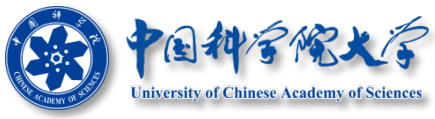
课程编号180086085404P2008H 2024-2025学年秋季学期

计 算 机 体 系 结 构

第 16讲 作业习题讲解（上）

主讲教师：高婉铃

2024年12月24日



第一次作业

- HW1(1)
- HW1(2)
- HW1(3)
- HW1(4)

HW1（1）

Consider the 32-bit hexadecimal number 0xcafe2b3a.

1. What is the binary representation of this number in *little endian* format? Please clearly mark the bytes and number them from low (0) to high (3).

2. What is the binary representation of this number in *big endian* format? Please clearly mark the bytes and number them from low (0) to high (3).

Word:	103	102	101	100
	MSB FF	FF	00	01 LSB
	100	101	102	103

little endian word 100#

big endian word 100#

小端方式（Little Endian）：LSB所在的地址是数的地址；
或者说，字地址或数的地址上存放数据的LSB

大端方式（Big Endian）：MSB所在的地址是数的地址；
或者说，字地址或数的地址上存放数据的MSB

小端方式 little endian format

3a	2b	fe	ca
0	1	2	3

大端方式 big endian format

ca	fe	2b	3a
0	1	2	3

HW1（2）

1.16 [10/20/20/20/25] <1.10> 在实现一个应用程序的并行化时，理想加速比应当等于处理器的个数。但它受到两个因素的限制：可并行化的应用程序百分比和通信成本。Amdahl 定律考虑了前者，但没有考虑后者。

a. [10] <1.10> 如果应用程序的 80%可以并行化，N 个处理器的加速比为多少？（忽略通信成本。）

b. [20] <1.10> 如果每增加一个处理器，通信开销为原执行时间的 0.5%，则 8 个处理器的加速比为多少？

c. [20] <1.10> 如果处理器数目每增加一倍，通信开销增加原执行时间的 0.5%，则 8 个处理器的加速比为多少？

d. [20] <1.10> 如果处理器数目每增加一倍，通信开销增加原执行时间的 0.5%，则 N 个处理器的加速比为多少？

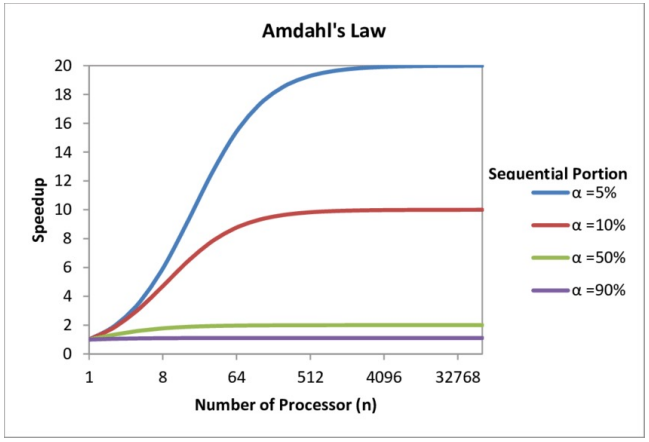
e. [25] <1.10> 写出求解这一问题的一般公式：如果一个应用程序 P%的原执行时间可以并行化，并且处理器数目每增加一倍，通信成本增加原执行时间的 0.5%，则达到最高加速比的处理器数目为多少？

有哪些定律（曾经）统治或影响着计算机及集成电路的发展？

❑ 阿姆达尔（Amdahl）定律：关注短板

- 通过使用某种较快的执行方式所获得的性能的提高，受可使用这种较快执行方式的时间所占的百分比例的限制
- 通过更快的处理器来获得加速是由慢的系统组件所限制

Gene M. Amdahl
Key developer of
IBM System/360
1922/11/16-2015/11/10



$$ExTime_{new} = ExTime_{old} \times \left[(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}} \right]$$

$$Speedup_{overall} = \frac{ExTime_{old}}{ExTime_{new}} = \frac{1}{(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}}$$

HW1（2）

1.16 [10/20/20/20/25] <1.10> 在实现一个应用程序的并行化时，理想加速比应当等于处理器的个数。但它受到两个因素的限制：可并行化的应用程序百分比和通信成本。Amdahl 定律考虑了前者，但没有考虑后者。

a. [10] <1.10> 如果应用程序的 80%可以并行化，N 个处理器的加速比为多少？（忽略通信成本。）

b. [20] <1.10> 如果每增加一个处理器，通信开销为原执行时间的 0.5%，则 8 个处理器的加速比为多少？

c. [20] <1.10> 如果处理器数目每增加一倍，通信开销增加原执行时间的 0.5%，则 8 个处理器的加速比为多少？

d. [20] <1.10> 如果处理器数目每增加一倍，通信开销增加原执行时间的 0.5%，则 N 个处理器的加速比为多少？

e. [25] <1.10> 写出求解这一问题的一般公式：如果一个应用程序 P%的原执行时间可以并行化，并且处理器数目每增加一倍，通信成本增加原执行时间的 0.5%，则达到最高加速比的处理器数目为多少？

$$\text{ExTime}_{new} = \text{ExTime}_{old} \times \left[(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}} \right]$$

$$\text{Speedup}_{overall} = \frac{\text{ExTime}_{old}}{\text{ExTime}_{new}} = \frac{1}{(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}} + \text{通信开销}}$$

HW1（2）

a. [10]<1.10> 如果应用程序的 80%可以并行化，N 个处理器的加速比为多少？（忽略通信成本。）

$$1/(.2 + .8/N)$$

b. [20]<1.10> 如果每增加一个处理器，通信开销为原执行时间的 0.5%，则 8 个处理器的加速比为多少？

通信开销 = 0.005 * 处理器数量

$$1/(.2 + 8 \times 0.005 + 0.8/8) = 2.94$$

c. [20]<1.10> 如果处理器数目每增加一倍，通信开销增加原执行时间的 0.5%，则 8 个处理器的加速比为多少？

通信开销 = $\log_2$ (处理器数量) * 0.005

$$1/(.2 + 3 \times 0.005 + 0.8/8) = 3.17$$

$$\text{ExTime}_{new} = \text{ExTime}_{old} \times \left[(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}} \right]$$

$$\text{Speedup}_{overall} = \frac{\text{ExTime}_{old}}{\text{ExTime}_{new}} = \frac{1}{(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}} + \text{通信开销}}$$

HW1（2）

d. [20] <1.10> 如果处理器数目每增加一倍，通信开销增加原执行时间的 0.5%，则 N 个处理器的加速比为多少？

通信开销 = $\log_2$ (处理器数量) * 0.005

$$1/(0.2 + log_2 N * 0.005 + 0.8/N)$$

e. [25] <1.10> 写出求解这一问题的一般公式：如果一个应用程序 P%的原执行时间可以并行化，并且处理器数目每增加一倍，通信成本增加原执行时间的 0.5%，则达到最高加速比的处理器数目为多少？

$$\frac{d}{dN}\left(\frac{1}{(1-P\%)+\frac{P\%}{N}+\log_2 N \times 0.5\%}\right)$$

$$\frac{d}{dN}\left(\frac{1}{f(N)}\right) = -\frac{f'(N)}{(f(N))^2}$$

对 $\frac{P\%}{N}$ 的导数是 $-\frac{P\%}{N^2}$。
对 $\log_2 N \times 0.5\%$ 的导数是 $0.5\% \cdot \frac{1}{N \ln 2}$

$$(\log_2 N)' = \frac{1}{N \ln 2}$$

$$N = 2P \ln 2 = \frac{2P}{\log_2 e}$$

作业1（3）

C.3 [5/15/10/10] <C.2> 我们首先考虑一个采用单周期实现的计算机。在按功能分割流水级时，这些流水级需要的时间不一定相同。原机器的时钟周期时间为 7 ns。在流水线被分割之后，测得的时间数据为：IF，1 ns；ID，1.5 ns；EX，1 ns；MEM，2 ns；WB，1.5 ns。流水线寄存器延迟为 0.1 ns。

a. [5] <C.2> 5 级流水化机器的时钟周期时间为多少？
b. [15] <C.2> 如果每 4 条指令有一次停顿，则新机器的 CPI 为多少？
c. [10] <C.2> 流水化机器相对于单周期机器的加速比为多少？
d. [10] <C.2> 如果流水化机器有无限个流水级，那它相对于单周期机器的加速比为多少？

流水化的时钟周期由流水线中**最长的阶段延迟**决定，同时需要考虑流水线寄存器的延迟（0.1ns）

a. 最长阶段是 **MEM** 阶段，需要 2ns，再加上流水线寄存器的延迟 0.1ns：
时钟周期=2 ns+0.1 ns=2.1 ns

b. 每4条指令有一次停顿，意味着4条指令执行需要5个时钟周期
CPI = 5/4 = 1.25

作业1（3）

C.3 [5/15/10/10] <C.2> 我们首先考虑一个采用单周期实现的计算机。在按功能分割流水级时，这些流水级需要的时间不一定相同。原机器的时钟周期时间为 7 ns。在流水线被分割之后，测得的时间数据为：IF，1 ns；ID，1.5 ns；EX，1 ns；MEM，2 ns；WB，1.5 ns。流水线寄存器延迟为 0.1 ns。

a. [5] <C.2> 5 级流水化机器的时钟周期时间为多少？

b. [15] <C.2> 如果每 4 条指令有一次停顿，则新机器的 CPI 为多少？

c. [10] <C.2> 流水化机器相对于单周期机器的加速比为多少？

d. [10] <C.2> 如果流水化机器有无限个流水级，那它相对于单周期机器的加速比为多少？

$$Speedup = \frac{Execution\ Time_{single-cycle}}{Execution\ Time_{pipelining}} = \frac{InstCount_{single-cycle} \times CPI_{single-cycle} \times CycleTime_{single-cycle}}{InstCount_{pipelining} \times CPI_{pipelining} \times CycleTime_{pipelining}}$$

c. 单周期机器的时钟周期时间为 7ns，每条指令需要 1 个周期，
流水化执行时间=CPI×流水化时钟周期=1.25×2.1ns=2.625ns
加速比 = 7/2.625=2.67

作业1（3）

C.3 [5/15/10/10] <C.2> 我们首先考虑一个采用单周期实现的计算机。在按功能分割流水级时，这些流水级需要的时间不一定相同。原机器的时钟周期时间为 7 ns。在流水线被分割之后，测得的时间数据为：IF，1 ns；ID，1.5 ns；EX，1 ns；MEM，2 ns；WB，1.5 ns。流水线寄存器延迟为 0.1 ns。

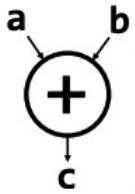
a. [5] <C.2> 5 级流水化机器的时钟周期时间为多少？
b. [15] <C.2> 如果每 4 条指令有一次停顿，则新机器的 CPI 为多少？
c. [10] <C.2> 流水化机器相对于单周期机器的加速比为多少？
d. [10] <C.2> 如果流水化机器有无限个流水级，那它相对于单周期机器的加速比为多少？

$$Speedup = \frac{Execution\ Time_{single-cycle}}{Execution\ Time_{pipelining}} = \frac{InstCount_{single-cycle} \times CPI_{single-cycle} \times CycleTime_{single-cycle}}{InstCount_{pipelining} \times CPI_{pipelining} \times CycleTime_{pipelining}}$$

d. 考虑无限个流水级，则每个阶段的延迟无限小，趋近于0
理想情况下，CPI=1，CycleTime=0.1ns，则speedup = 7/0.1=70
考虑停顿情况下，CPI=1.25， CycleTime=0.1ns，则speedup = 7/0.125=56

作业1（4）

We often use the "addition node":



to represent the addition of two input tokens. If we think of the tokens as binary numbers, we can model a simple logic circuit using dataflow graphs.[1] Note that a token can be used as an input to only *one* node. If the same value is needed by more than one node, it first should be replicated using one or more copy nodes, and then each copied token can be supplied to one node only.

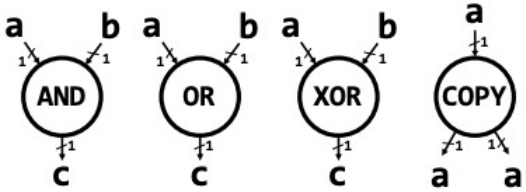
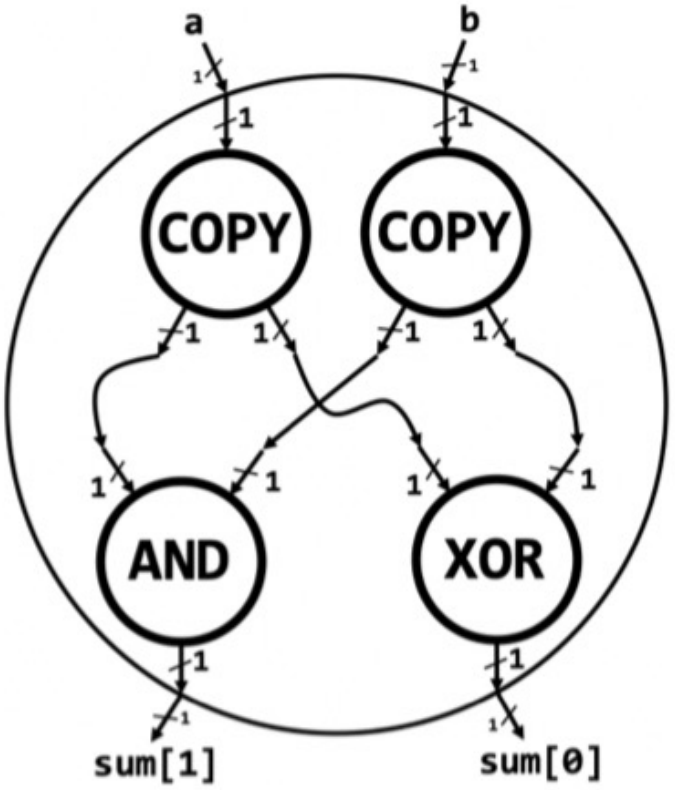


Figure 3: Dataflow nodes of basic bitwise operations allowed in Part (a).

(a) [5 points] Implement the single-bit binary addition of two "1-bit" input tokens `a` and `b` as a dataflow graph using *only* 2-input {AND, OR, XOR} nodes and COPY nodes if necessary (illustrated in Figure 3). Fill in the internal implementation below, where inputs and outputs (labeled with their corresponding bit-widths) have been provided:

作业1（4）



真值表：

a	b	sum[1]	sum[0]
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

a、b、sum[0]：
当两个输入值不相同时，输出为高（1）；
当两个输入值相同时，输出为低（0）

XOR

a、b、sum[1]：
当两个输入均为1时，输出才为1

AND

第二次作业

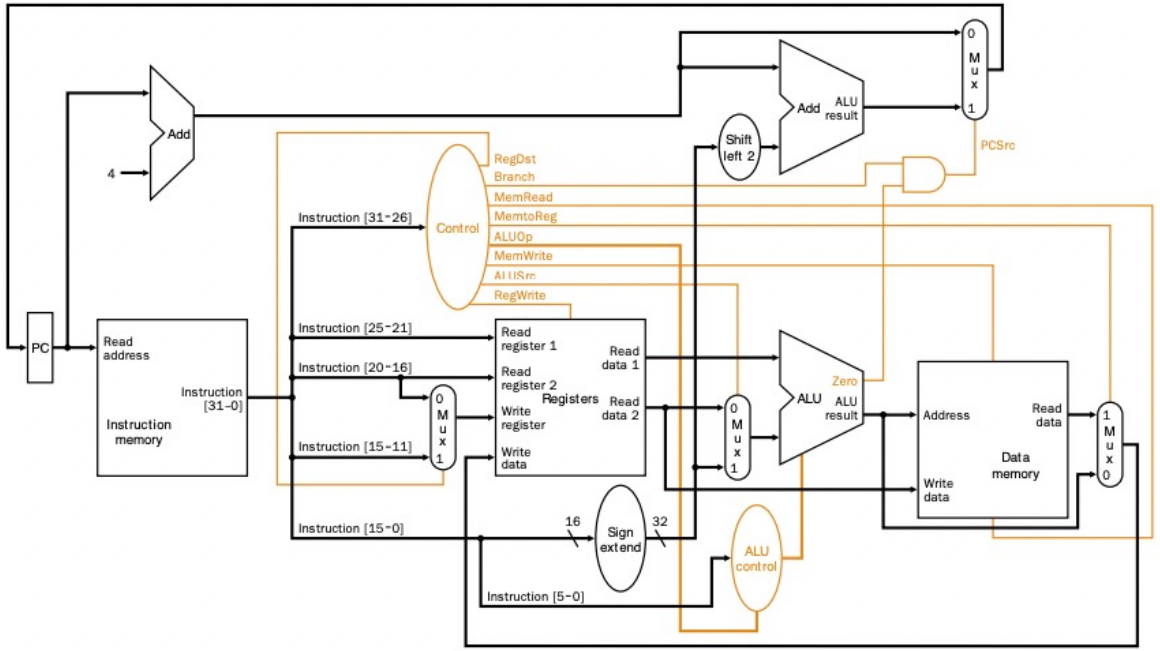
- HW2(1)
- HW2(2)
- HW2(3)
- HW2(4)

作业2（1）

Modify the single-cycle processor datapath to include a version of the `lw` instruction, called `lw2`, that adds two registers to obtain the effective address. The datapath that you will modify is provided below. Your job is to implement the necessary data and control signals to support the new `lw2` instruction, which we define to have the following semantics:

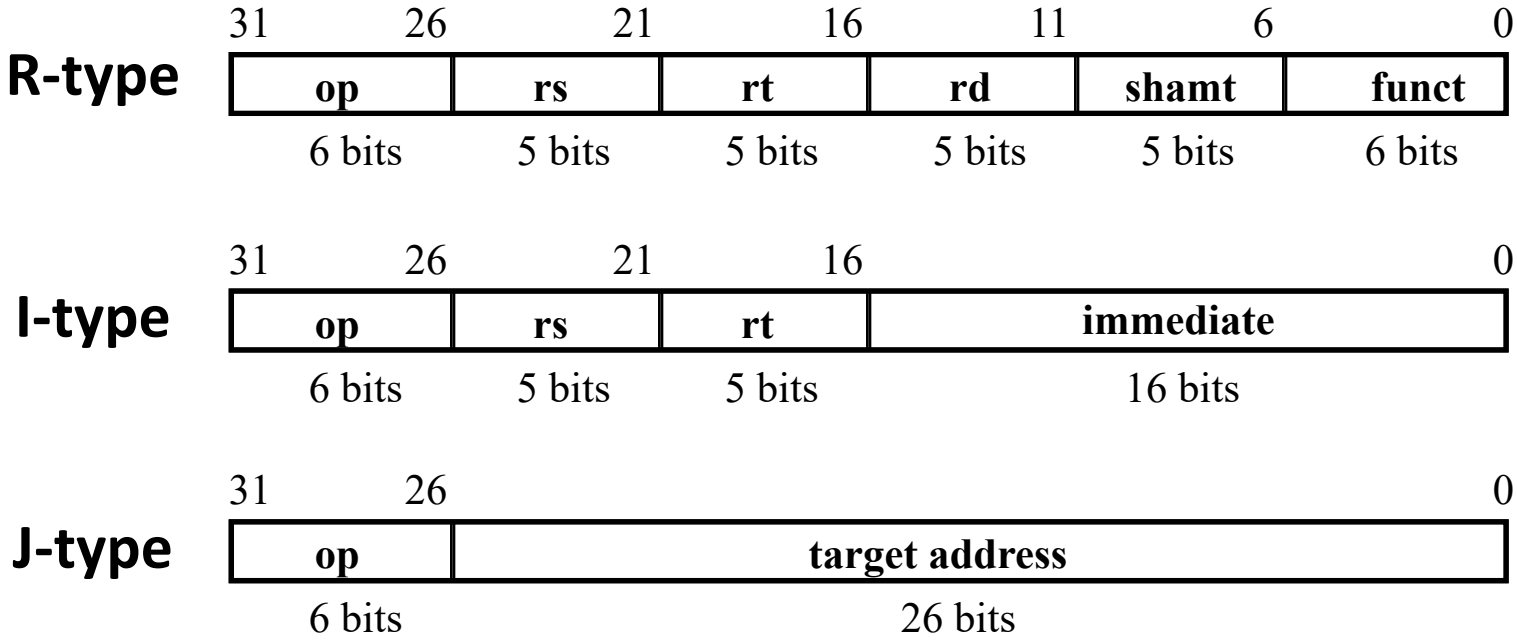
`lw2`: Rd ← Memory[Rs + Rt]
PC ← PC + 4

Add to the datapath any necessary data and control signals (if necessary) to implement the `lw2` instruction. Draw and label all components and wires very clearly (give control signals meaningful names; if selecting a subset of bits from many, specify exactly which bits are selected; and so on).



ALU opcode	Operation
00	Add
01	Subtract
10	Controlled by `funct`
11	Not used

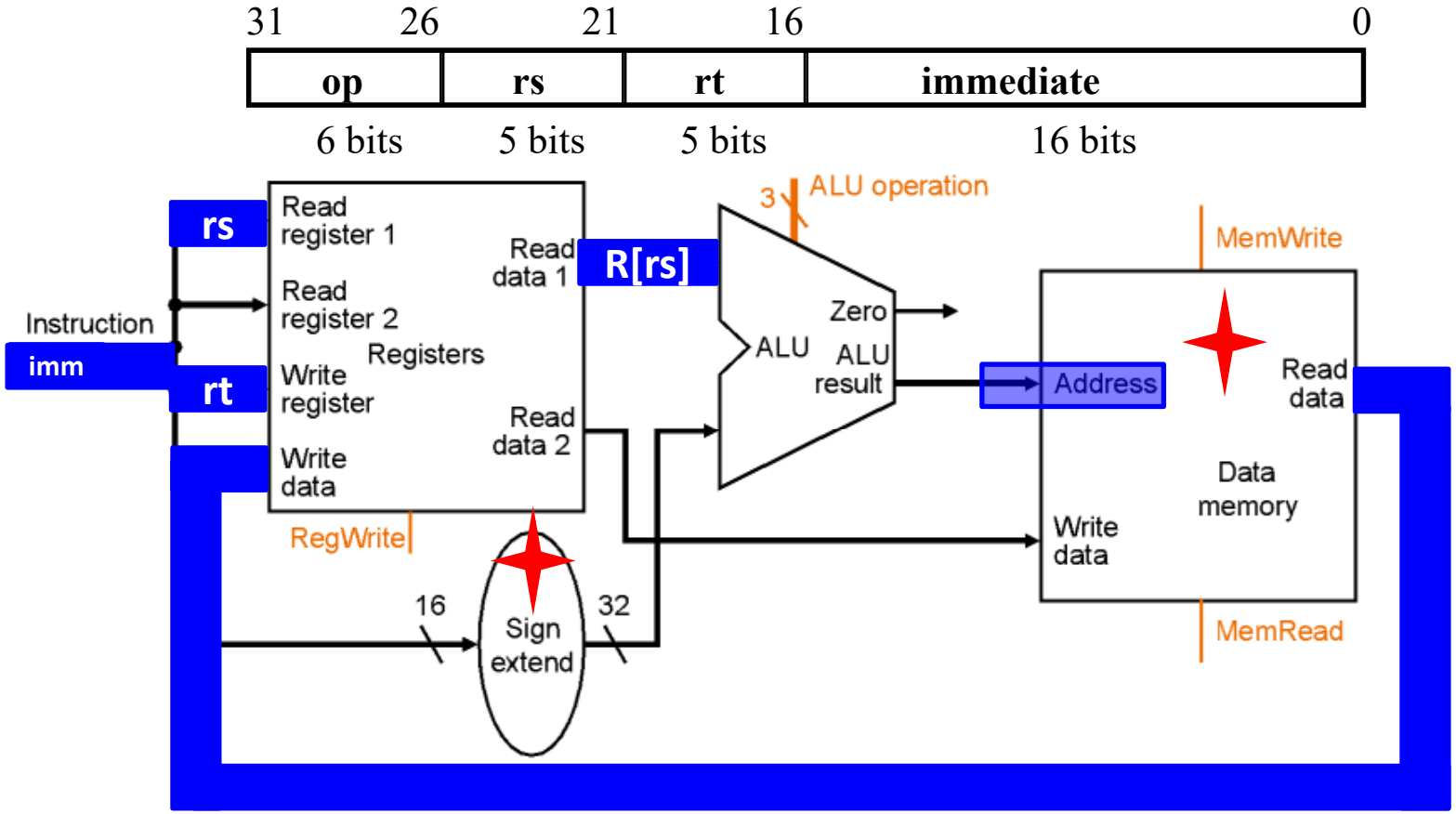
Instruction Formats in MIPS ISA



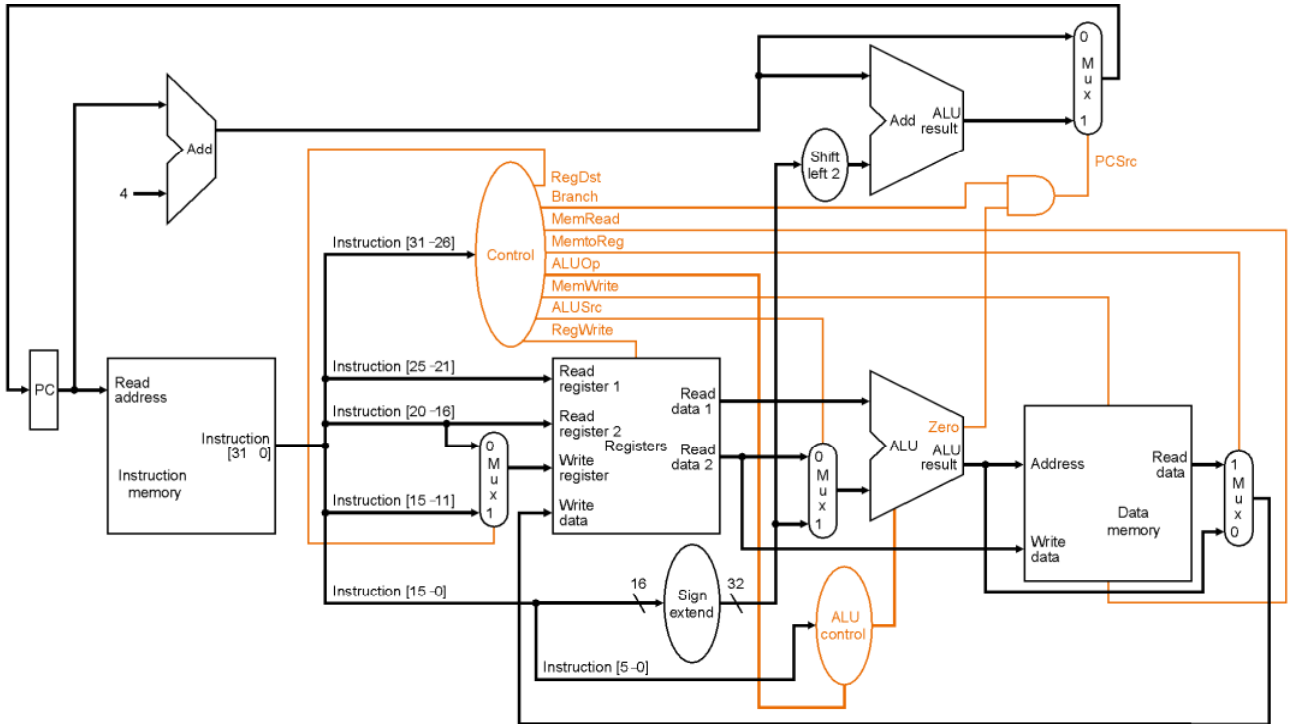
Datapath for Load Operations

❑ R[rt] <- Mem[R[rs] + SignExt[imm16]]

- Example: ***lw rt, rs, imm16***



Controlling the CPU



ALUOp (2 bits):

- 00 – lw,sw
- 01 – beq
- 10 – R-format

Instruction	RegDst	ALUSrc	Memto Reg	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0
R-format	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

ALU control bits

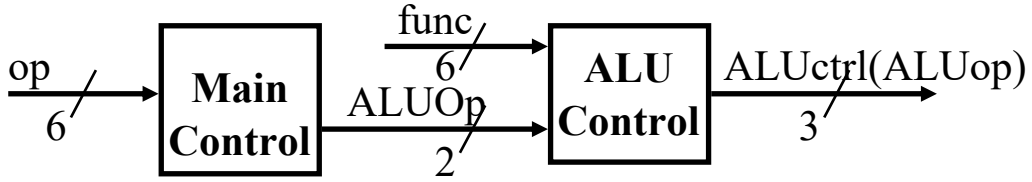
- 5-Function ALU

ALU control input	Function	Operations
000	And	and
001	Or	or
010	Add	add, lw, sw
110	Subtract	sub, beq
111	Slt	slt

Cin或Binv　Oper

- Decoding based on opcode (bits 31-26) and function code (bits 5-0) of the instruction

- ALU doesn't need to know all opcodes--we will summarize opcode with ALUOp (2 bits):
  - 00 – lw,sw　　01 – beq　　10 – R-format



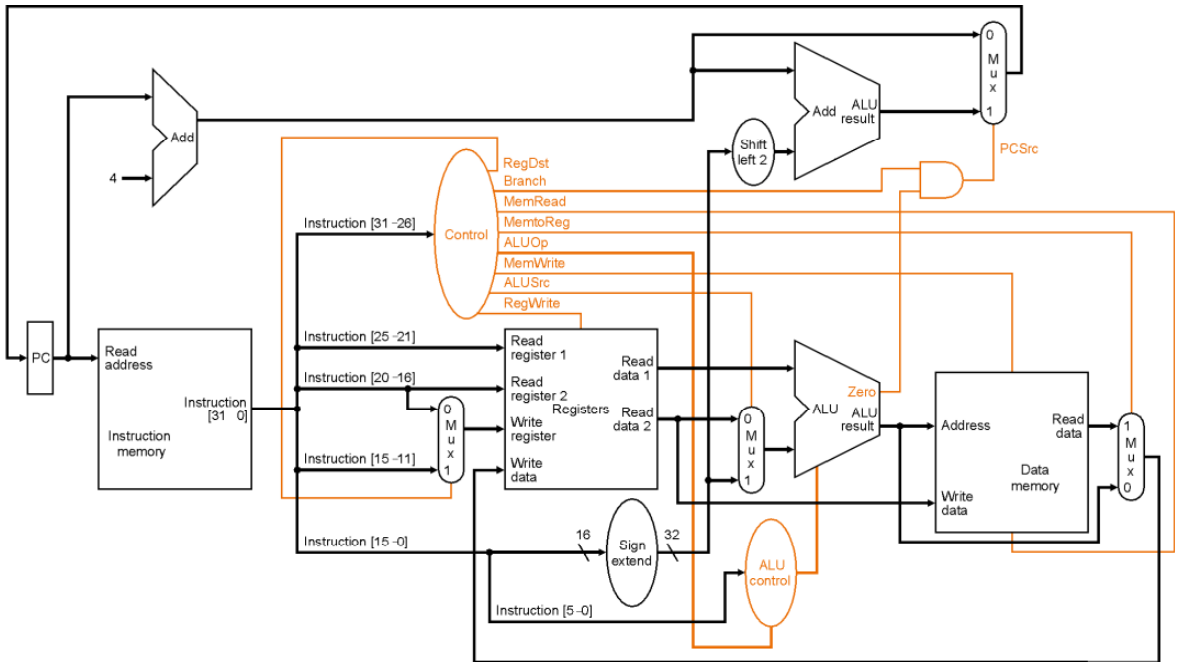
多层解码：减小主控单元规模，提高控制速度

lw2

lw2: $Rd \leftarrow Memory[Rs + Rt]$
$PC \leftarrow PC + 4$

不需要新的组件和线路。主要区别是ALU必须**使用“读取数据2”，而不是符号扩展单元的输出**。新的lw2将是R型，而不是I型。控制信号如下:

RegDst = 1;
ALUScr = 0;
MemtoReg = 1;
RegWrite = 1;
MemRead = 1;
MemWrite = 0;
ALUop = 00;
Branch = 0.



作业2（2）

一个多周期处理器P1执行读内存（Load）指令需要10个周期，写内存（Store）指令需要8个周期，算术指令需要4个周期，分支指令需要4个周期。考虑一个应用程序A，其中20%的指令是Load指令，20%的指令是Store指令，50%的指令是算术指令，10%的指令是分支指令。

（1）在处理器P1上执行A程序，CPI（cycles per instruction）是多少？请写明计算过程。

（2）新的处理器设计P2将P1的时钟频率加倍，但是Load、Store、算术和分支指令的延迟分别增加了2、2、2和1个周期。P1和P2使用相同的编译器，程序A在P1和P2上编译后生成相同数量的指令。请计算在P2处理器上执行应用程序A时，CPI是多少？请写明计算过程。

（3）程序A在哪个处理器上运行更快？快多少倍？请写明计算过程。

（4）假设在处理器P1的芯片中有一些额外的空间，可以容纳额外的硬件。备选方案一是在P1中加入一个更快的分支执行单元，该分支执行单元能将分支指令的延迟缩短4倍；备选方案二是在P1中加入一个更快的存储设备，该存储设备能将内存相关操作的延迟缩短2倍。请结合阿姆达尔定律（Amdahl's Law）说明应该选择哪种备选方案，并写明计算过程。

$CPI = 0.2 \times 10 + 0.2 \times 8 + 0.5 \times 4 + 0.1 \times 4 = 6$

$CPI = 0.2 \times 12 + 0.2 \times 10 + 0.5 \times 6 + 0.1 \times 5 = 7.9$

$Execution\_Time\_P1 = instructions \times CPI_{P1} \times clock\_time = instructions \times 6 \times clock\_time$

$Execution\_Time\_P2 = instructions \times CPI_{P2} \times clock\_time/2 = instructions \times 7.9 \times clock\_time/2$

$Execution\_Time\_P1 / Execution\_Time\_P2 = 1.52$

程序A包含20%的Load指令，20%的Store指令，共计40%访存指令；包含10%的分支指令。

根据阿姆达尔定律：

$Speedup_{branch} = 1/((1-0.1)+0.1/4)=1.08$

$Speedup_{memory} = 1/((1-0.4)+0.4/2)=1.25$

因此，应选择备选方案二，加入更快的存储设备。

作业2（3）

- 假设我们测试两个处理器 A 和 B 在基准程序上的性能。我们发现每个处理器的情况如下：
  - 处理器 A 的 CPI 为 2，每秒执行 40 亿条指令。
  - 处理器 B 的 CPI 为 1，每秒执行 80 亿条指令。
- 哪个处理器在这个程序上的性能更高？请回答并说明理由。
- 注：CPI 代表每条指令的周期数。

无法判断。
虽然题目提供了CPI（每条指令的周期数）和每秒执行指令数的信息，但这还不足以推断处理器的性能。处理器可能支持不同的指令集架构，在这种情况下，基准测试程序将被编译成不同的汇编代码。其中一个处理器每秒执行的指令数更多，并不一定意味着该处理器在这个程序上具有更高的性能。

效率评价指标1：MIPS

- **MIPS：Million Instructions Per Second**
  - 优点：从指令执行的角度，衡量计算机性能
  - 缺点？
    - **体系结构相关指标：不同体系结构的ISA差异，导致架构A上的1 MIPS和架构B上的1 MIPS并非执行和完成相同的工作**

- **IPC： Instructions Per Cycle**
  - IPC 和 MIPS的异同是什么？
    - **IPC从微观角度（cycle）衡量系统性能**
    - **MIPS从宏观角度（时间）衡量系统性能**

效率评价指标2： FLOPS

- FLOPs **(Floating-point Operations)**
  - 64位的浮点乘、加操作
- 测试工具：Linpack
  - 采用主元高斯消去法求解双精度稠密线性代数方程组(AX=b)，结果以每秒浮点运算次数（FLOPS）表示

Operation type	Operation counts
Addition	328 350
Multiplication	333 300
Reciprocal	99
Absolute value	5364
Comparison	4950
Comparison with zero	5247

$(2*n^3)/3+2*n^2+O(n)$. n=100

- 实测峰值 & 理论峰值

Q & A ?


中国科学院大学
University of Chinese Academy of Sciences
ICT
中科院计算所
INSTITUTE OF COMPUTING TECHNOLOGY,CAS