Clock = load value into register

RISCEE 3 Architecture

Clock

ADD1

2

P C

Read address

Instruction [15-0]

Instruction Memory

Instruction[7-0]

ALUop

1 X+0

2 X-Y

3 0+Y

4 0

5 X+Y

Zero

AND gate

BZ

OR gate

P0

RegDst

ADD2

RegWrite

Read Data

Write Data

Accumulator

Read Data

Write Data

MemWrite

AND

NOT

MemWrite

Read Data

Read Data

ADD2

Address

Write Data

Write Data

ALU3

X

Y

Y

X

Clock
Instruction: clear

Operation: A=0;

RISCEE 3 Architecture

RegDst=2 ALU=4 MemWrite=0 RegWrite=1 BZ=0 P0=0
Instruction: \textit{addi data8}

Operation: \[ A = A + \text{data8}; \]

RISCEEE 3 Architecture

\begin{itemize}
  \item \textbf{RegDst}=2
  \item \textbf{ALU}=5
  \item \textbf{MemWrite}=0
  \item \textbf{RegWrite}=1
  \item \textbf{BZ}=0
  \item \textbf{P0}=0
\end{itemize}
Instruction: add addr8  
Operation: \( A = A + \text{Memory}[\text{addr8}] \);
Instruction: store addr8

Operation: Memory[addr8] = A;

RISCEE 3 Architecture

RegDst = X  ALU = 3  MemWrite = 1  RegWrite = 0  BZ = 0  P0 = 0
Instruction: bne addr8

Operation: if (A != 0) { pc=addr8; }

RegDst = X  ALU = 3  MemWrite = 1  RegWrite = 0  BZ = 0  P0 = 0
Instruction: apc  
Operation: A=pc+2;

RISCEEE 3  
Architecture

RegDst=X  ALU=X  MemWrite=1  RegWrite=0  BZ=0  P0=0
Clock = load value into register

RISCEEE 4 Architecture

Clock

P0 | (~AluZero & BZ)

PC

IorD

MemRead

Read

Data

Write

Data

MemWrite

MDR

MDR2

Instruction[7-0]

IRWrite

IR

Read

Data

Write

Data

Read

Data

Write

Data

RegWrite

RegDst

ALU

ALUop

1 X+0

2 X-Y

3 0+Y

4 0

5 X+Y

ALUsrcA

ALUsrcB

PCSsrc

1 0 2

0 1 2

0 1

1 0

0

2

Y

X

ALU Out

Out

P0 | (~AluZero & BZ)
T₁ all instructions  # IR=Mem[PC] & pc=pc+2

- P₀ = 0 | (~AluZero & BZ)
- P₀ = 1
- BZ = X
- ALUsrcB = 0
- ALUsrcA = 0
- ALUop:
  1 X+0
  2 X−Y
  3 0+Y
  4 0
  5 X+Y

- MDR:
  - MDR2
  - Instruction[7-0]
  - IR
  - IRWrite
  - MemRead
  - MemWrite

- PC:
  - PC

- Accumulator:
  - Read Data
  - Write Data
  - RegWrite
  - RegDst

- ALU:
  - ALUop
  - ALUOut

- Out

- Y

- X
T<sub>2</sub> all instructions #opcode decode

RISCEE 4 Architecture

P0 | (~AluZero & BZ)

P0 = 0
BZ = 0

= X

PCSrc

ALUsrcB = X

= X

MDR2

Instruction[7-0]

ALU

ALUop
1 X+0
2 X-Y
3 0+Y
4 0
5 X+Y

ALUOut

MemRead

= 0

MDR

MemWrite

= 0

Read Data

Write Data

IR

Write Data

IRWrite

RegWrite

= X

RegDst

= X

1 0

Accumulator

Read Data

Write Data

1 0

P0 = 0

BZ = 0

= X

IorD

= 0

PCSrc

= X
RISCEE 4 Architecture

T₃ clear # ALUOut=0

P₀ | (~AluZero & BZ)

P₀=0
BZ=0

=X
PCSrc

ALUsrcB =X

ALUop
1 X+0
2 X-Y
3 0+Y
4 0
5 X+Y

ALU

ALU Out

Y

X

=4

MDR2

=0

MemRead

Instruction[7-0]

IRWrite

MDR

RegWrite

RegDst

1
0

Accumulator

Write Data

Read Data

Read

Write

Data

Data

MemWrite

=X

=X

T3   clear  # ALUOut=0
T₄ clear # A=ALUOut

RISCEE 4 Architecture

P₀ | (~AluZero & BZ)
P₀=0  BZ=0

ALU

ALUoperation:
1 X+0
2 X-Y
3 0+Y
4 0
5 X+Y

ALUsrcA =X

ALUsrcB =X

MDR =X

MDR2 =X

Instruction[7-0]

IR

IRWrite

RegWrite

RegDst

MemWrite

MemRead

address

Read Data

Write Data

PC

0

1

2

ALUOut

Y

ALU

0

1

2

=4

=1

=0
RISCEE 4 Architecture

T₃  add # ALUOut=IR[7-0]

P₀ = 0
BZ = 0

P₀ | (~AluZero & BZ)

= X

PCSrc

IR

指令流[7-0]

MDR2

MemRead

0

ALU

MemWrite

ALUSrcB = 2

ALUSrcA

ALUop

1 X+0
2 X-Y
3 0+Y
4 0
5 X+Y

Y

积存器

Write Data

RegWrite

= X

MDR

RegDst

= X

1 0

地址

Read Data

Write Data

指令流

= X

IorD

0

MemWrite

= X

ALUop

1 X+0
2 X-Y
3 0+Y
4 0
5 X+Y

X

ALUOut

= X

Y

= X

X

= X

X

= X

X

T3: add # ALUOut=IR[7-0]
T₄ add  # MDR2=Mem[ALUOut]

P₀ | (~AluZero & BZ)
P₀ = 0
BZ = 0

X

MDR2

Instruction[7-0]

IRWrite

IR

MDR

MemWrite

address

Read Data

Write Data

MemRead

=1

=0

IorD

Read Data

Write Data

RegWrite

RegDst

1 0

Accumulator

Write Data

Read Data

ALU

ALUsrcB = X

ALUsrcA

ALUop

1 X+0
2 X-Y
3 0+Y
4 0
5 X+Y

ALU Out

Y

X

0 1

2

0

1

= X

= X

= X

= X

= X

= X
\[
T_5 \quad \text{add} \quad \# \quad \text{ALUOut} = \text{MDR} + A
\]
The diagram illustrates the operations and data flow for the RISCEEE 4 Architecture, specifically the instruction T6: `add # A=ALUOut`.

Key components and operations include:
- **ALU:** Performs arithmetic and logic operations.
- **MDR:** Memory Data Register.
- **MDR2:** Additional Memory Data Register.
- **IR:** Instruction Register.
- **IRWrite:** Writes to the Instruction Register.
- **MemRead:** Reads from memory.
- **MemWrite:** Writes to memory.
- **P0:** Program Counter (PC) Address.
- **BZ:** Branch Zero.
- **ALUsrcA, ALUsrcB:** Source operands for the ALU.
- **ALUop:** ALU operation codes: 1: X+0, 2: X-Y, 3: 0+Y, 4: 0, 5: X+Y.
- **RegWrite:** Writes to the accumulator.
- **RegDst:** Dst register.

The diagram shows the logical flow and control signals for the instruction execution, including conditions like `P0 | (~AluZero & BZ)`, and the final result `ALUOut`.
T₃  bne  # if(A!=0) { PC=IR[7-0] }

P₀ = 0  
BZ = 1  

= X  
IorD  

= 0  
MemRead  

MDR2  

= X  
RegWrite  

= X  
RegDst  

= X  
MDR  

= = 0  
MemWrite  

= 0  
PC  

= = = 0  
Read Data  
Write Data  

address  

= 0  
IR  

= 1  
IRWrite  

= 2  
PCsrc  

= 1  
ALUsrcA  

= 1  
ALUsrcB  

= 1  
ALU  

= 0  
ALUop  

1 X+0  
2 X-Y  
3 0+Y  
4 0  
5 X+Y  

RISCEE 4 Architecture