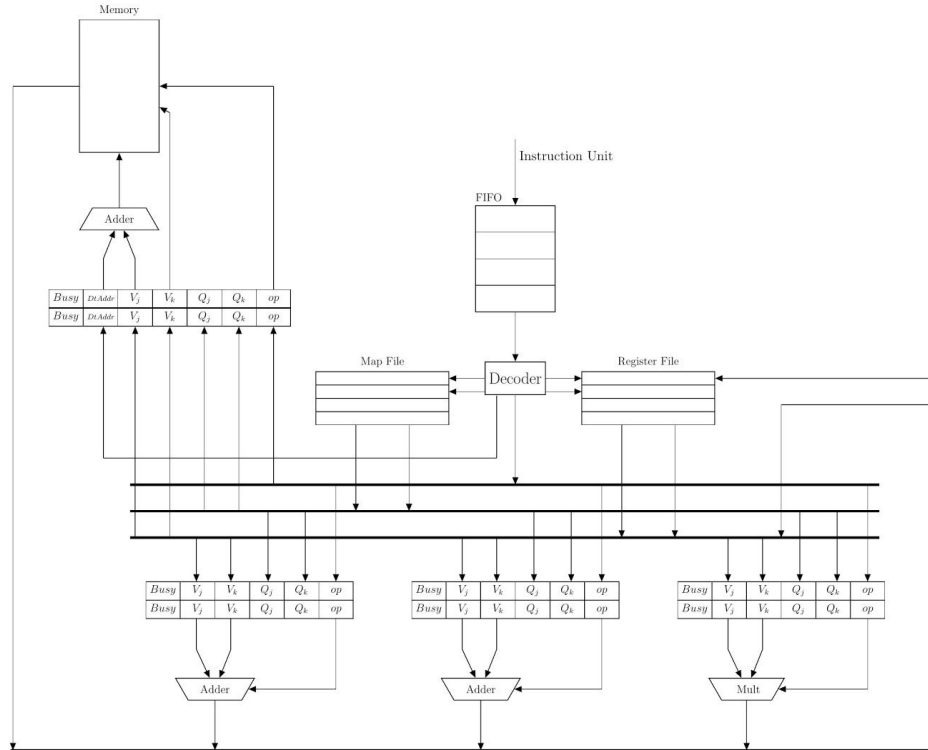
- Resolve conflitos de dados (WAW/WAR)



# Arquitetura do Circuito Tomasulo

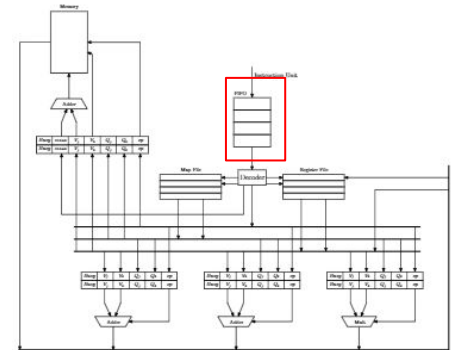
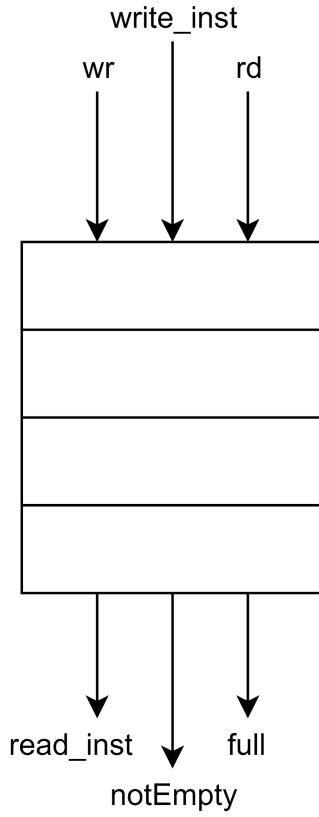


# Arquitetura do Circuito Tomasulo

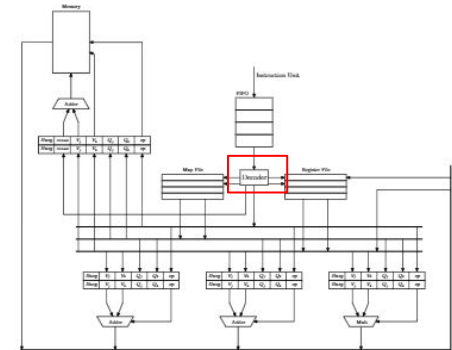
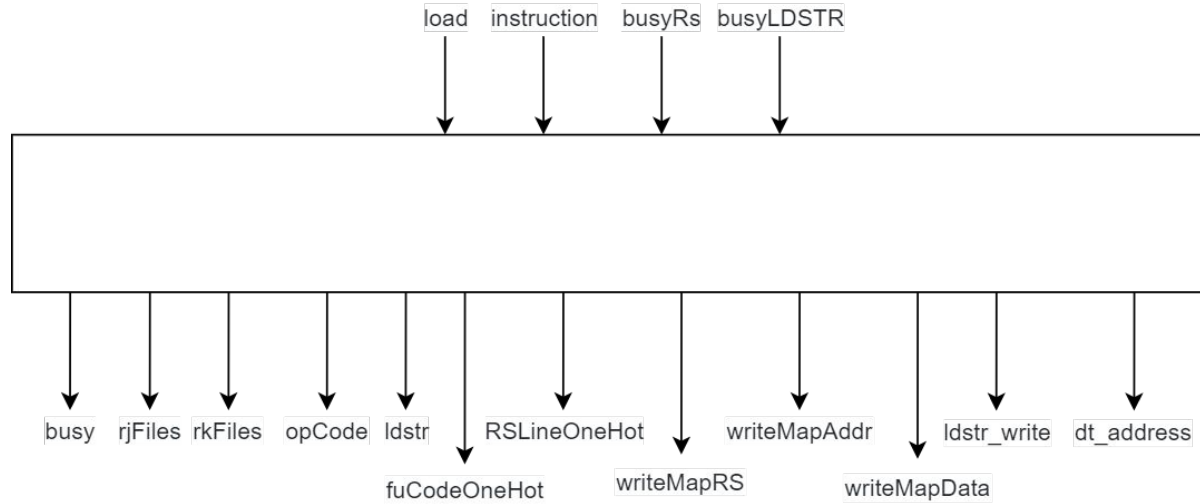


- FIFO
- Decoder
- Register File
- Map Table
- Reservation Station (unidades funcionais)
- Unidades Funcionais
- Reservation Station (memória)
- Adder
- Memória
- Barramento de Dados Comum (CDB)

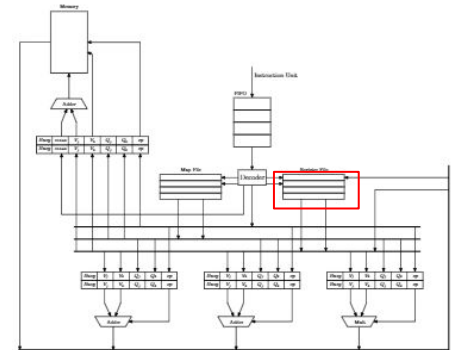
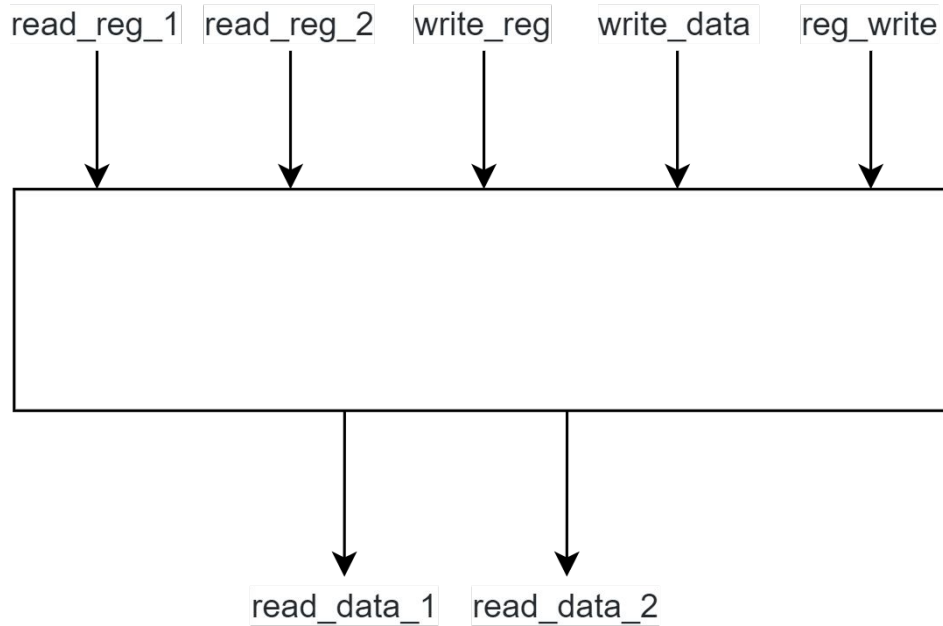
# FIFO



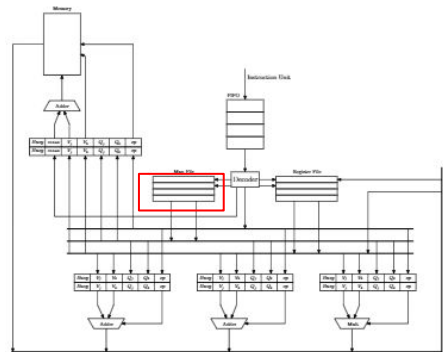
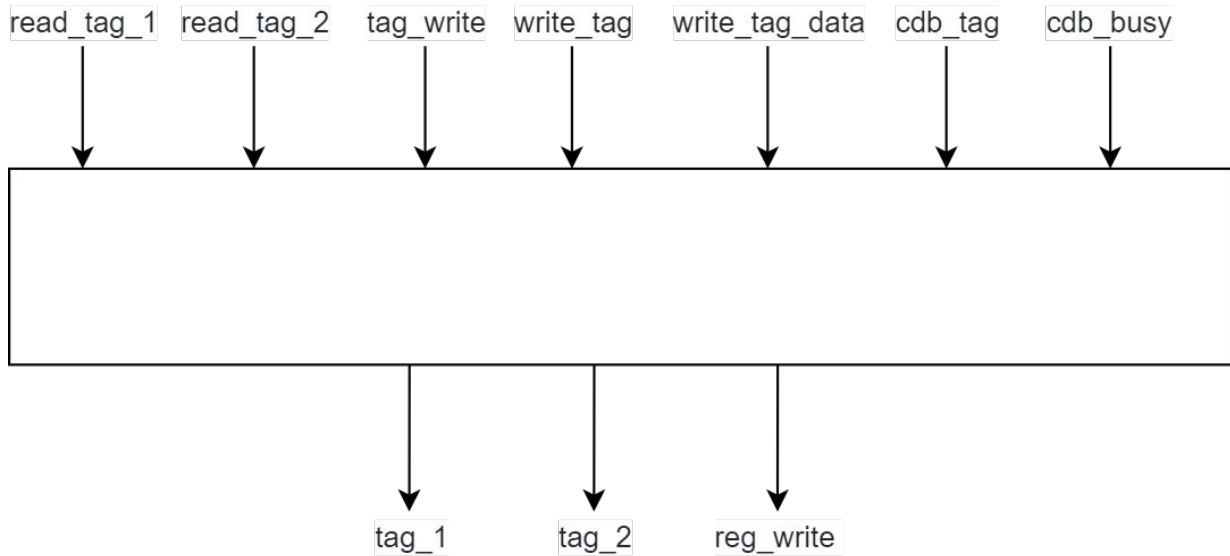
# Decoder



# Register File

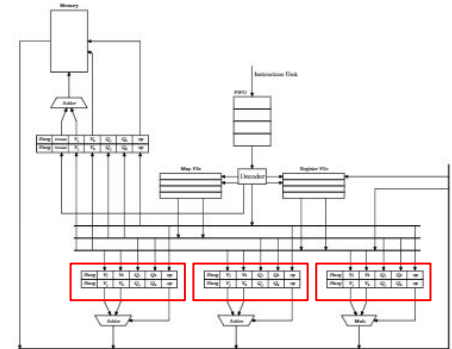
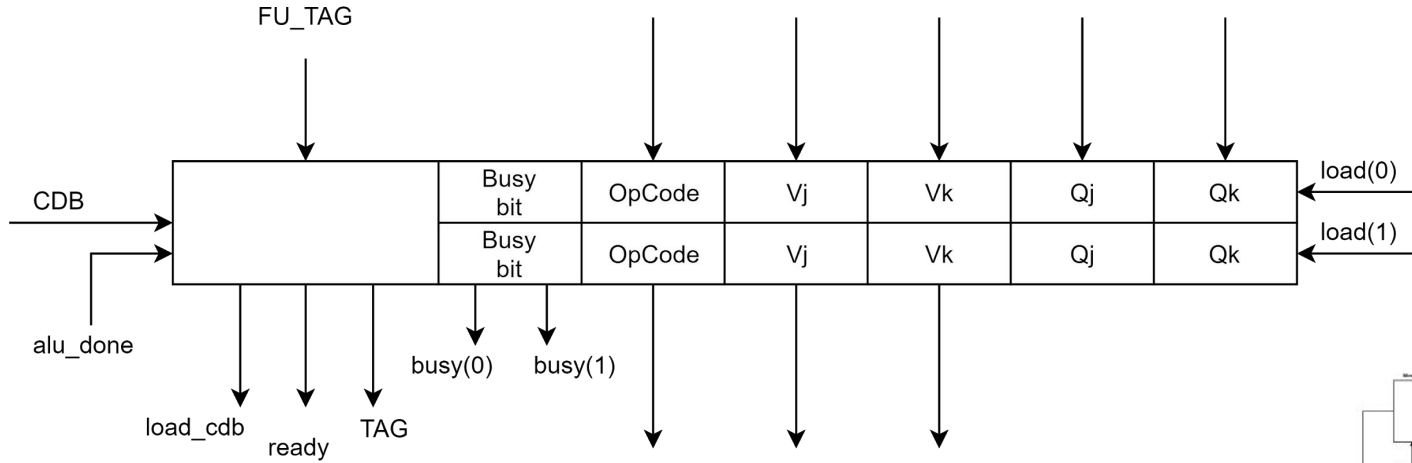


# Map Table

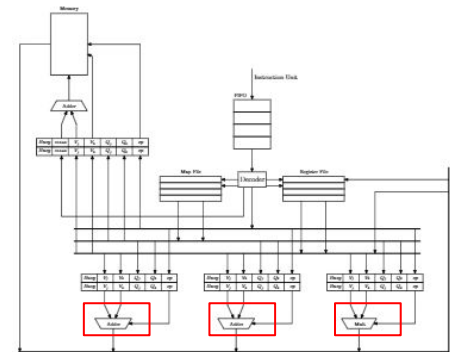
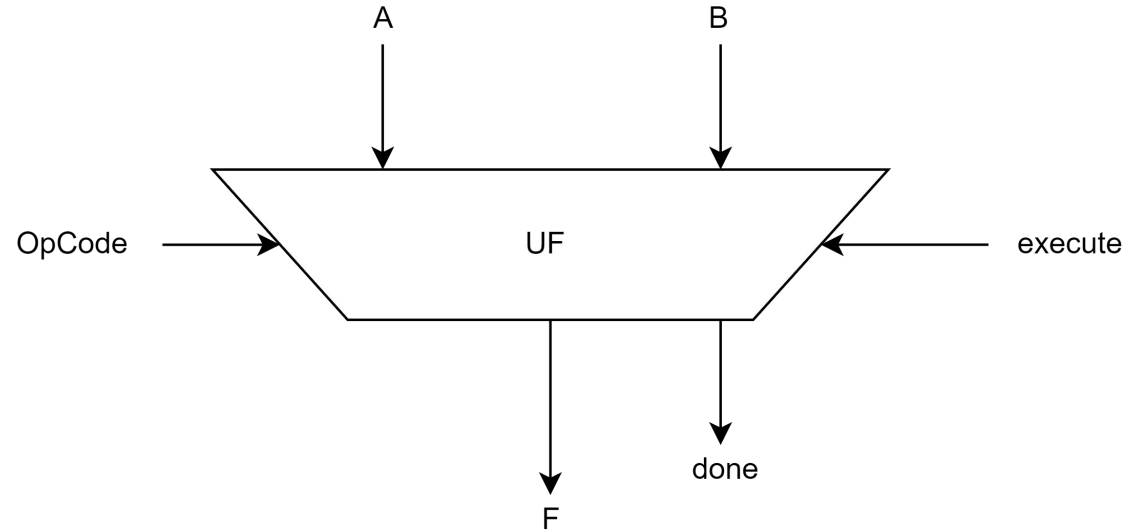




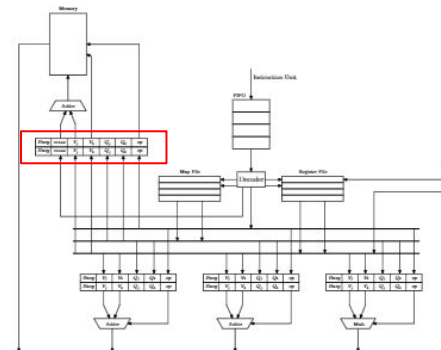
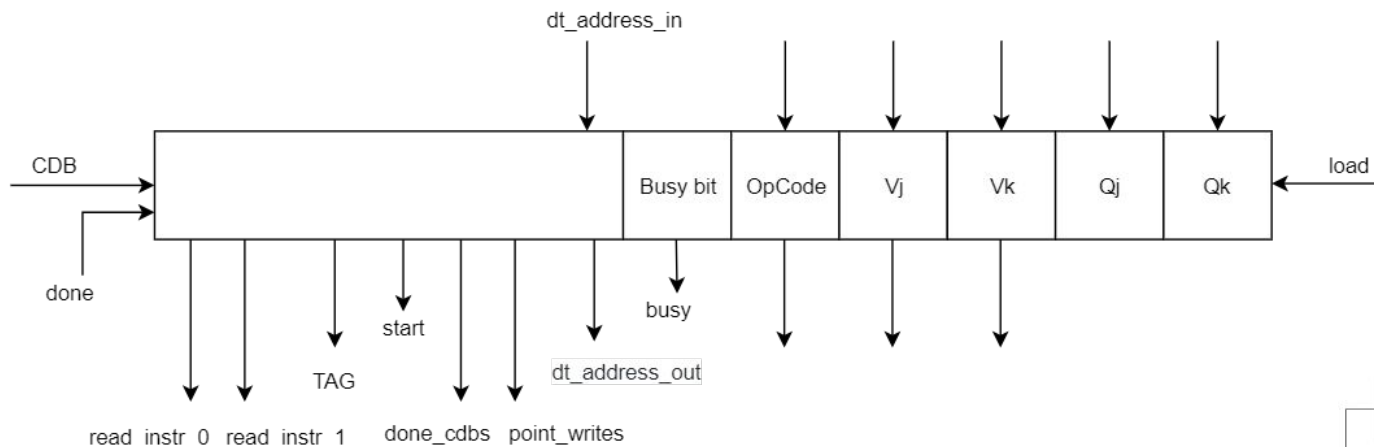
# Reservation Stations (UF)



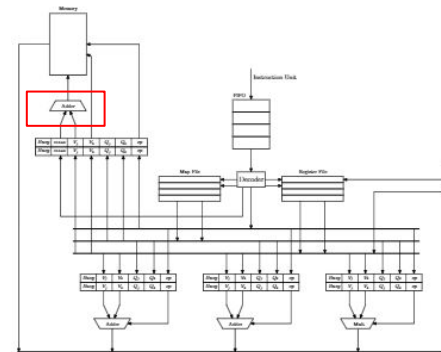
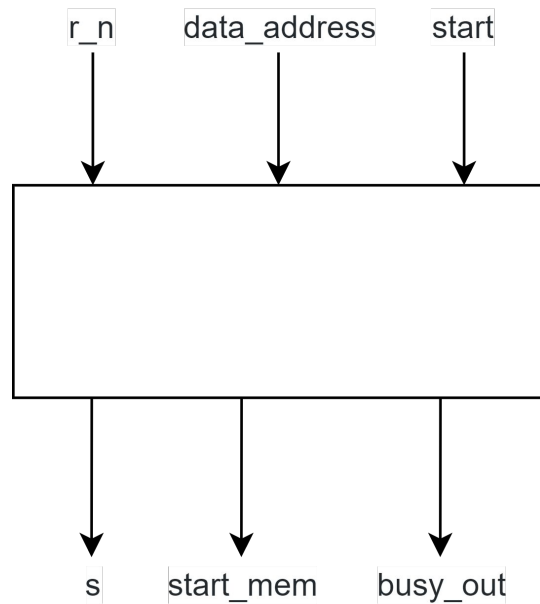
# Unidades Funcionais



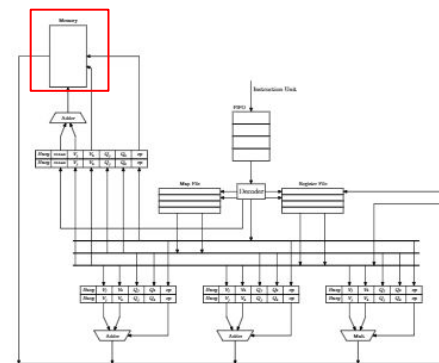
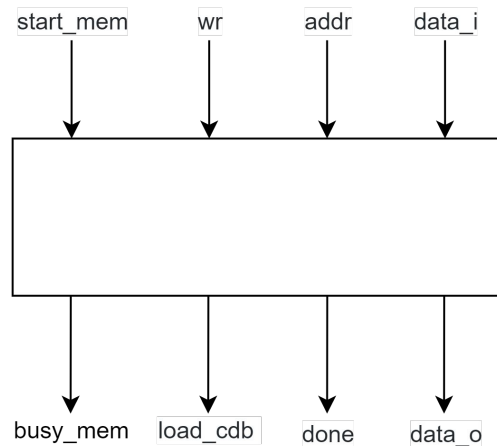
# Reservation Station (Memory)



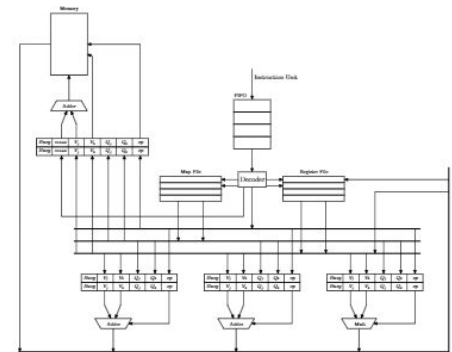
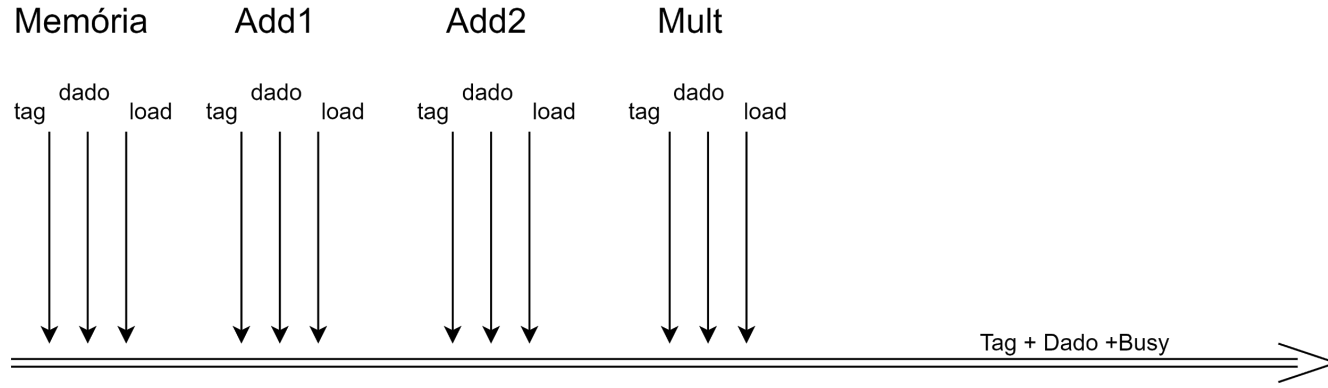
# Adder Memória



# Memória



# Common Data Bus (CDB)



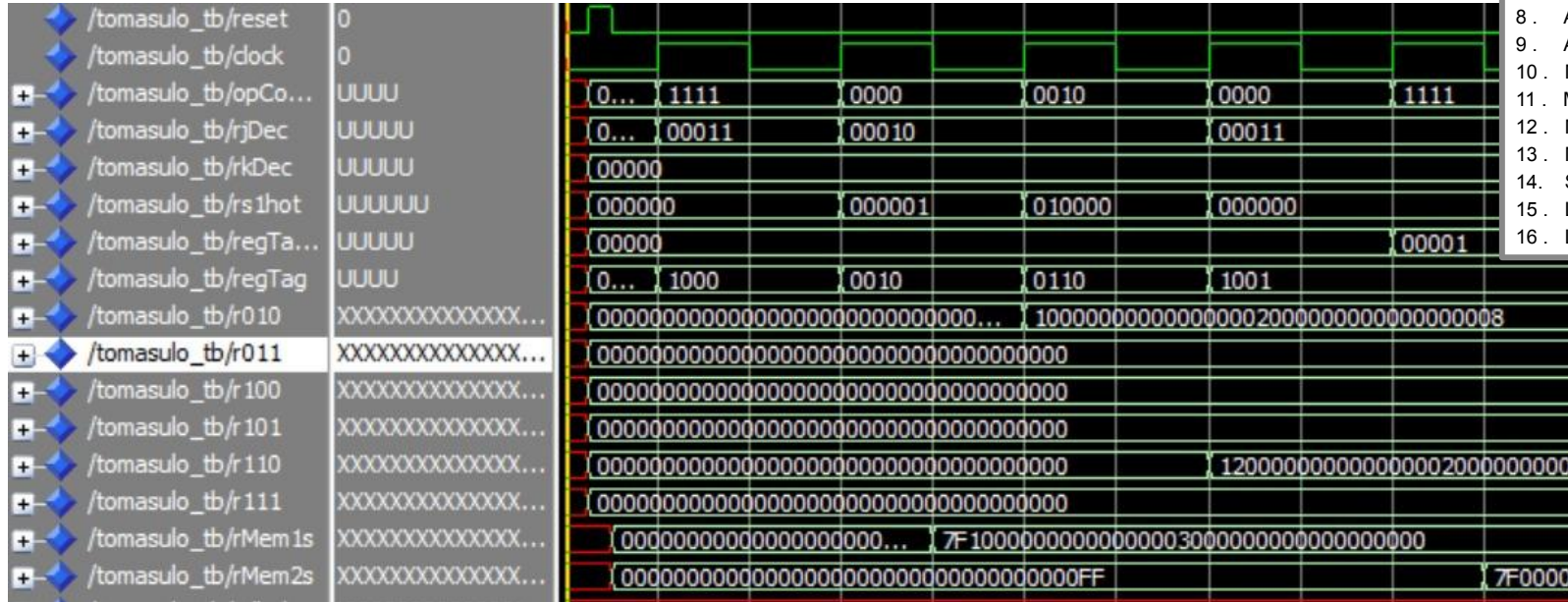
# Testbench do Circuito

```
1. LD    R0 , ( R3+504)
2. ADD   R0 , R0 , R2
3. MULT  R0 , R0 , R2
4. STR   R0 , ( R3+504)
5. LD    R1 , ( R3+504)
6. ADD   R4 , R3 , R3
7. ADD   R1 , R1 , R1
8. ADD   R1 , R1 , R1
9. ADD   R1 , R1 , R1
10. MULT R1 , R2 , R1
11. MULT R1 , R2 , R1
12. MULT R1 , R2 , R1
13. MULT R1 , R2 , R1
14. STR  R1 , ( R3+505)
15. LD   R0 , ( R3+505)
16. LD   R2 , ( R3+505)
```

## Programa Teste para simulação:

- Conflito de dados
- Paralelização de execução
- RS cheias
- Execução fora de ordem

# Testbench do Circuito

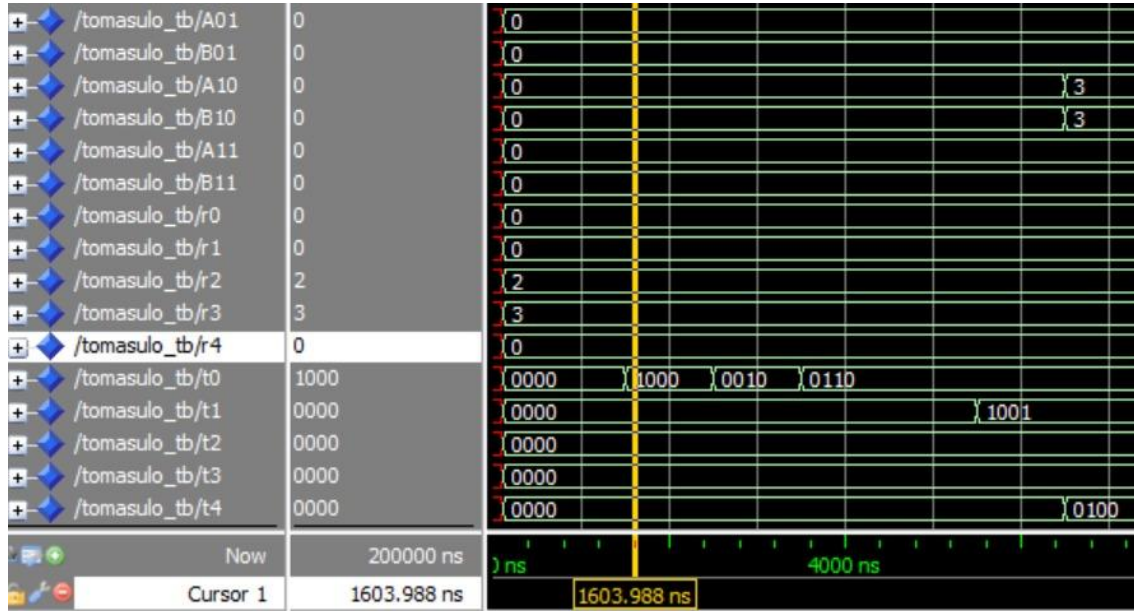


1. LD R0, (R3+504)
2. ADD R0, R0, R2
3. MULT R0, R2
4. STR R0, (R3+504)
5. LD R1, (R3+504)
6. ADD R4, R3, R3
7. ADD R1, R1, R1
8. ADD R1, R1, R1
9. ADD R1, R1, R1
10. MULT R1, R2, R1
11. MULT R1, R2, R1
12. MULT R1, R2, R1
13. MULT R1, R2, R1
14. STR R1, (R3+505)
15. LD R0, (R3+505)
16. LD R2, (R3+505)

- Expedição das primeiras instruções pelo Decoder
- Chegada das instruções nas RS



# Testbench do Circuito



```
1. LD R0, (R3+504)
2. ADD R0, R0, R2
3. MULT R0, R0, R2
4. STR R0, (R3+504)
5. LD R1, (R3+504)
6. ADD R4, R3, R3
7. ADD R1, R1, R1
8. ADD R1, R1, R1
9. ADD R1, R1, R1
10. MULT R1, R2, R1
11. MULT R1, R2, R1
12. MULT R1, R2, R1
13. MULT R1, R2, R1
14. STR R1, (R3+505)
15. LD R0, (R3+505)
16. LD R2, (R3+505)
```

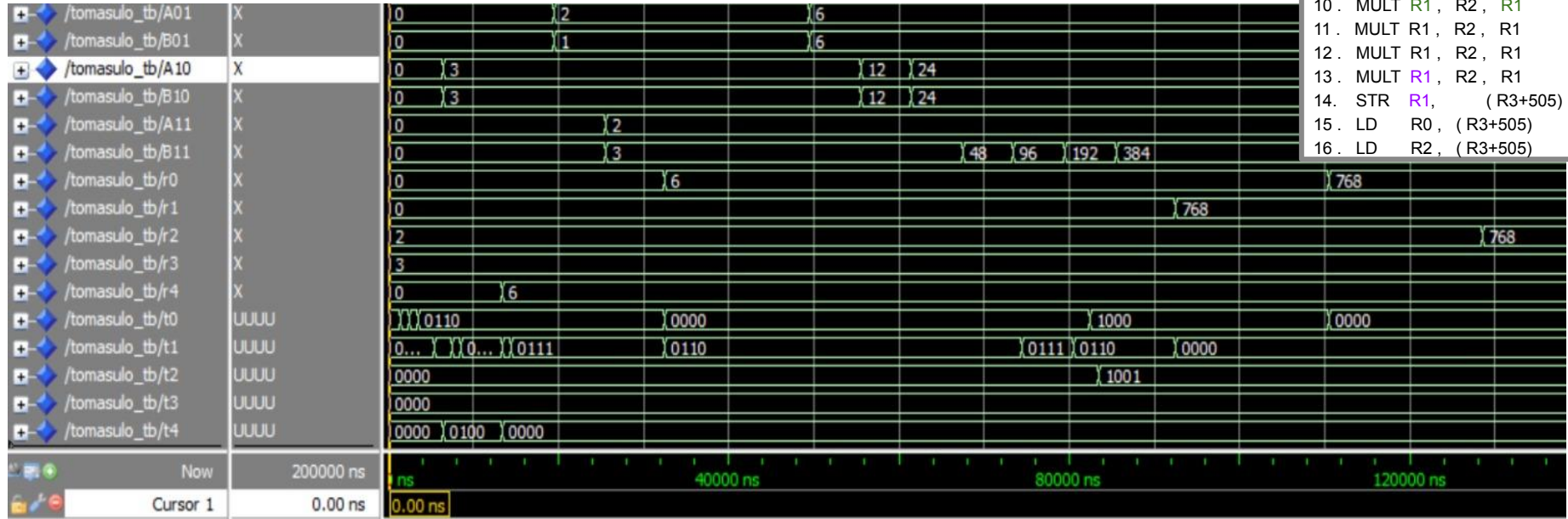
- Tags relativos as primeiras instruções sendo escritas na MapTable

# Testbench do Circuito

Address	Disassembly	Comment
0	ADD R1, R1, R1	
1	ADD R1, R1, R1	
0000	MULT R1, R2, R1	
00001	MULT R1, R2, R1	
00001	MULT R1, R2, R1	
000010	MULT R1, R2, R1	
00001	STR R1, (R3+505)	
0011	LD R0, (R3+505)	
1000000000000000...	LD R2, (R3+505)	
0000000000000000...		
1000000000000000...		
0000000000000000...		
0000000000000000...		
1200000000000000...		
0000000000000000...		
7F 100000000000...		
7F 000000000000...		
XXXXXXXXXXXXXXXX...		
0		
0		
3		
3		

- Instrução 6. ADD R4, R3, R3 sendo executada. ( $R3 = 3$ )
- Entrando na RS100 (Tag = 4) (RS de Add),
- Executada pela FU10
- Carregada no CDB (sinal 400...06) onde 4 é a Tag da RS ( $100 = 4$ ) 6 é o resultado ( $R4 = 3 + 3$ )

# Testbench do Circuito



- Sinais das entradas A e B de cada FU
- Execução fora de ordem (Inst 6 antes de 2,3,4,5)
- Final da execução (Valores nos R0, R1 e R2)

Questões ?

Obrigado !