

**Computer Architecture:  
Fundamentals, Tradeoffs, Challenges**

**Chapter 3: Microarchitecture Basics**

**Yale Patt**

W

**The University of Texas at Austin**

**Austin, Texas**

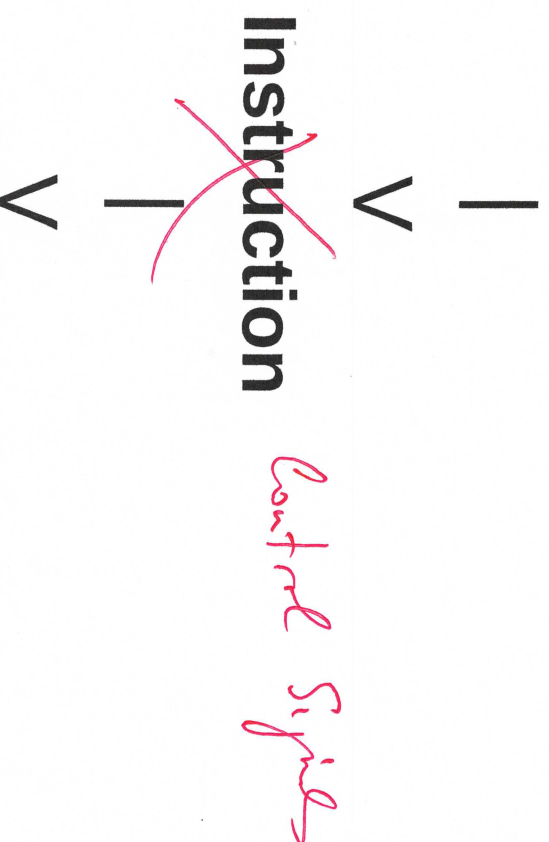
**Fall, 2022**

# Outline

- ***The job of the Microarchitecture***
  - *Changes the state of the machine, based on the instruction*
- ***Wilkes' original diode matrix***
- ***An example: The LC-3b***
  - *State machine, Data Path, Microsequencer, Control Signals*
- ***Tradeoffs***
- ***Pipelining***

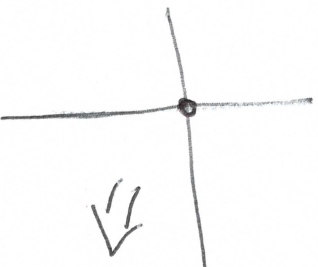
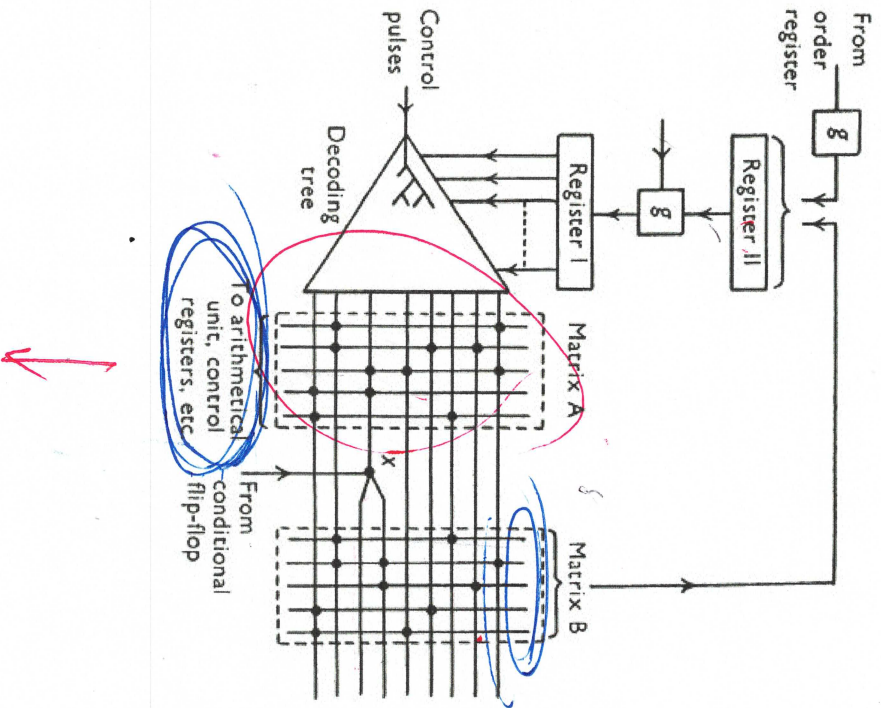
# Job of the Microarchitecture

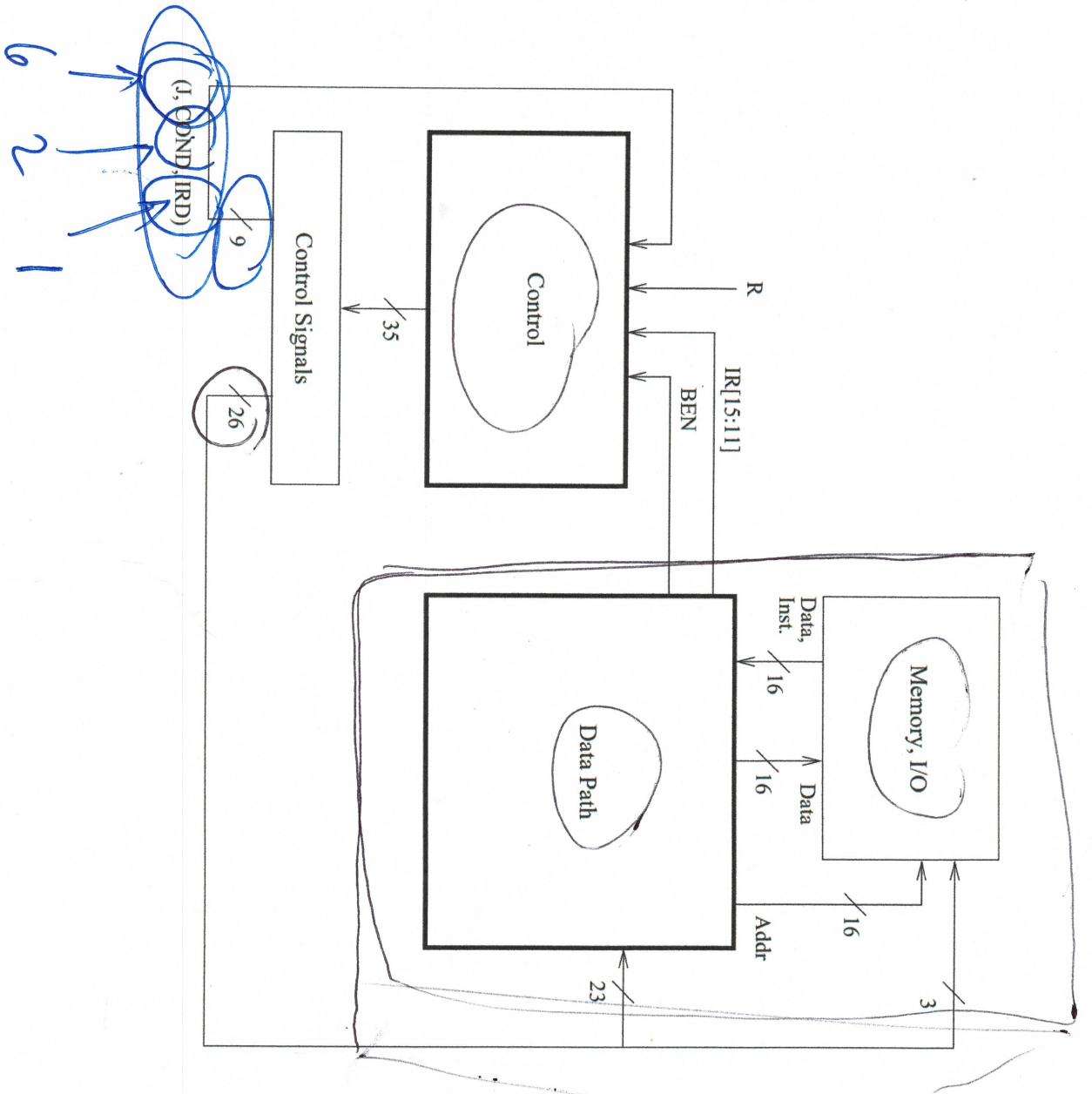
The machine is in a state



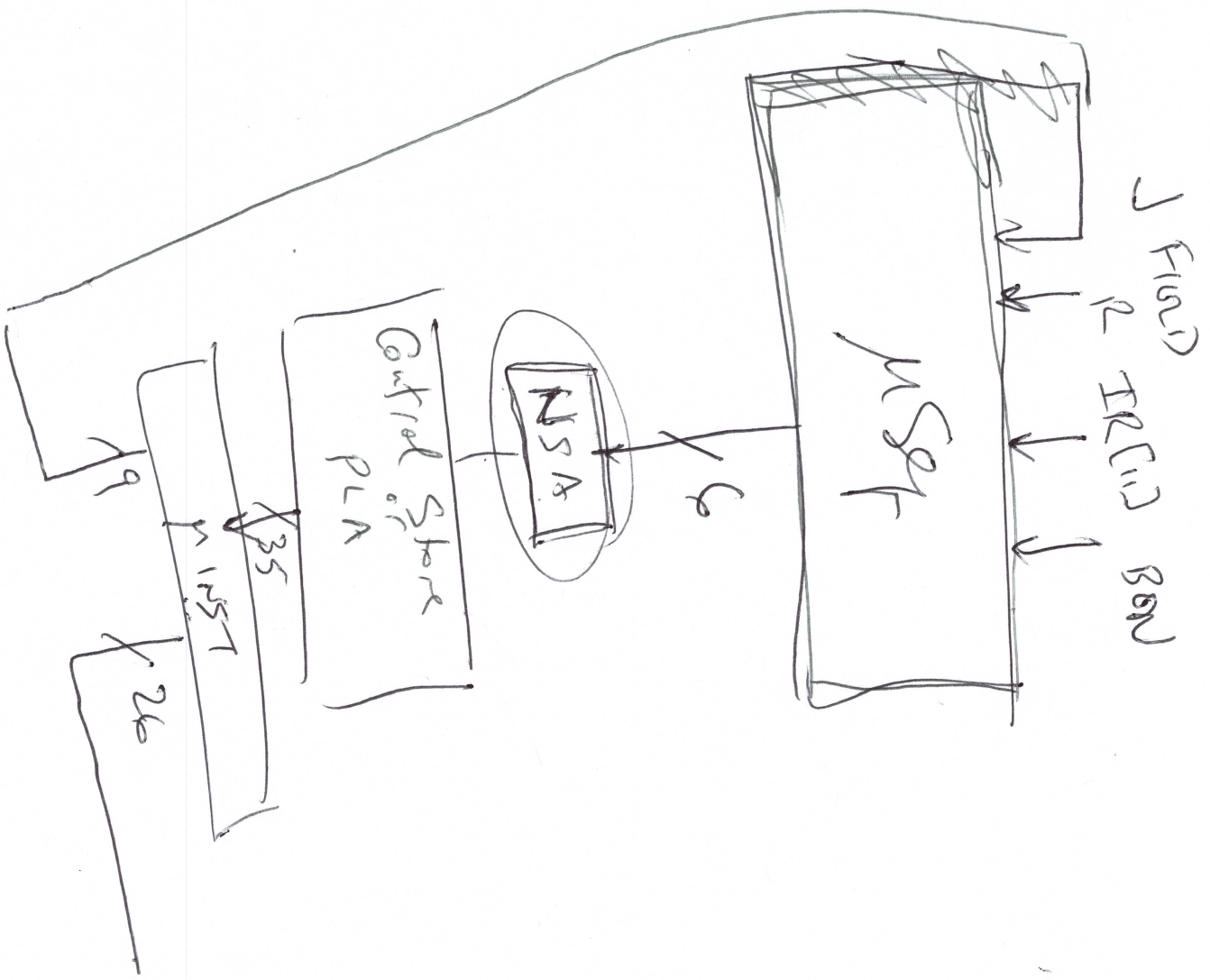
The machine is in a **new** state

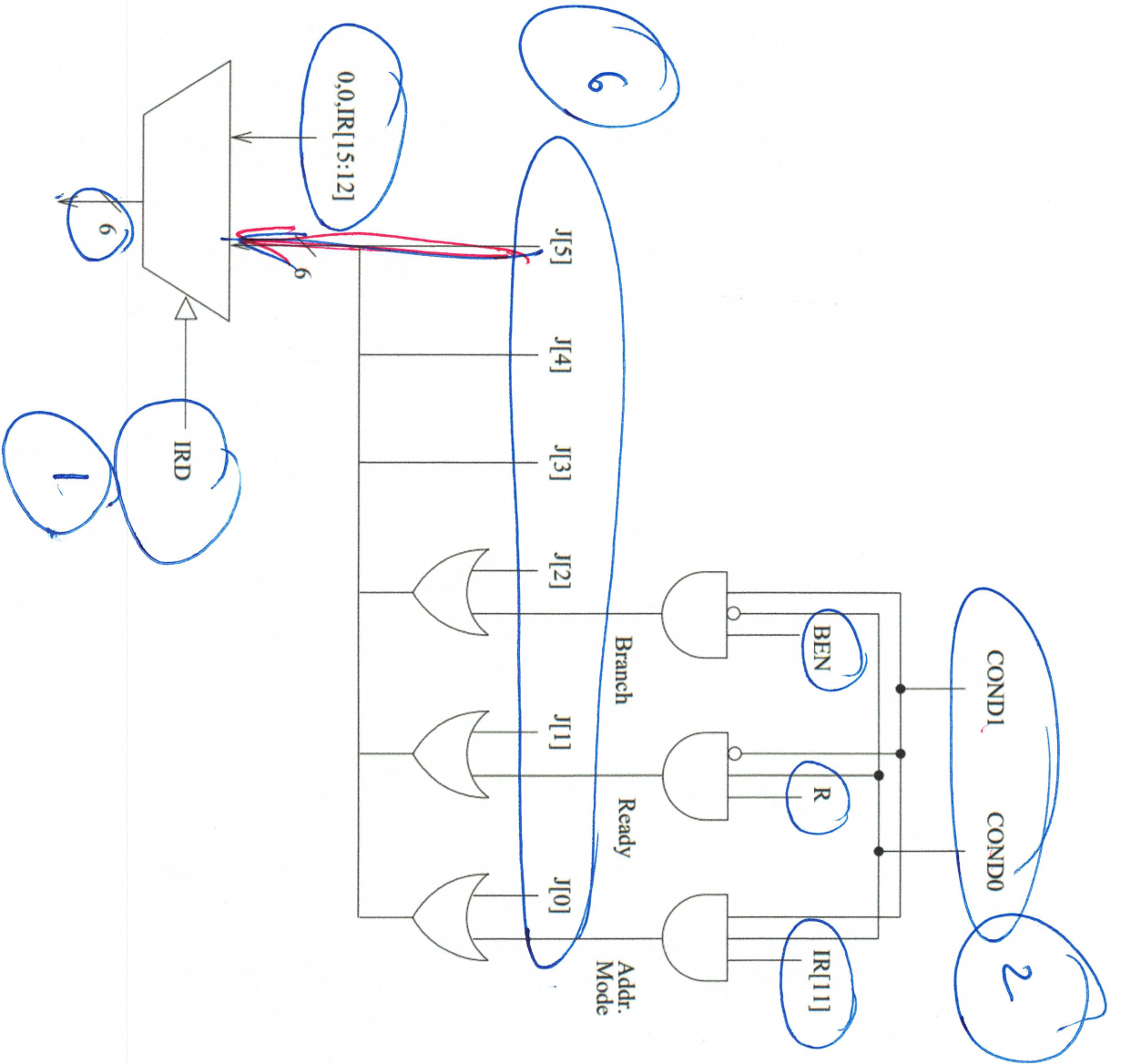
# Wilkes original diode matrices

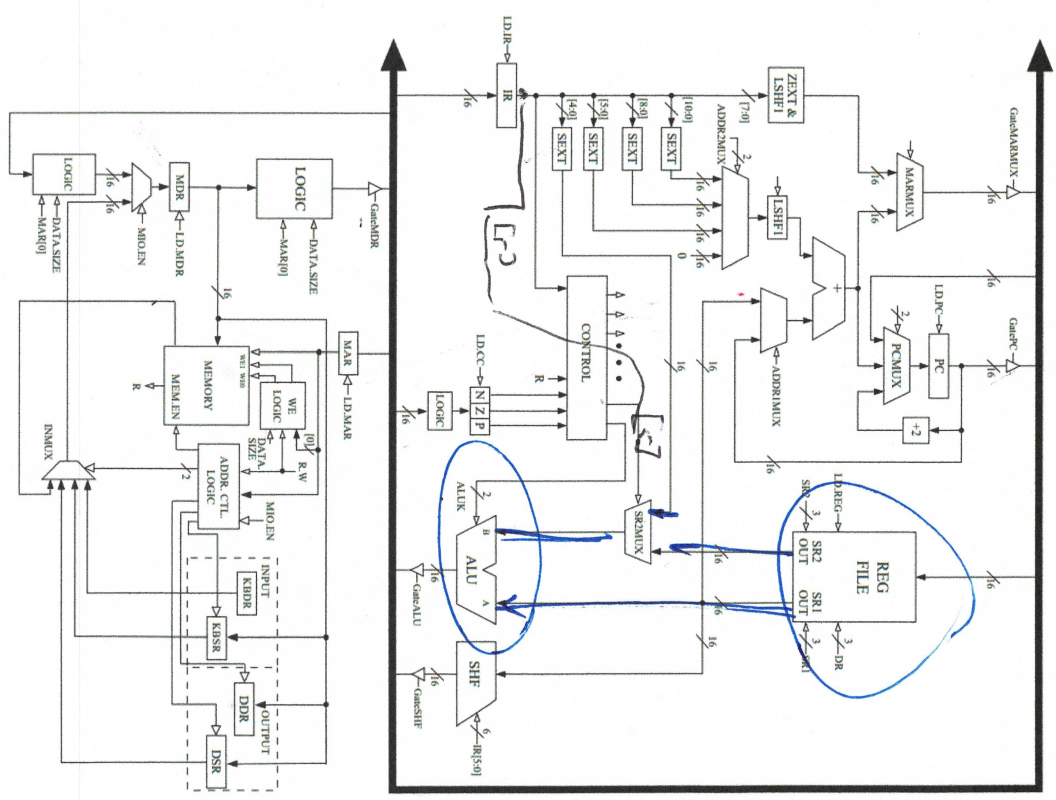




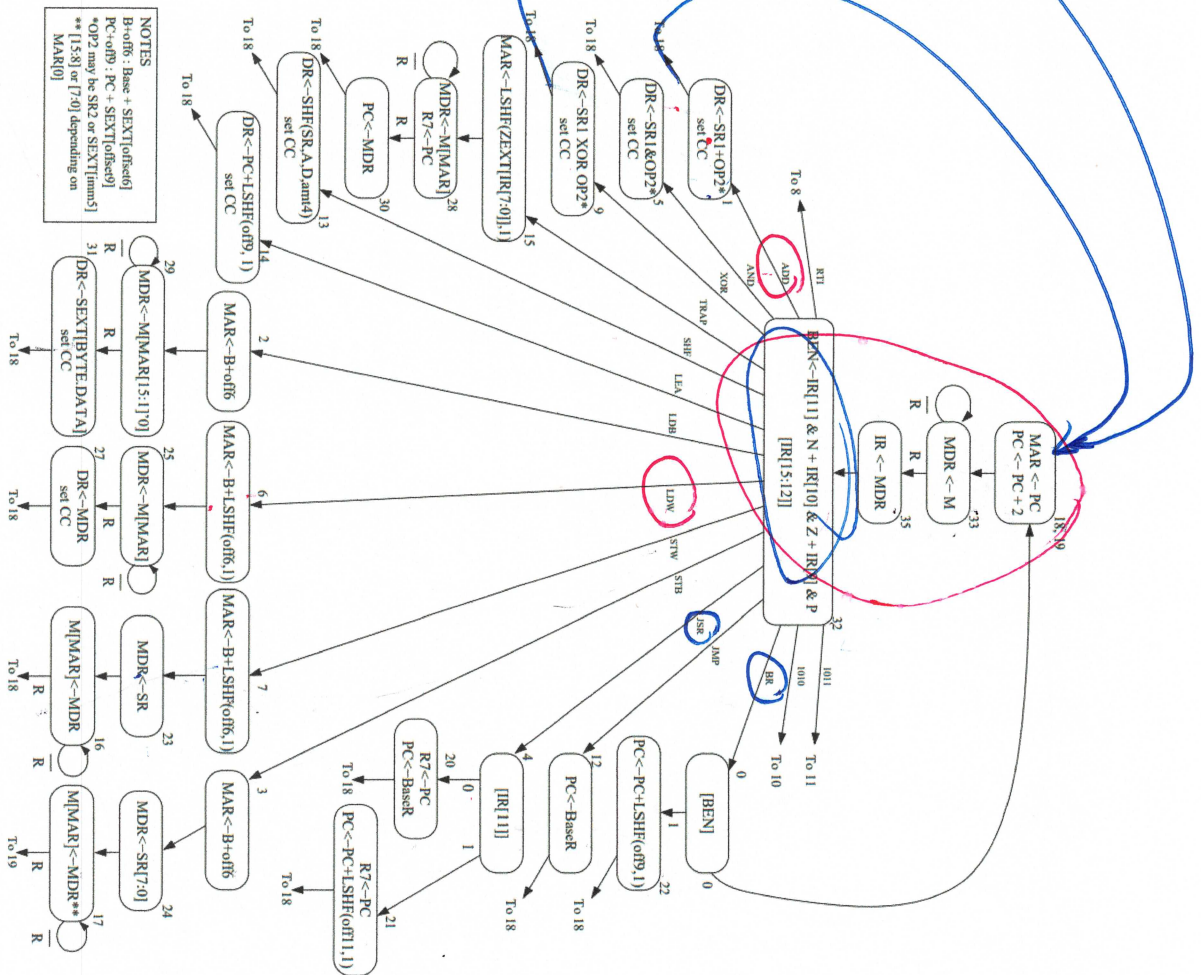
JSR  
 JMP  
 BR  
 TRAP  
 RTI











NOTES  
 B+ofb6 : Base + SEXT[ofs6]  
 PC+ofb9 : PC + SEXT[ofs9]  
 \*OP2 may be SR2 or SEXT[imm5] \*\* [1:5] or [7:0] depending on MAR[0]

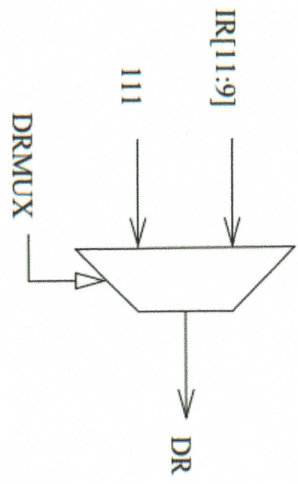
LD\_MAK  
 GATE\_PC / 5  
 GATE\_MON / 0  
 " ? / 0  
 REMUK / +2  
 LD\_PC / YES

Signal Name	Signal Values
LD.MAR/1:	NO, LOAD
LD.MDR/1:	NO, LOAD
LD.IR/1:	NO, LOAD
LD.BEN/1:	NO, LOAD
LD.REG/1:	NO, LOAD
LD.CC/1:	NO, LOAD
LD.PC/1:	NO, LOAD
GatePC/1:	NO, YES
GateMDR/1:	NO, YES
GateALU/1:	NO, YES
GateMARMUX/1:	NO, YES
GateSHF/1:	NO, YES
PCMUX/2:	PC+2 BUS ADDER
	:select pc+2 :select value from bus :select output of address adder
DRMUX/1:	11,9 R7
	:destination IR[11:9] :destination R7
SRMUX/1:	11,9 8,6
	:source IR[11:9] :source IR[8:6]
ADDR1MUX/1:	PC, BaseR
ADDR2MUX/2:	ZERO offset6 PCoffset9 PCoffset11
	:select the value zero :select SEXT[IR[5:0]] :select SEXT[IR[8:0]] :select SEXT[IR[10:0]]
MARMUX/1:	7,0 ADDER
	:select LSHF(ZEXT[IR[7:0]],1) :select output of address adder
ALUK/2:	ADD, AND, XOR, PASSA
MIO.EN/1:	NO, YES
R.W/1:	RD, WR
DATA.SIZE/1:	BYTE, WORD
LSHF/1:	NO, YES

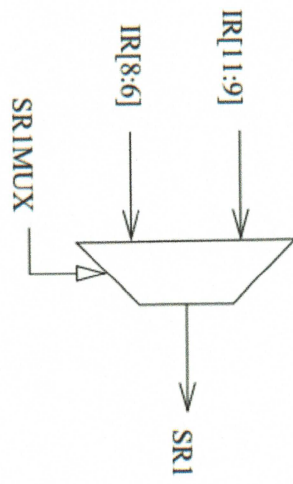
Table C.1: Data path control signals

Signal Name	Signal Values
J/6:	
COND/2:	COND <sub>0</sub> : Unconditional COND <sub>1</sub> : Memory Ready COND <sub>2</sub> : Branch COND <sub>3</sub> : Addressing Mode
IRD/1:	NO, YES

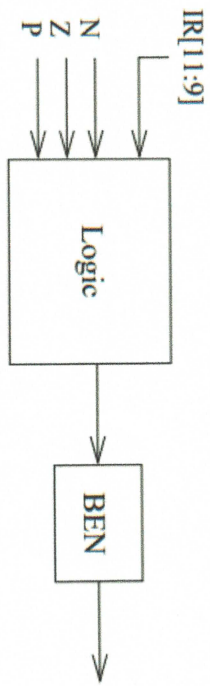
Table C.2: Microsequencer control signals



(a)



(b)



# Tradeoffs

- **Vertical vs Horizontal Microcode**
  - Selective (generally one micro-op per ~~micro-op~~ *clock cycle*)
  - Constructive (many fields, each specified)
- **One level vs Two level Microcode**
  - Example: the Nanodata QM-1 (360 bit nanoinstructions)
- **CPI vs cycle time (or, IPC vs frequency)**
- **In-order vs out-of-order execution**
- **Issue width**
- **Use of chip real estate**
  - Better branch predictor, accelerators, microcode
- **Unified vs separate instruction and data caches**
- **On-chip interconnect (cost, contention, latency)**
- **Prefetching**

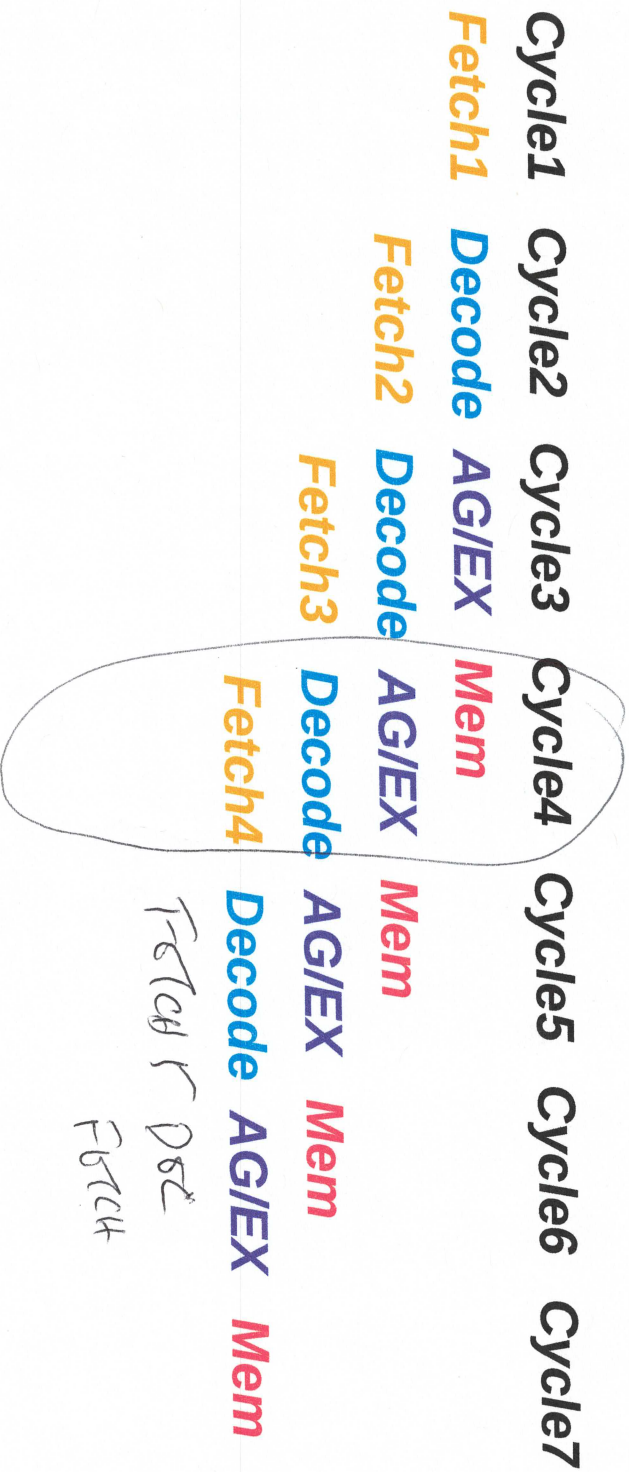
## TRADEOFFS (CONTINUED)

- SPECULATE VS STAND AROUND AND WAIT
- ASIC VS PROGRAM CONTROL
- PIPELINE DEPTH (ACTUALLY, CPI VS CYCLE TIME)
- CACHE STRUCTURES
- ON-CHIP LATENCY (NEAR NEIGHBOR COMMUNICATION)
- ~~MAKES~~ FAULT TOLERANCE (AVAILABILITY)
  - YESTERDAY, A NICHE MARKET, TANDEM 2 CORRS IN LOCK STEP
  - TODAY/TOMORROW - SOFT ERRORS, ALL MACHINES
  - TWO CORRS IN LOCKSTEP OR EXECUTE AND REPEAT
  - TRIPLE MODULAR REDUNDANCY
- PARTITIONING (UNTIL WHEN)

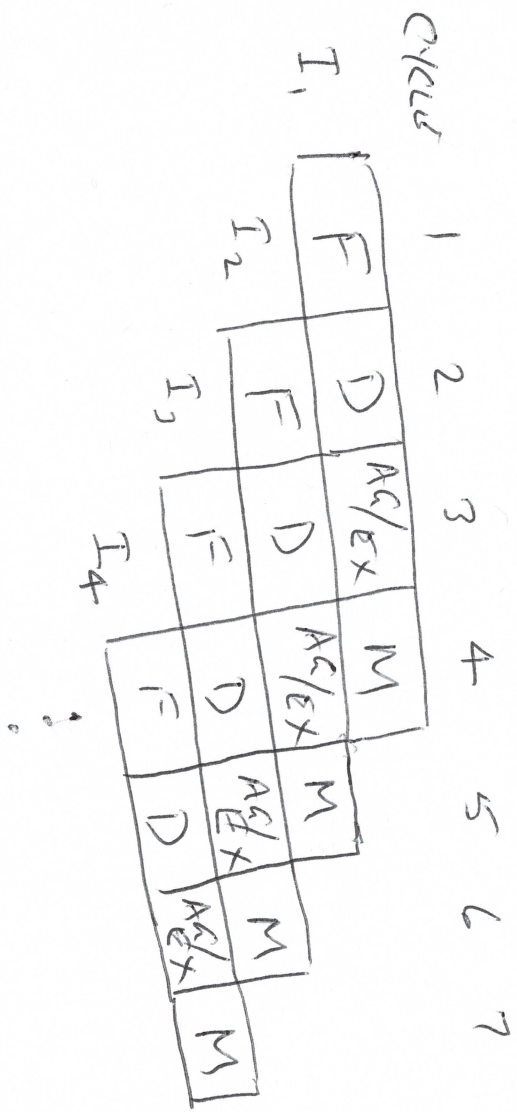
# A Simple Pipeline

ASSEMBLY  
LINK

- Four stage pipeline (each stage takes one cycle)
  - **Fetch** (instruction fetched from on chip storage)
  - **Decode** (instruction decoded)
  - **Address Generation & Execute** (result latched)
  - **Memory access** (loads and stores to on chip storage)



# A PIPELINE



MAC-015

~~ADD R1, R2, R3~~

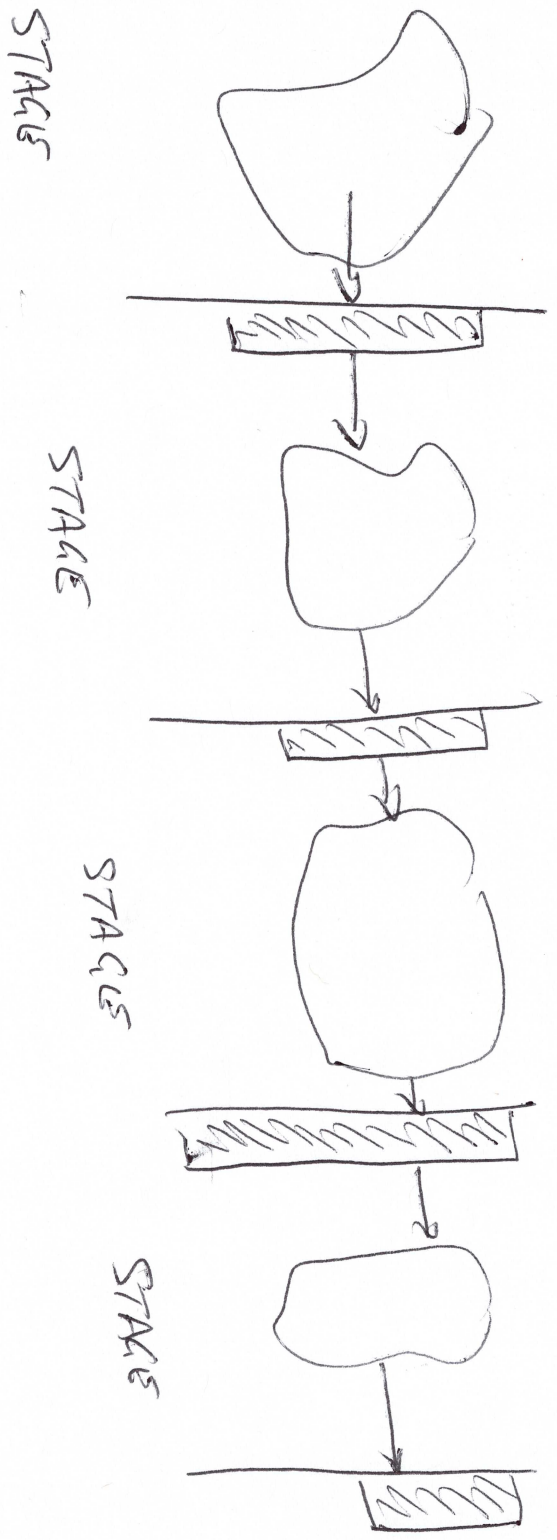
~~ADD R1, R2, R3~~



~~WRITE MEMORIES~~



# A PIPELINE

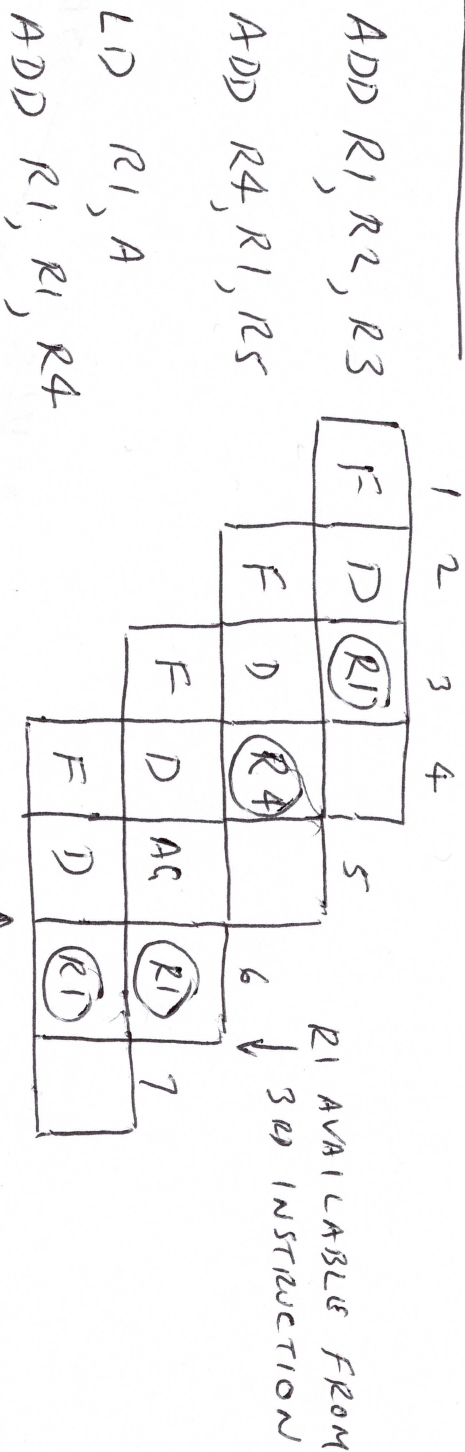


# Pipeline Problems

- **Flow dependency**
  - Instruction 1: ADD R1,R2,R3 (R1 loaded at end of cycle 3)
  - Instruction 2: ADD R4,R1,R5 (R4 loaded at end of cycle 4)
  - Instruction 3: LD R1,A (R1 loaded at end of cycle 6)
  - Instruction 4: ADD R1,R1,R4 (R1 read at start of cycle 6)  
**Incorrect data read from R1 !!!**
- **Conditional branches**
  - Instruction 1: ADD R1,R2,R3 (cc loaded at end of cycle 3)
  - Instruction 2: BRZ A
  - Instruction 3: Fetch address needed at start of cycle 3)  
**Microarchitecture does not know the fetch address !!!**

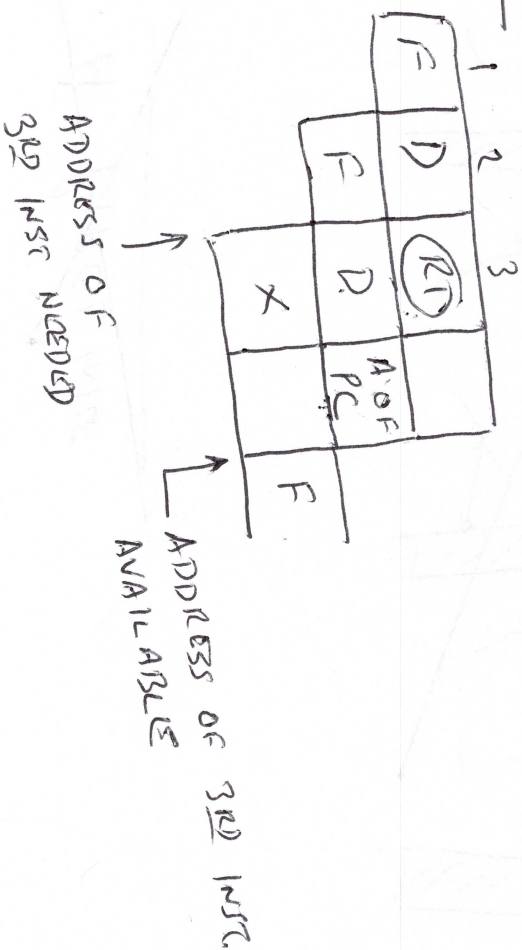
# PIPELINE PROBLEMS (Cont.)

## Flow Dependency



## CONDITIONAL BRANCHES

ADD R1, R2, R3  
BR2 A  
3RD INSTRUCTION

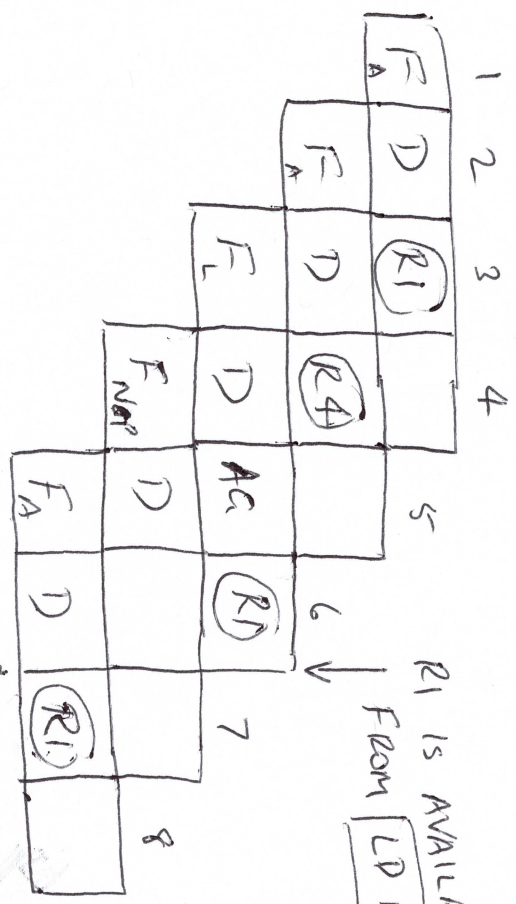


# MIPS SOLUTION

(WITHOUT INTERLOCKS)

```

ADD R1, R2, R3
ADD R4, R1, R5
LD R1, A
NOP
ADD R1, R1, R4
    
```



R1 IS AVAILABLE FROM LD R1, A

R1 IS NEEDED FOR ADD R1, R1, R4

# SCOREBOARD SOLUTION (HARDWARE INTERLOCKS)

A BIT FOR EVERY RESOURCE

EXAMPLE

ADD R0, R1, R2

BEFORE:  
DECODE

R0	1					
R1	1					
R2	1					
...						
Rn						

SCOREBOARD BIT

REGISTER  
FILE

AFTER  
DECODE

R0	0					5
R1	1					6
R2	1					7
...						...
Rn						

REGISTER  
FILE

**Shukraan jazilaan!**