Computer Architecture: Fundamentals, Tradeoffs, Challenges

Chapter 3: Microarchitecture Basics

Yale Patt

The University of Texas at Austin

Austin, Texas

Spring, 2023

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***

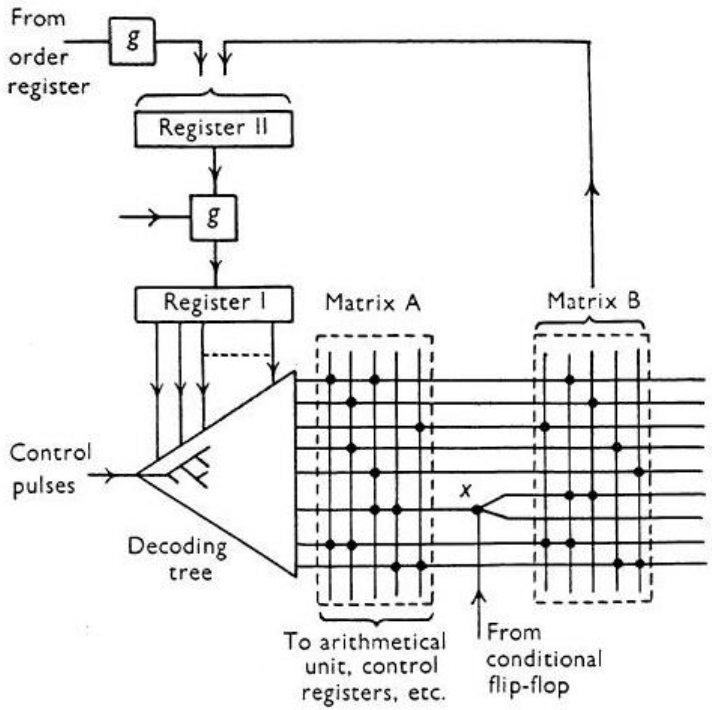
Job of the Microarchitecture

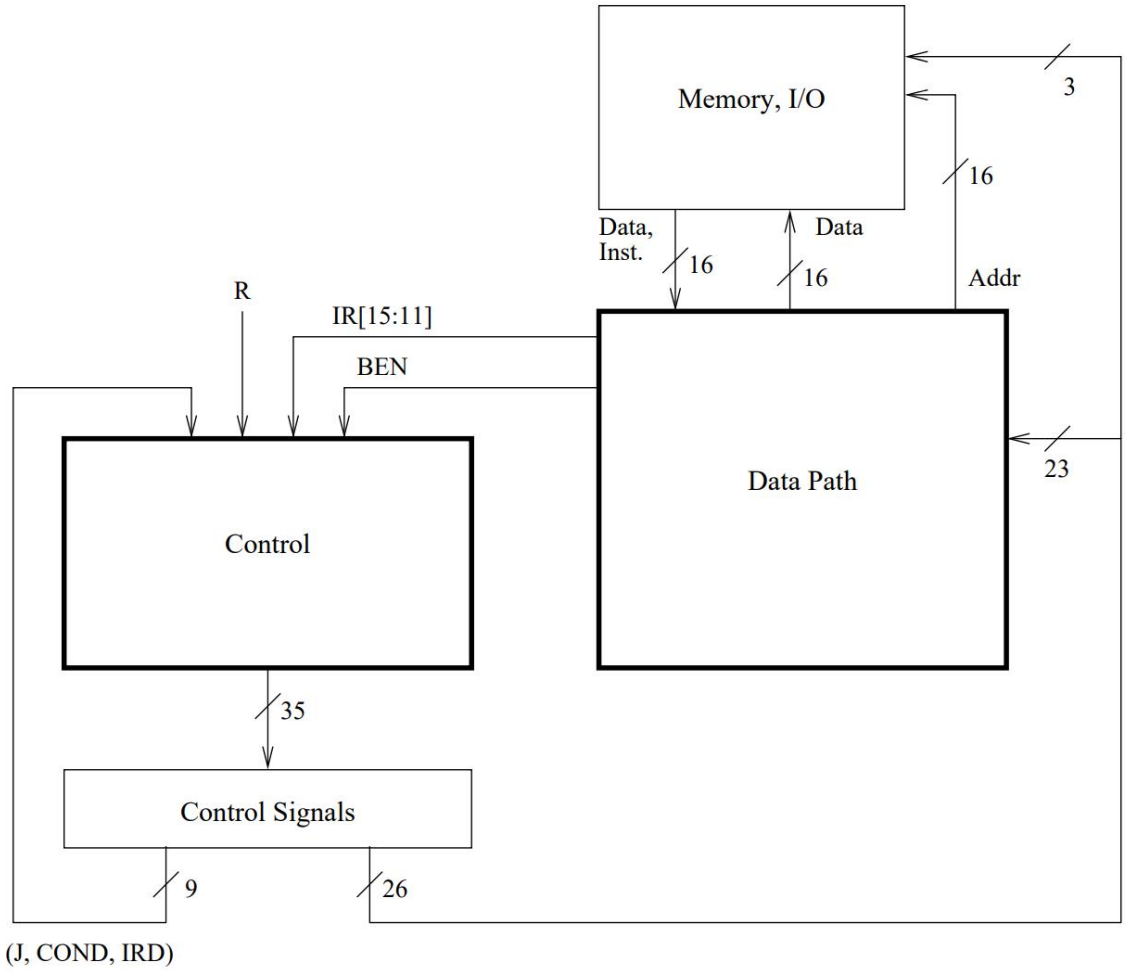
The machine is in a state

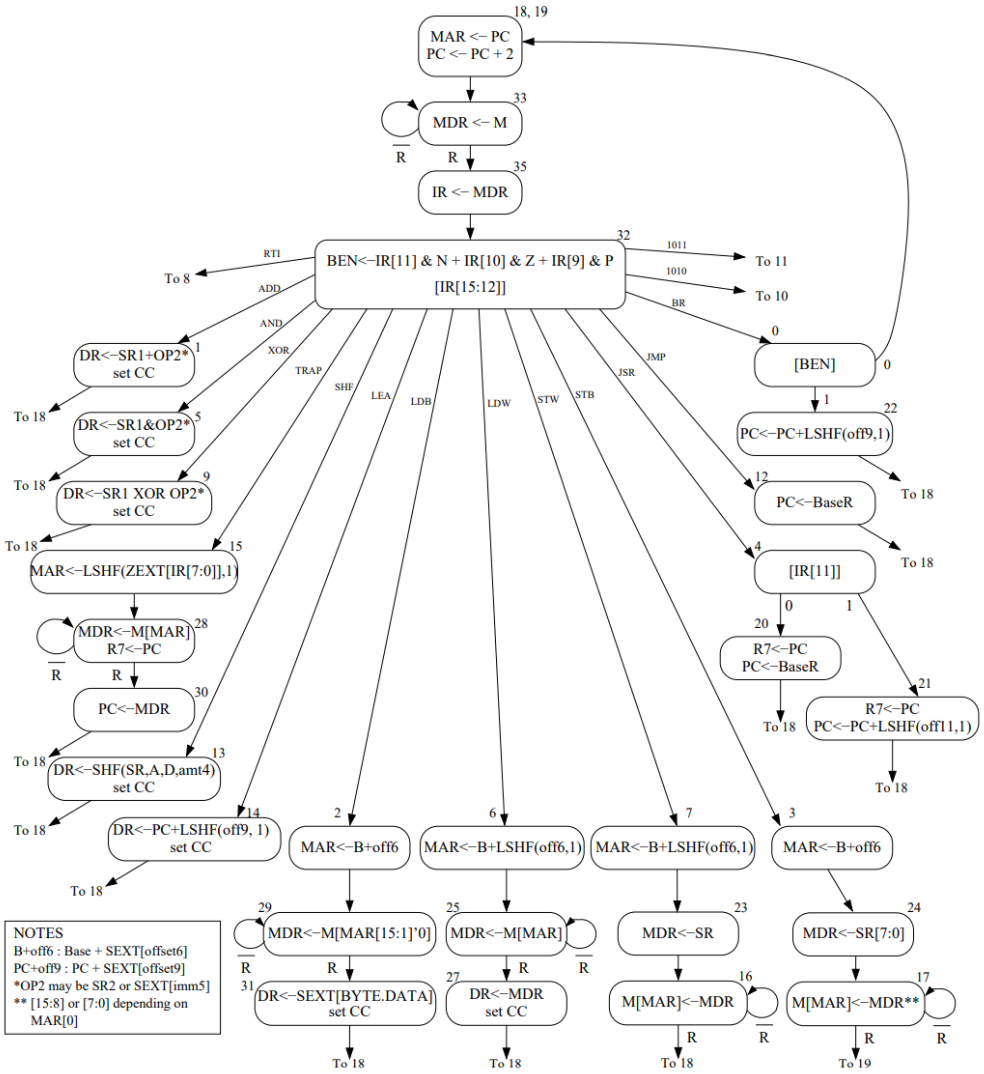
Instruction

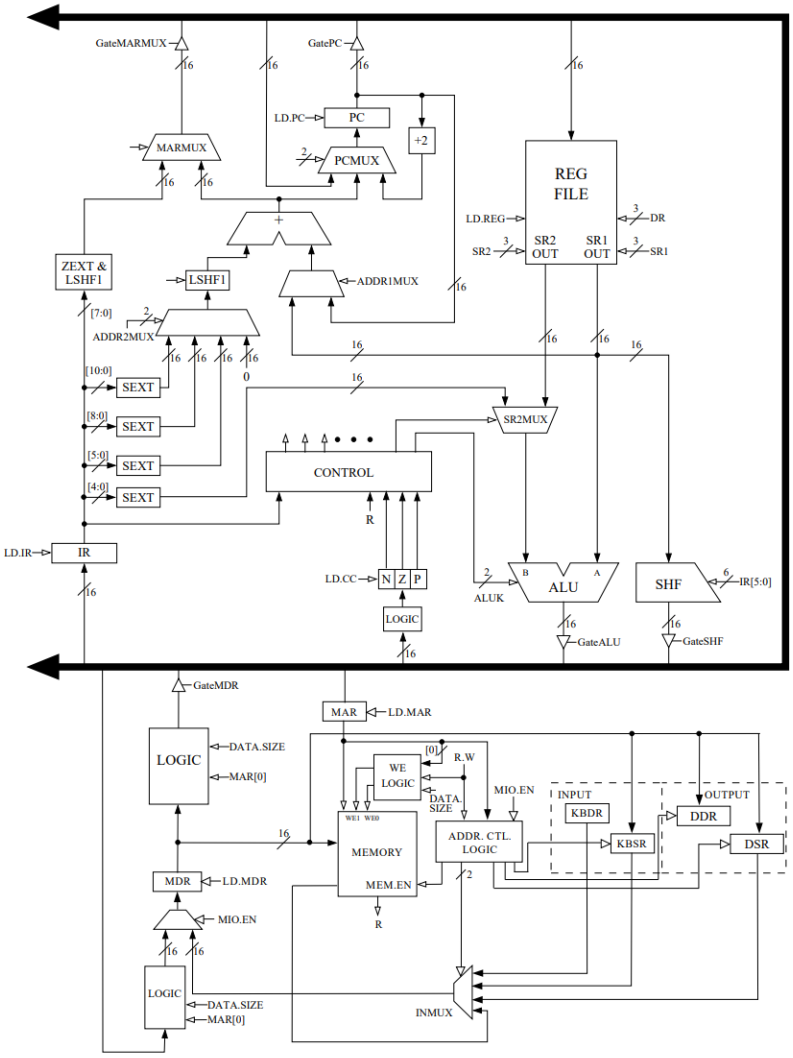
The machine is in a new state

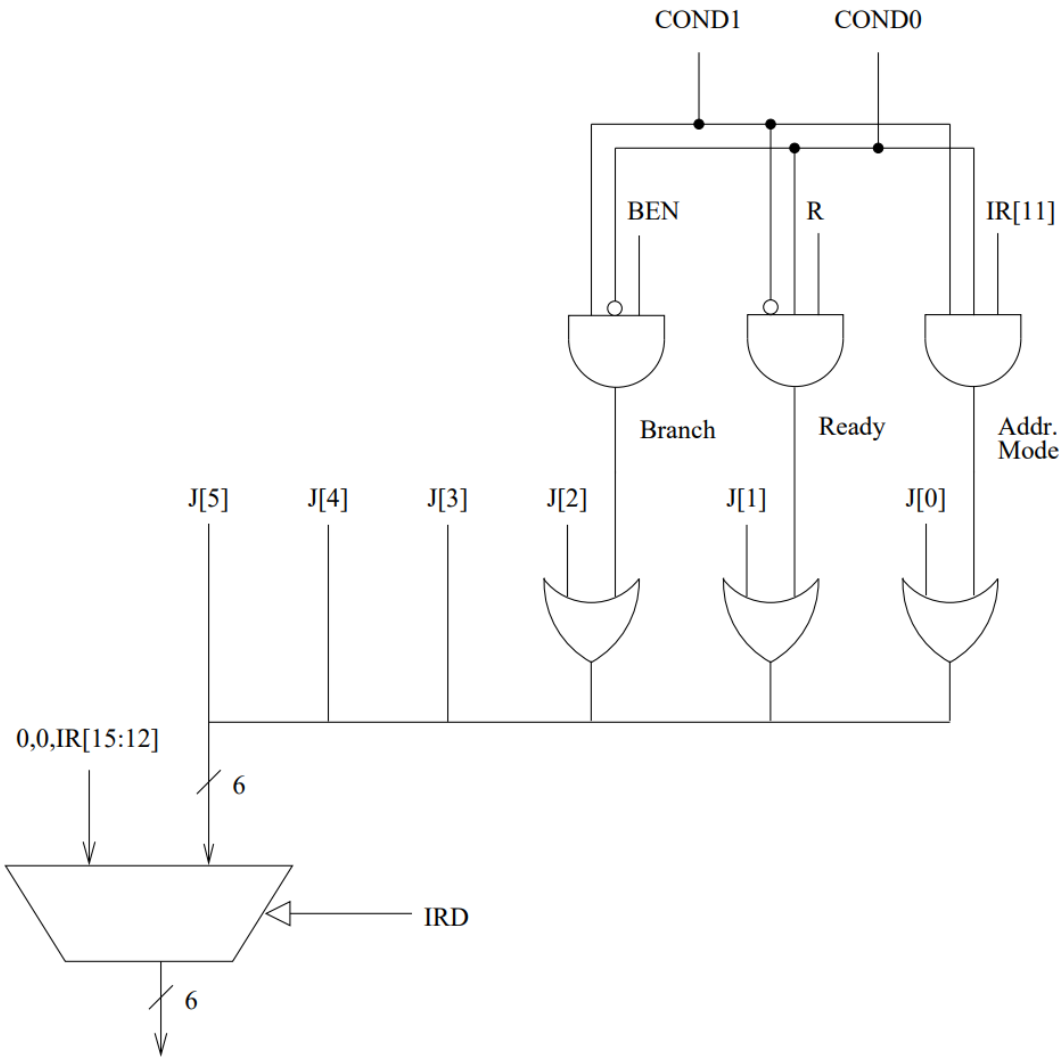
Wilkes original diode matrices

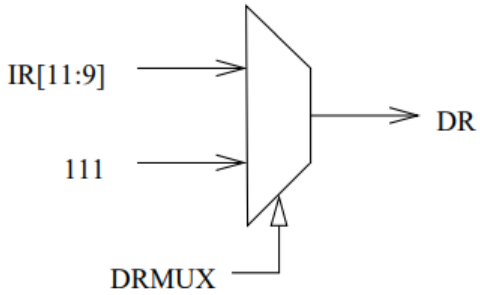



Memory, I/O
3
16
Data,
Inst.
16
Data
16
Addr
R
IR[15:11]
BEN
Control
Data Path
23
35
Control Signals
9
26
(J, COND, IRD)

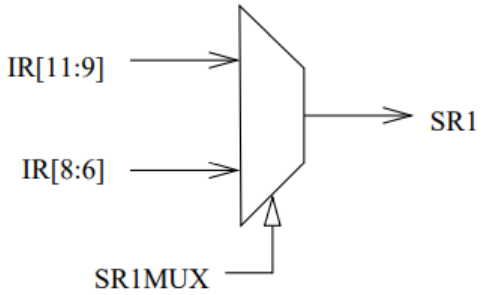

18, 19
MAR <- PC
PC <- PC + 2
33
MDR <- M
R
R
35
IR <- MDR
32
BEN<-IR[11] & N + IR[10] & Z + IR[9] & P
[IR[15:12]]
RTI
To 8
ADD
AND
XOR
TRAP
SHF
LEA
LDB
LDW
STW
STB
JSR
JMP
1011
To 11
1010
To 10
BR
0
[BEN]
0
1
22
PC<-PC+LSHF(off9,1)
To 18
12
PC<-BaseR
To 18
4
[IR[11]]
0
1
20
R7<-PC
PC<-BaseR
To 18
21
R7<-PC
PC<-PC+LSHF(off11,1)
To 18
1
DR<-SR1+OP2*
set CC
To 18
5
DR<-SR1&OP2*
set CC
To 18
9
DR<-SR1 XOR OP2*
set CC
To 18
15
MAR<-LSHF(ZEXT[IR[7:0]],1)
28
MDR<-M[MAR]
R7<-PC
R
R
30
PC<-MDR
To 18
13
DR<-SHF(SR,A,D,amt4)
set CC
To 18
14
DR<-PC+LSHF(off9, 1)
set CC
To 18
2
MAR<-B+off6
29
MDR<-M[MAR[15:1]'0]
R
R
31
DR<-SEXT[BYTE.DATA]
set CC
To 18
6
MAR<-B+LSHF(off6,1)
25
MDR<-M[MAR]
R
R
27
DR<-MDR
set CC
To 18
7
MAR<-B+LSHF(off6,1)
23
MDR<-SR
16
M[MAR]<-MDR
R
R
To 18
3
MAR<-B+off6
24
MDR<-SR[7:0]
17
M[MAR]<-MDR**
R
R
To 19
NOTES
B+off6 : Base + SEXT[offset6]
PC+off9 : PC + SEXT[offset9]
*OP2 may be SR2 or SEXT[imm5]
** [15:8] or [7:0] depending on
MAR[0]


GateMARMUX
GatePC
LD.PC
PC
+2
PCMUX
MARMUX
REG FILE
LD.REG
DR
SR2
SR2 OUT
SR1 OUT
SR1
ZEXT & LSHF1
LSHF1
ADDR1MUX
ADDR2MUX
[7:0]
[10:0]
[8:0]
[5:0]
[4:0]
SEXT
SR2MUX
CONTROL
R
LD.IR
IR
LD.CC
N
Z
P
LOGIC
ALUK
ALU
SHF
IR[5:0]
GateALU
GateSHF
GateMDR
MAR
LD.MAR
DATA.SIZE
MAR[0]
WE LOGIC
R.W
MIO.EN
INPUT
OUTPUT
KBDR
DDR
KBSR
DSR
MEMORY
MEM.EN
ADDR. CTL. LOGIC
MDR
LD.MDR
INMUX

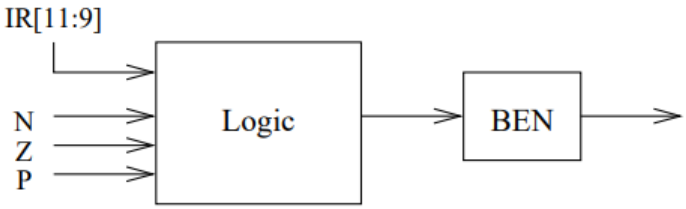

COND1
COND0
BEN
R
IR[11]
Branch
Ready
Addr.
Mode
J[5]
J[4]
J[3]
J[2]
J[1]
J[0]
0,0,IR[15:12]
6
IRD
6



(a)



(b)



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	;select pc+2
	BUS	;select value from bus
	ADDER	;select output of address adder
DRMUX/1:	11.9	;destination IR[11:9]
	R7	;destination R7
SR1MUX/1:	11.9	;source IR[11:9]
	8.6	;source IR[8:6]
ADDR1MUX/1:	PC, BaseR	
ADDR2MUX/2:	ZERO	;select the value zero
	offset6	;select SEXT[IR[5:0]]
	PCoffset9	;select SEXT[IR[8:0]]
	PCoffset11	;select SEXT[IR[10:0]]
MARMUX/1:	7.0	;select LSHF(ZEXT[IR[7:0]],1)
	ADDER	;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	
LSHF1/1:	NO, YES	

Table C.1: Data path control signals

Signal Name	Signal Values	
J/6:		
COND/2:	$COND_0$	;Unconditional
	$COND_1$	;Memory Ready
	$COND_2$	;Branch
	$COND_3$	;Addressing Mode
IRD/1:	NO, YES	

Table C.2: Microsequencer control signals

Tradeoffs

- ***Vertical vs Horizontal Microcode***
  - ***Selective (generally one micro-op per micro-op)***
  - ***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***

A Simple Pipeline

- ***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)***

Cycle1	Cycle2	Cycle3	Cycle4	Cycle5	Cycle6	Cycle7
Fetch1	Decode	AG/EX	Mem			
	Fetch2	Decode	AG/EX	Mem		
		Fetch3	Decode	AG/EX	Mem	
			Fetch4	Decode	AG/EX	Mem

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)***

  Microarchitecturre does not know the fetch address !!!

Shukraan jazilaan!