计算机组成
(2022秋)

计算机组成课程组
（刘旭东、高小鹏、肖利民、栾钟治、万寒）
北京航空航天大学计算机学院中德所
栾钟治

1

习题3——主存储器

- 已发布
  - Spoc平台
- 10月21日截止
  - 23:55
- 在sopc提交
  - 电子版，可手写

2

习题4——汇编语言

- 已发布
  - Spoc平台
- 10月28日截止
  - 23:55
- 在sopc提交
  - 电子版，可手写

3

回顾：典型指令系统

- MIPS指令系统
  - 通用寄存器（个数更多），结构规整，指令功能简单，指令格式和寻址方式简洁
- CISC/RISC
  - 指令集体系结构设计的哲学，软件和硬件的划分

4

回顾：CPU的功能与组成

- CPU的功能：控制指令执行
  - 指令的基本操作
- 指令周期（一般性概念）：CPU从指令存储器中读出并执行指令功能的全部时间称为指令周期。包括：
  - 取指周期，取数周期，执行周期
- CPU所需的功能部件
  - 取指令：从存储器中读出指令和分析指令（译码）
    - 指令地址部件，指令寄存部件，译码部件
  - 执行指令：实现指令应该具有的操作功能（包括取数和执行）
    - 执行部件，控制信号逻辑部件

1.1 CPU的功能与组成

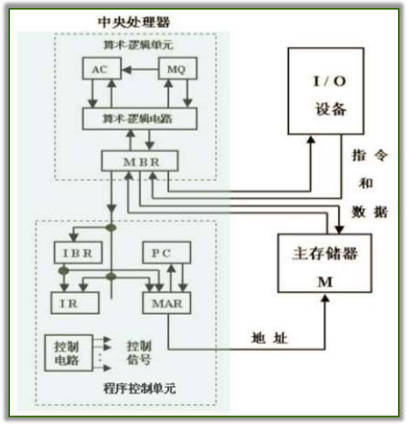
- CPU的组成
  - 执行单元（数据通路，datapath）
    - 运算单元：算术逻辑运算单元（ALU）
    - 寄存器：通用寄存器组（GPRs），标志寄存器（FR，又称程序状态字PSW），临时寄存器（TR）
  - 控制单元（控制器，control）：
    - 指令地址部件：程序计数器（PC — Program Counter）
    - 指令寄存部件：指令寄存器（IR — Instruction Register）
    - 译码部件：指令译码器（ID — Instruction Decoder）
    - 微操作控制信号产生部件：产生计算机其他部件所需要的所有微操作控制信号，有组合逻辑和微程序等实现方式。
    - 时序部件：产生时序信号

1.1 CPU的功能与组成

- 数据通路
  - 指令执行过程中，指令数据流所经过的部件和路径总称，用以实现数据的传送、处理和存储等功能，是指令的执行部件。
  - 构成
    - 组合逻辑元件（操作元件）：ALU、译码器、多路选择器等
    - 存储元件（状态元件）：存储器、寄存器等
  - 部件间连接方式
    - 总线连接方式（CPU内部总线）
    - 分散连接方式
- 控制器
  - 对指令进行译码并生成指令执行所需的控制信号，以实现对数据通路中各件的功能控制，以及相应路径的开关控制等，是指令的控制部件。

1.1 CPU的功能与组成

- 简单的数据通路示例
  - 取指令路径
    - PC→MAR
    - Read Mem.
    - M→MBR→IBR→IR
  - 取操作数的路径
    - 操作数地址→MAR
    - Read Mem.
    - M→MBR→ALU
  - 运算结果保存路径
    - ALU结果→MBR
    - 结果地址→MAR
    - Write Mem.



早期累加器型数据通路

1.1 CPU的功能与组成

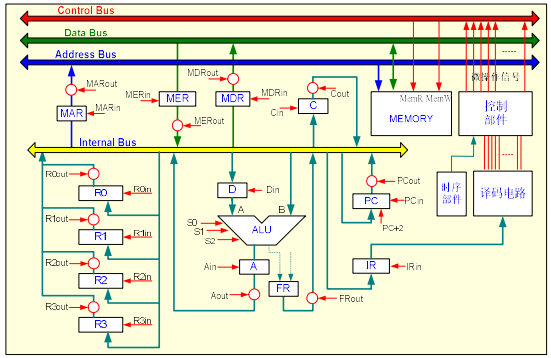
❖单总线数据通路示例

➢取指令路径
- PC→IB→MAR
- MemR
- M→MER→IB→IR

➢取操作数的路径
- 地址→IB→MAR
- MemR
- M→MER→IB→ALU

➢运算结果保存路径
- ALU结果→IB→MDR
- 结果地址→IB→MAR
- MemW

单总线数据通路



9

1.1 CPU的功能与组成

❖指令功能的形式化描述：RTL（Register Transfer Language，寄存器传送语言）

➢ ← ：数据传送方向；

➢ R[a]：寄存器 a;

➢ M[a]：主存中地址为a的单元；

➢ PC ：程序计数器

➢ f(data)：表示对数据data进行f操作

❖示例

➢R[c] ← R[a] + R[b] // 寄存器a加寄存器b的结果送寄存器c

➢R[c] ← R[a] *op* R[b] // 寄存器a与寄存器b进行*op*运算结果送寄存器c

➢Signext(imm16) // 对数imm16进行Signext（符号扩展）运算

➢R[a] ← M[b] // 取数操作，读取主存单元b的数据传送至寄存器a

➢M[a] ← R[b] // 存数操作，将寄存器b中的数据写入主存单元a中

10

1.2 处理器设计的一般方法

❖ 设计步骤

1. 分析指令系统需求：包括指令格式、指令类型、每种指令的功能、寻址方式等；
2. 数据通路构建
   - ① 根据指令需求选择数据通路部件，如PC、ALU、寄存器堆、指令/数据存储器、多路开关等等；
   - ② 根据指令执行流程构建每种类型指令的数据通路；
   - ③ 对所有类型指令执行数据通路综合形成综合数据通路。
3. 控制器设计
   - ① 确定控制器时序控制方式（单周期、或多周期或其他）
   - ② 根据每种类型指令执行流程，确定该指令执行时各个数据通路部件所需要的控制信号与相应状态、条件；
   - ③ 对控制信号进行综合以得到每个控制信号的逻辑方程；
   - ④ 逻辑电路实现各个控制信号。

11

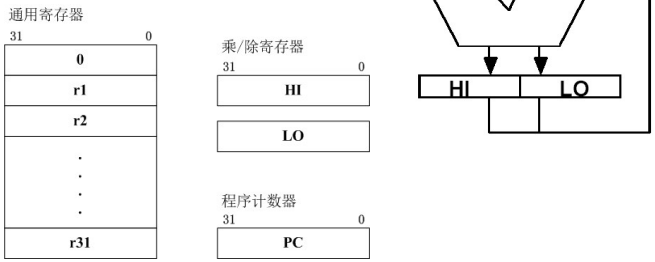
第六讲 MIPS处理器设计

12

2.1 MIPS模型机指令集

❖MIPS 寄存器结构

- 32位虚拟地址空间
- 32个32位GPRs（通用寄存器）
- 32个32 位FPRs（浮点数寄存器）
- HI, LO, PC



13

2.1 MIPS模型机指令集

MIPS 模型机寄存器使用约定

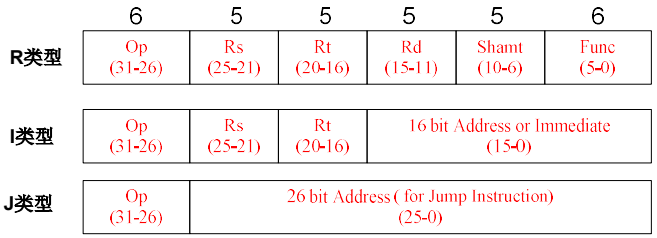
寄存器编号	寄存器名称	用途
0	$zero	常数0
1	$at	保留给汇编器使用
2 ~ 3	$v0 ~ $v1	结果值和表达式求值
4 ~ 7	$a0 ~ $a3	参数
8 ~ 15	$t0 ~ $t7	临时变量
16 ~ 23	$s0 ~ $s7	数据寄存器
24 ~ 25	$t8 ~ $t9	其他临时变量
26 ~ 27	$k0 ~ $k1	保留给操作系统使用
28	$gp	全局指针
29	$sp	栈指针
30	$fp	帧指针
31	$ra	返回地址

14

2.1 MIPS模型机指令集

❖MIPS 指令格式

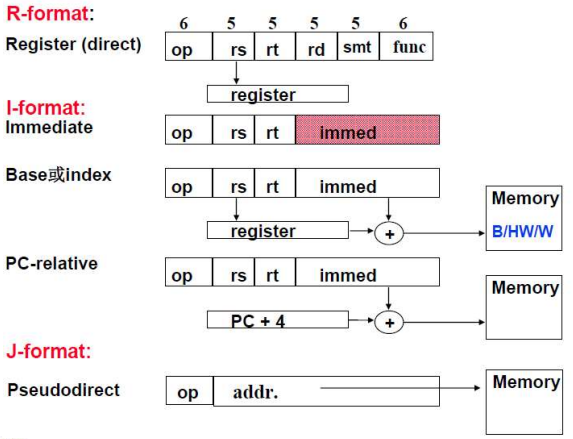
- Op： 6 bits, Opcdoe
- Rs: 5 bits, The first register source operand
- Rt: 5 bits, The second register source operand
- Rd: 5 bits, The register destination operand
- Shamt: 5 bits, Shift amount (shift instruction)
- Func: 6 bits, function code (another Opcode)



15

2.1 MIPS模型机指令集

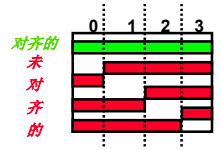
❖MIPS 寻址方式



16

一些细节

- 字对齐
  - Alignment(字对齐)：对象的起始地址一定要是字长的整数倍



地址的最后一个16进制数:
0, 4, 8, or C_{hex}
1, 5, 9, or D_{hex}
2, 6, A, or E_{hex}
3, 7, B, or F_{hex}

  - 字vs.字节vs.双字节
  - 未对齐？
- 填充
  - 符号扩展填充vs.无符号填充
  - load vs. store
  - 逻辑和算术移位
- 溢出
  - 检测溢出vs.不检测溢出

2.1 MIPS模型机指令集

模型机指令集（8条指令）

指令格式	指令	功能	说明
R类型	add rd, rs, rt	R[rd]← R[rs] + R[rt]	加运算：寄存器 rs 和寄存器 rt 相加，结果送寄存器 rd
	sub rd, rs, rt	R[rd]← R[rs] - R[rt]	减运算：寄存器 rs 和寄存器 rt 相减，结果送寄存器 rd
	and rd, rs, rt	R[rd]← R[rs] & R[rt]	与运算：寄存器 rs 和寄存器 rt 按位与，结果送寄存器 rd
	or rd, rs, rt	R[rd]← R[rs] | R[rt]	或运算：寄存器 rs 和寄存器 rt 按位或，结果送寄存器 rd
I类型	lw rt, rs, imm16	Add = R[rs] +Signext(imm16) R[rt] ← M[Add]	取字：寄存器 rs 和立即数 imm16（符号扩展至 32 位）相加得到内存地址，从内存该地址单元读取数据送 rt
	sw rt, rs, imm16	Add = R[rs] +Signext(imm16) M[Add] ← R[rt]	存字：寄存器 rs 和立即数 imm16（符号扩展至 32 位）相加得到内存地址，寄存器 rt 数据写入内存该地址单元
	beq rs, rt, imm16	If (R[rs] – R[rt] =0) then PC ← PC + Signext(imm16)<<2	分支：如果寄存器 rs 与 rt 相等，则转移（imm16 符号扩展至 32 位），否则顺序执行。（取指令后，PC+4）
J类型	j target	PC(31:2)←PC(31:28) || target(25:0)	跳转：当前 PC 的高 4 位与 target（26 位）拼接成 30 位目标地址送 PC（31:2）。（取指令后，PC+4）。

2.1 MIPS模型机指令集

模型机指令编码

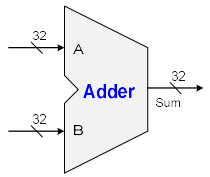
R 类型格式	OP (31 ~ 26)	Rs (25 ~ 21)	Rt (20 ~ 16)	Rd (15 ~ 11)	Shamt (10 ~ 6)	Funct (5 ~ 0)
add rd, rs, rt	000000	rs	rt	rd	XXXXX	100000
sub rd, rs, rt	000000	rs	rt	rd	XXXXX	100010
and rd, rs, rt	000000	rs	rt	rd	XXXXX	100100
or rd, rs, rt	000000	rs	rt	rd	XXXXX	100101

I 类型格式	OP (31 ~ 26)	Rs (25 ~ 21)	Rt (20 ~ 16)	16 bits immediate or address (15 ~ 0)
lw rt, rs, imm16	100011	rs	rt	imm16
sw rt, rs, imm16	101011	rs	rt	imm16
beq rs, rt, imm16	000100	rs	rt	imm16

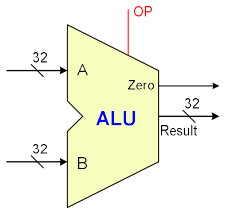
J 类型格式	OP (31 ~ 26)	26 bits address
j target	000010	target

2.2 数据通路部件

- 组合部件：
  - 加法器（Adder）、算术逻辑运算单元（ALU）



输入（32位）		输出（32位）
A	B	Sum=A+B

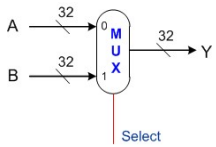


数据输入（32位）		控制输入OP（4位）	输出
A	B	0000	Result = A&B
A	B	0001	Result = A|B
A	B	0010	Result = A+B
A	B	0110	Result = A-B
A	B	0111	If A=B then Result=1

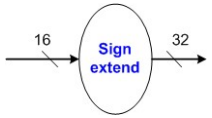
2.2 数据通路部件

❖组合部件

- ➢多路选择器（Mux）、符号扩展器（Signext）
- ➢多路选择器有二选一，三选一，四选一等。



输入（32位）		Select输入	输出（32位）
A	B	无效（低电平）	Y=A
A	B	有效（高电平）	Y=B



16位二进制数按符号位扩展至32位

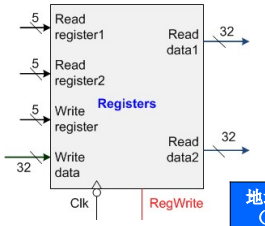
如： Signext(8000H) = FFFF8000H

Signext(7800H) = 00007800H

2.2 数据通路部件

❖寄存器堆（32个寄存器）

- ➢两个32位输出端口
- ➢一个32位输入端口



寄存器堆读操作（输出）

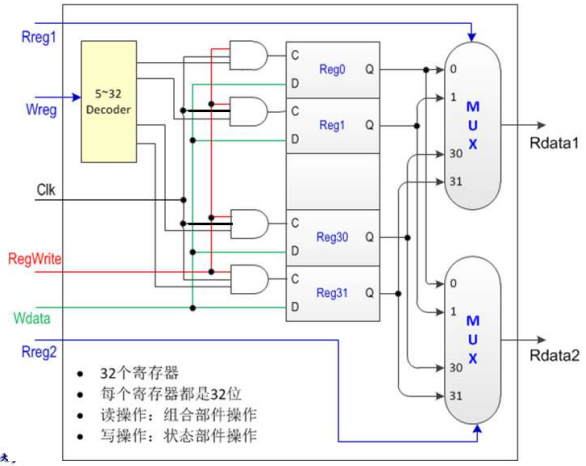
地址输入（5位）		数据输出
Read register1	Read register2	Read data1=R[Readregister1] Read data2=R[Readregister2]

寄存器堆写操作（输入）

地址输入（5位）	数据输入（32位）	控制输入 RegWrite	操作
Write register	Write data	无效	无
Write register	Write data	有效（Clk时钟下跳沿）	R[Writeregister] ← Write data

2.2 数据通路部件

❖寄存器堆内部结构

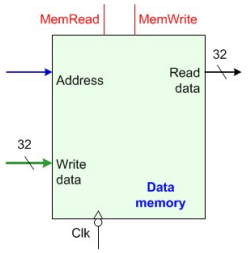


- 32个寄存器
- 每个寄存器都是32位
- 读操作：组合部件操作
- 写操作：状态部件操作

2.2 数据通路部件

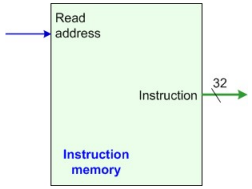
❖数据存储器DM（理想存储器）

- ➢单输入总线：Write data
- ➢单输出总线：Read data
- ➢读： MemRead控制信号有效时，地址线(Address)选择的存储字被放在Read data输出总线上（组合元件操作）
- ➢写：MemWrite控制信号有效且时钟信号Clk下跳沿时，Write data总线上的数据被写入地址选择的存储单元中（状态元件操作）



❖指令存储器IM

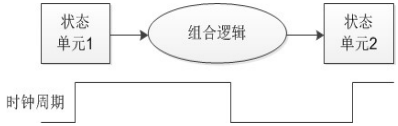
- ➢指令地址(Read address)选择的指令被放在Instruction输出线上（组合元件操作）



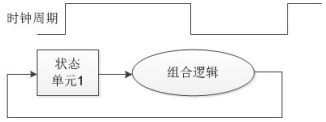
2.3 时钟同步方法

❖时钟同步方法

- 以时钟周期信号为基准，确定数据读出和写入的时刻。
- 采用边沿触发的时钟同步方法，如下跳沿触发，意味着所有状态元件（寄存器、存储器）的数据写入都发生在时钟周期的下跳沿时刻。



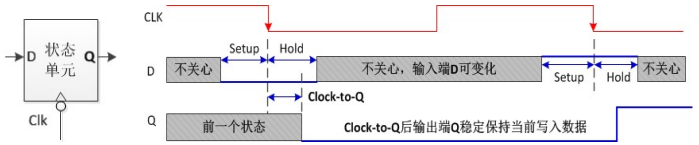
- 组合逻辑操作时钟周期内完成；
- 所有信号在时钟周期内从状态单元1经组合逻辑传送到状态单元2。
- 时钟信号上跳沿同步



- 组合逻辑操作时钟周期内完成；
- 所有信号在时钟周期内从状态单元输出经组合逻辑处理再回到状态单元输入。
- 时钟信号上跳沿同步

2.3 时钟同步方法

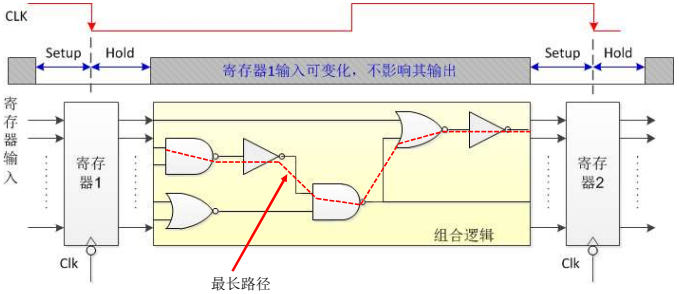
❖状态单元的时序



- 建立时间（Setup Time）：触发时钟边沿之前输入必须稳定的时间；
- 保持时间（Hold Time）：触发时钟边沿之后输入仍需稳定的时间；
- Clock-to-Q：从触发时钟边沿到输出有效的时间。

2.3 时钟同步方法

❖时钟周期



- 数据通路由“ … + 状态元件+ 操作元件(组合电路) + 状态元件+ …” 组成
- 状态元件存储信息，所有操作元件从状态单元接收输入，并将输出写入状态单元中。其输入为前一时钟生成的数据，输出为当前时钟所用的数据
- Cycle Time = Clock-to-Q + Longest Delay Path + Setup + Clock Skew

第六讲 MIPS处理器设计

MIPS模型机指令集

模型机指令集（8条指令）

指令格式	指令	功能	说明
R类型	add rd, rs, rt	R[rd]← R[rs] + R[rt]	加运算：寄存器 rs 和寄存器 rt 相加，结果送寄存器 rd
	sub rd, rs, rt	R[rd]← R[rs] - R[rt]	减运算：寄存器 rs 和寄存器 rt 相减，结果送寄存器 rd
	and rd, rs, rt	R[rd]← R[rs] & R[rt]	与运算：寄存器 rs 和寄存器 rt 按位与，结果送寄存器 rd
	or rd, rs, rt	R[rd]← R[rs] | R[rt]	或运算：寄存器 rs 和寄存器 rt 按位或，结果送寄存器 rd
I类型	lw rt, rs, imm16	Add = R[rs] +Signext(imm16) R[rt] ← M[Add]	取字：寄存器 rs 和立即数 imm16（符号扩展至 32 位）相加得到内存地址，从内存该地址单元读取数据送 rt
	sw rt, rs, imm16	Add = R[rs] +Signext(imm16) M[Add] ← R[rt]	存字：寄存器 rs 和立即数 imm16（符号扩展至 32 位）相加得到内存地址，寄存器 rt 数据写入内存该地址单元
	beq rs, rt, imm16	If (R[rs] – R[rt] =0) then PC ← PC + Signext(imm16)<<2	分支：如果寄存器 rs 与 rt 相等，则转移（imm16 符号扩展至 32 位），否则顺序执行。（取指令后，PC+4）
J类型	j target	PC(31:2)←PC(31:28) || target(25:0)	跳转：当前 PC 的高 4 位与 target（26 位）拼接成 30 位目标地址送 PC（31:2）。（取指令后，PC+4）。

数据通路部件

- 组合部件
  - 加法器（Adder）、算术逻辑运算单元（ALU）
  - 多路选择器（Mux）、符号扩展器（Signext）、左移两位（shift left 2）
- 存储部件
  - 寄存器堆
  - 数据存储器DM
  - 指令存储器IM

3.1 单周期数据通路设计

- 单周期
  - 所有指令执行周期固定为单一时钟周期，CPI=1
- 通路设计考虑
  - 哈佛体系结构：使用指令存储区（IM）和数据存储区（DM）分别保存指令和数据
  - 先为每类指令设计独立的数据通路，然后再考虑数据通路合并。
- 指令执行的共性
  - 根据PC从指令存储器读取指令，取指令后，PC+4；
  - 模型机7条指令在读取寄存器后，都要使用ALU
    - LW/SW（存储访问）指令用ALU计算数据地址
    - ADD/SUB/AND/OR(算术逻辑)指令用ALU完成算术逻辑运算
    - BEQ（分支）指令用ALU进行比较（减法运算）
  - R类型指令除了在ALU中进行的运算不同，其它操作都一样

3.1 单周期数据通路设计

- 分析指令执行步骤，确定数据通路所需部件和部件间连接
  - 模型机指令执行过程一般会分为如下几个步骤：
    - 取指令：根据PC访问指令存储器获得指令，然后PC+4；
    - 读寄存器：根据指令格式读取相应寄存器操作数
    - ALU运算：在ALU完成相应的算术逻辑运算
    - 数据存取：LW/SW指令的数据存储器访问
    - 写寄存器：运算类指令和LW指令要把数据写入寄存器
- 使用数据通路设计表格
  - 表格记录数据通路部件输入端的输入来源
  - 暂不考虑控制信号

指令	Adder		PC	IM Add.	Registers				ALU		DM	
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata

3.1 单周期数据通路设计——取指与PC自增

1. 取指和PC自增数据通路（所有指令）

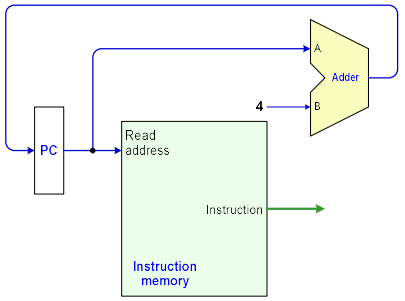
- 功能描述
  - 取指：IM Address ←PC, instruction=IM[PC]
  - PC自增：PC← PC + 4
- 所需部件：PC，Adder（实现PC加4），指令存储器IM

指令	Adder		PC	IM Add.	Registers				ALU		DM	
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata
R型指令	PC	4	Adder	PC								
Lw	PC	4	Adder	PC								
Sw	PC	4	Adder	PC								
Beq	PC	4	Adder	PC								

3.1 单周期数据通路设计——取指与PC自增

1. 取指和PC自增数据通路图

- 功能描述
  - 取指：instruction = IM[PC]
  - PC自增：PC← PC + 4
- 所需部件：PC，Adder（实现PC加4），IM



3.1 单周期数据通路设计——R型指令数据通路

2. R型指令数据通路（add,sub,and,or指令，以add为例）

- add rd, rs, rt
- 功能描述
  - R[rd] ← R[rs] + R[rt]
- 通路部件：寄存器堆、ALU

Op (31-26)	Rs (25-21)	Rt (20-16)	Rd (15-11)	Shamt (10-6)	Func (5-0)

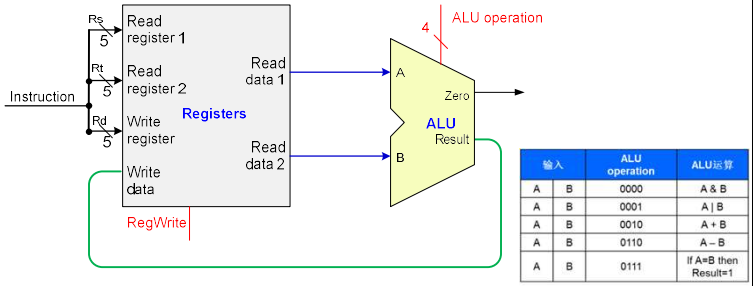
指令	Adder		PC	IM Add.	Registers				ALU		DM	
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata
R型指令	PC	4	Adder	PC	Rs	Rt	Rd	ALU	Rdata1	Rdata2		
Lw	PC	4	Adder	PC								
Sw	PC	4	Adder	PC								
Beq	PC	4	Adder	PC								

3.1 单周期数据通路设计——R型指令数据通路

2. R型指令数据通路（add,sub,and,or指令，以add为例）

- add rd, rs, rt
- 功能描述
  - R[rd] ← R[rs] + R[rt]
- 通路部件：寄存器堆、ALU

Op (31-26)	Rs (25-21)	Rt (20-16)	Rd (15-11)	Shamt (10-6)	Func (5-0)



输入		ALU operation	ALU运算
A	B	0000	A & B
A	B	0001	A | B
A	B	0010	A + B
A	B	0110	A – B
A	B	0111	If A=B then Result=1

3.1 单周期数据通路设计——LW指令数据通路

3. 取数指令（lw）数据通路

- lw rt, rs, imm16

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)

- 功能描述：
  - R[rt] ← DM[R[rs] + Signext(imm16)]
- 通路部件：寄存器堆，ALU，符号扩展单元Signext，数据存储器DM

指令	Adder		PC	IM Add.	Registers				ALU		DM		Sign-ext
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata	
R型指令	PC	4	Adder	PC	Rs	Rt	Rd	ALU	Rdata1	Rdata2			
Lw	PC	4	Adder	PC	**Rs**		**Rt**	**DM**	**Rdata1**	**Sign-ext**	**ALU**		**imm16**
Sw	PC	4	Adder	PC									
Beq	PC	4	Adder	PC									

37

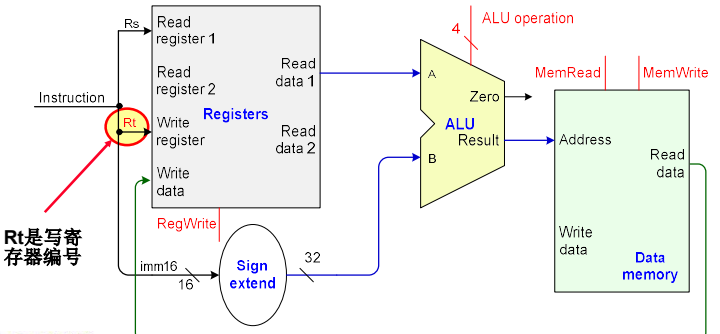
3.1 单周期数据通路设计——LW指令数据通路

3. 取数指令（lw）数据通路

- lw rt, rs, imm16

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)

- 功能描述：
  - R[rt] ← DM[R[rs] + Signext(imm16)]
- 通路部件：寄存器堆，ALU，符号扩展单元Signext，数据存储器DM



38

3.1 单周期数据通路设计——SW指令数据通路

4. 存数指令（sw）数据通路

- sw rt, rs, imm16

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)

- 功能描述：
  - DM[R[rs] + Signext(imm16)] ← R[rt]
- 通路部件：寄存器堆，ALU，符号扩展单元Signext，数据存储器DM

指令	Adder		PC	IM Add.	Registers				ALU		DM		Sign-ext
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata	
R型指令	PC	4	Adder	PC	Rs	Rt	Rd	ALU	Rdata1	Rdata2			
Lw	PC	4	Adder	PC	Rs		Rt	DM	Rdata1	Signext	ALU		imm16
Sw	PC	4	Adder	PC	**Rs**	**Rt**			**Rdata1**	**Sign-ext**	**ALU**	**Rdata2**	**imm16**
Beq	PC	4	Adder	PC									

39

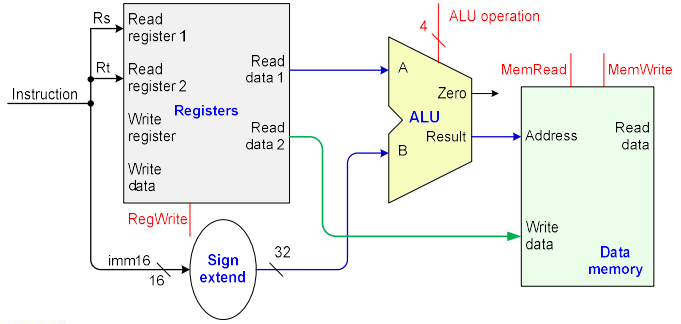
3.1 单周期数据通路设计——SW指令数据通路

4. 存数指令（sw）数据通路

- sw rt, rs, imm16

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)

- 功能描述：
  - DM[R[rs] + Signext(imm16)] ← R[rt]
- 通路部件：寄存器堆，ALU，符号扩展单元Signext，数据存储器DM



40

3.1 单周期数据通路设计——R型指令与访存指令通路合并

5. R型指令与访存指令数据通路合并：增加多路选择Mux

Op (31-26)	Rs (25-21)	Rt (20-16)	Rd (15-11)	Shamt (10-6)	Func (5-0)

R型指令格式

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)

LW/SW指令格式

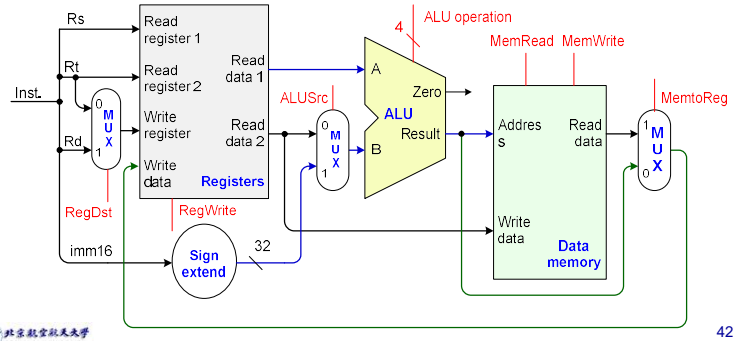
指令	Adder		PC	IM Add.	Registers				ALU		DM		Sign-ext
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata	
R型指令	PC	4	Adder	PC	Rs	Rt	Rd	ALU	Rdata1	Rdata2			
Lw	PC	4	Adder	PC	Rs		Rt	DM	Rdata1	Signext	ALU		imm16
Sw	PC	4	Adder	PC	Rs	Rt			Rdata1	Sign-ext	ALU	Rdata2	imm16
合并	PC	4	Adder	PC	Rs	Rt	Rd | Rt	ALU | DM	Rdata1	Rdata2 | Sign-ext	ALU	Rdata2	imm16

输入端数据源出现多个选择，需要加入多路选择器MUX

3.1 单周期数据通路设计——R型指令与访存指令通路合并

5. R型指令与访存指令数据通路合并

- 增加3个二选一多路选择器MUX
  - 寄存器堆写入端地址选择MUX，选择控制信号 RegDst
  - ALU输入端B数据源选择MUX，选择控制信号 ALUSrc
  - 寄存器堆写入端数据源选择MUX，选择控制信号 MemtoReg



3.1 MIPS的数据通路设计——Beq指令数据通路

6. 分支指令数据通路

- beq rs, rt, imm16

Op (31-26)	Rs (25-21)	Rt (20-16)	16 bit Address or Immediate (15-0)

- 功能描述：
  - If (R[rs] – R[rt] =0) then PC ← (PC + 4) + Signext(imm16)<<2
    else PC ← PC + 4
- 通路部件：寄存器堆，ALU， 增加一加法器Nadd，符号扩展Signext，移位器

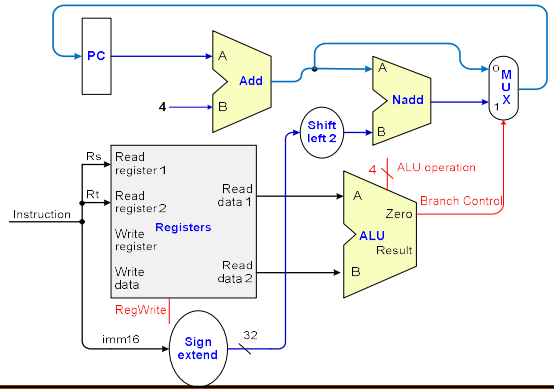
指令	Adder		PC	IM Add.	Registers				ALU		DM		Sign-ext	Nadd	
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata			
R型与访存	PC	4	Adder	PC	Rs	Rt	Rd | Rt	ALU | DM	Rdata1	Rdata2 | Sign-ext	ALU	Rdata2	imm16		
Beq	PC	4	Adder | Nadd	PC	Rs	Rt			Rdata1	Rdata2			imm16	Adder	Shift

需要一个MUX，ALU的判零输出端Zero可直接作为该MUX的选择控制

3.1 MIPS的数据通路设计——Beq指令数据通路

6. 分支指令数据通路

- beq rs, rt, imm16
- 通路部件：寄存器堆，ALU， 增加一Adder，符号扩展Signext，移位器



3.1 单周期数据通路设计

7. MIPS数据通路再合并

➢支持：R类型指令、内存访问指令（lw/sw)、beq指令

指令	Adder		PC	IM Add.	Registers				ALU		DM		Sign-ext	Nadd	
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata			
R型与访存	PC	4	Adder	PC	Rs	Rt	Rd | Rt	ALU | DM	Rdata1	Rdata2 | Sign-ext	ALU	Rdata2	imm16		
Beq	PC	4	Adder | Nadd	PC	Rs	Rt			Rdata1	Rdata2			imm16	Adder	Shift
合并	PC	4	Adder | Nadd	PC	Rs	Rt	Rd | Rt	ALU | DM	Rdata1	Rdata2 | Sign-ext	ALU	Rdata2	imm16	Adder	Shift

需要4个二选一多路选择器MUX

- PC输入端数据源选择MUX，选择控制信号 PCSrc
- 寄存器堆写入端地址选择MUX，选择控制信号 RegDst
- ALU输入端B数据源选择MUX，选择控制信号 ALUSrc
- 寄存器堆写入端数据源选择MUX，选择控制信号 MemtoReg

45

第六讲 MIPS处理器设计

46

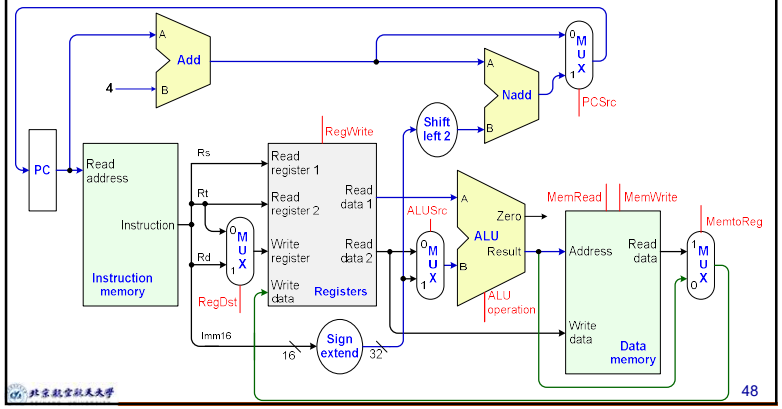
MIPS模型机指令集

模型机指令集（8条指令）

指令格式	指令	功能	说明
R类型	add rd, rs, rt	R[rd]← R[rs] + R[rt]	加运算：寄存器 rs 和寄存器 rt 相加，结果送寄存器 rd
	sub rd, rs, rt	R[rd]← R[rs] - R[rt]	减运算：寄存器 rs 和寄存器 rt 相减，结果送寄存器 rd
	and rd, rs, rt	R[rd]← R[rs] & R[rt]	与运算：寄存器 rs 和寄存器 rt 按位与，结果送寄存器 rd
	or rd, rs, rt	R[rd]← R[rs] | R[rt]	或运算：寄存器 rs 和寄存器 rt 按位或，结果送寄存器 rd
I类型	lw rt, rs, imm16	Add = R[rs] +Signext(imm16) R[rt] ← M[Add]	取字：寄存器 rs 和立即数 imm16（符号扩展至 32 位）相加得到内存地址，从内存该地址单元读取数据送 rt
	sw rt, rs, imm16	Add = R[rs] +Signext(imm16) M[Add] ← R[rt]	存字：寄存器 rs 和立即数 imm16（符号扩展至 32 位）相加得到内存地址，寄存器 rt 数据写入内存该地址单元
	beq rs, rt, imm16	If (R[rs] – R[rt] =0) then PC ← PC + Signext(imm16)<<2	分支：如果寄存器 rs 与 rt 相等，则转移（imm16 符号扩展至 32 位），否则顺序执行。（取指令后，PC+4）
J类型	j target	PC(31:2)←PC(31:28) || target(25:0)	跳转：当前 PC 的高 4 位与 target（26 位）拼接成 30 位目标地址送 PC（31:2）。（取指令后，PC+4）。

47

指令	Adder		PC	IM Add.	Registers				ALU		DM		Sign-ext	Nadd	
	A	B			Reg1	Reg2	Wreg	Wdata	A	B	Add.	Wdata		A	B
合并	PC	4	Adder | Nadd	PC	Rs	Rt	Rd | Rt	ALU | DM	Rdata1	Rdata2 | Sign-ext	ALU	Rdata2	imm16	Adder	Shift



48