第二讲 数据通路

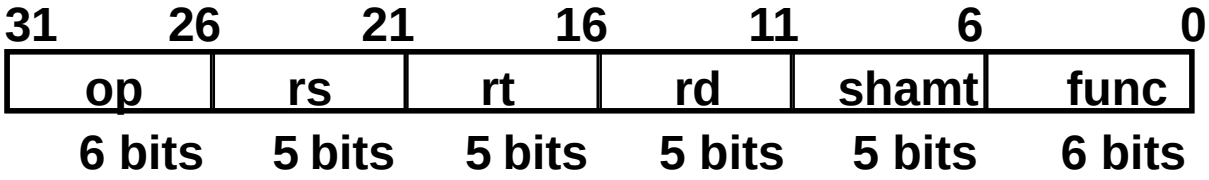
翟象平

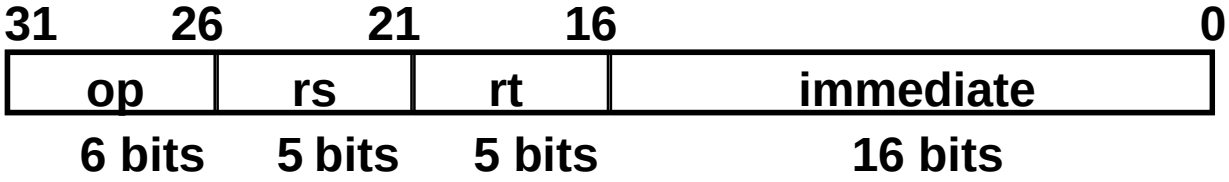
电邮: blueicezhaixp@nuaa.edu.cn

个人主页: http://cyber.nuaa.edu.cn

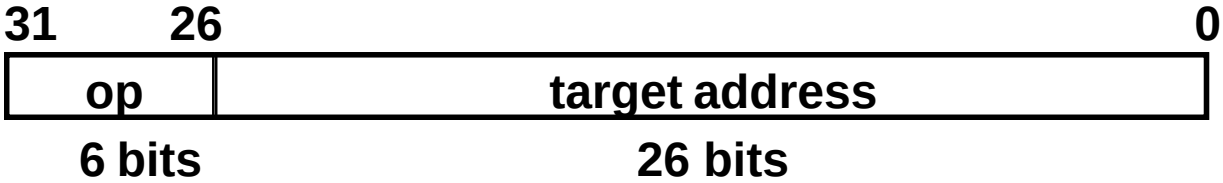
MIPS的三种指令类型

- ADD and SUBSTRACT
  - add rd, rs, rt
  - sub rd, rs, rt
- OR Immediate
  - ori rt, rs, imm16
- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16
- BRANCH
  - beq rs, rt, imm16
- JUMP
  - j target





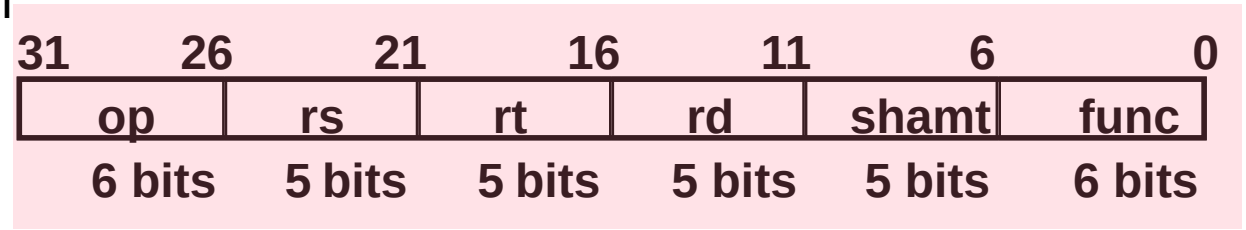
代表性指令：有算术运算、逻辑运算；有RR型、RI型；有访存指令；有条件转移、无条件转移。

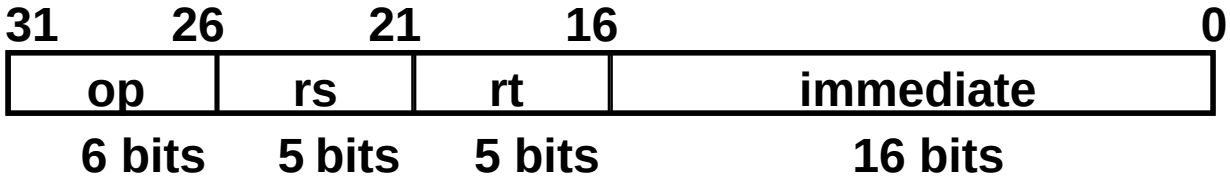


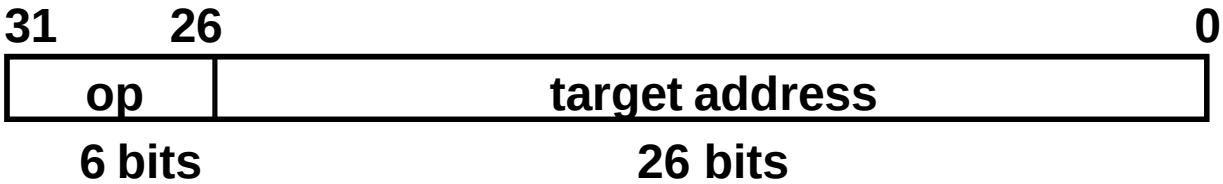
加法和减法指令(R-type类型)

- 实现目标（7条指令）：
- ADD and SUBSTRACT
  - add rd, rs, rt
  - sub rd, rs, rt
- OR Immediate
  - ori rt, rs, imm16
- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16
- BRANCH
  - beq rs, rt, imm16
- JUMP
  - j target

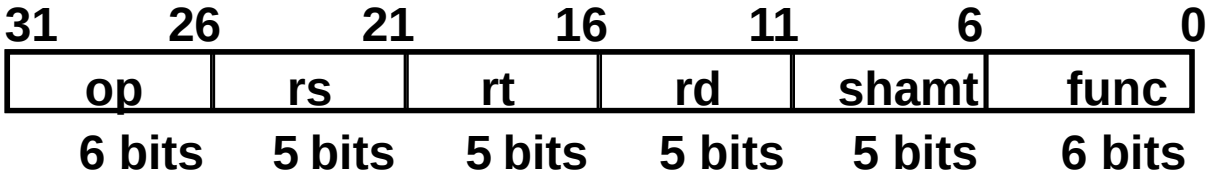
首先考虑add和sub指令（R-Type指令的代表）





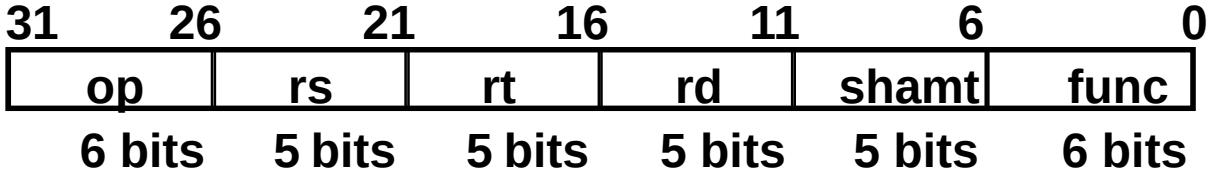


加法指令（ADD）

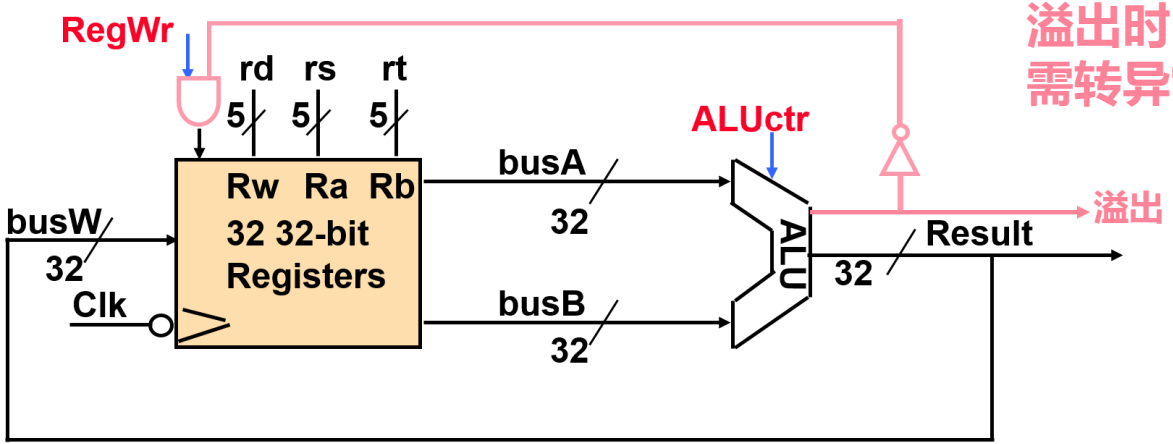


- add rd, rs, rt
  - M[PC] 从PC所指的内存单元中取指令
  - R[rd] ← R[rs] + R[rt] 从rs、rt 所指的寄存器中取数后相加。若结果不溢出，则将结果送rd 所指的寄存器中；若结果溢出，则不送结果，并转到“溢出处理程序”执行。
  - PC ← PC + 4 PC加4，使PC指向下一条指令

RR（R-type）型数据通路



- 功能：R[rd] ← R[rs] op R[rt]，如：add rd, rs, rt



溢出时，不写结果并需转异常处理程序

不考虑公共操作，仅R-Type指令执行阶段的数据通路

Ra, Rb, Rw 分别对应指令的rs, rt, rd

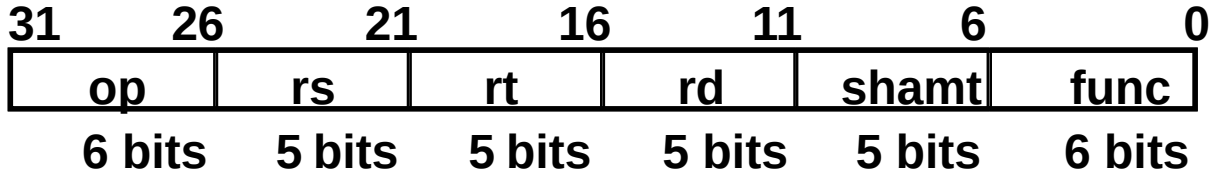
ALUctr, RegWr：指令译码后产生的控制信号

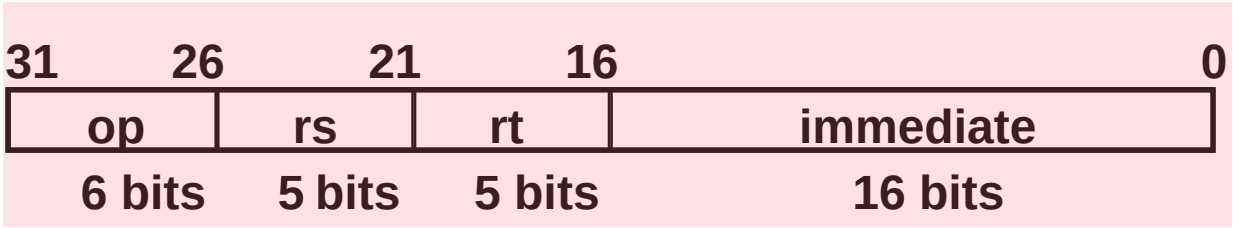
“add rd, rs, rt”控制信号？

ALUctr=add，RegWr=1

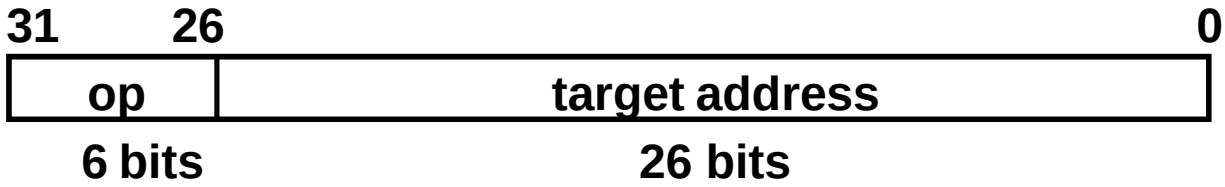
立即数逻辑指令（ori指令）

- 实现目标（7条指令）：
- ADD and SUBSTRACT
  - add rd, rs, rt
  - sub rd, rs, rt
- OR Immediate
  - ori rt, rs, imm16
- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16
- BRANCH
  - beq rs, rt, imm16
- JUMP
  - j target

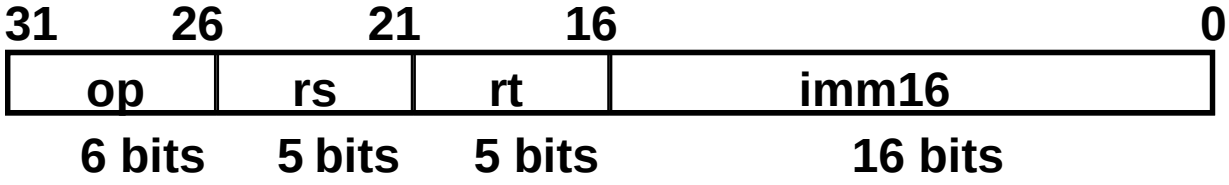




考虑ori 指令（I-Type指令和逻辑运算指令的代表）

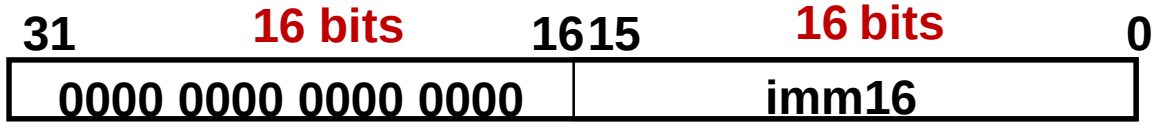


OR Immediate Instruction

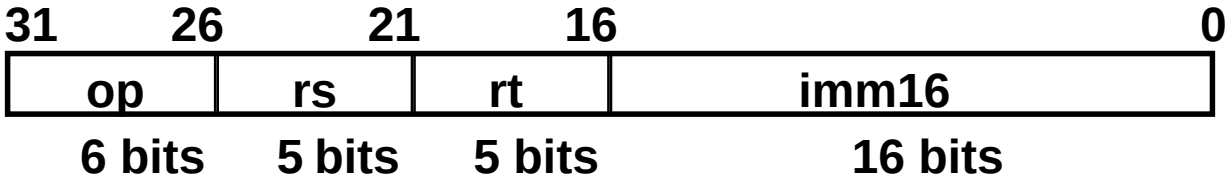


- ori rt, rs, imm16
  - M[PC]　　取指令（公共操作，取指部件完成）
  - R[rt] ← R[rs] or ZeroExt(imm16) 立即数零扩展，并与rs内容做“或”运算
  - PC ← PC + 4　　计算下地址（公共操作，取指部件完成）

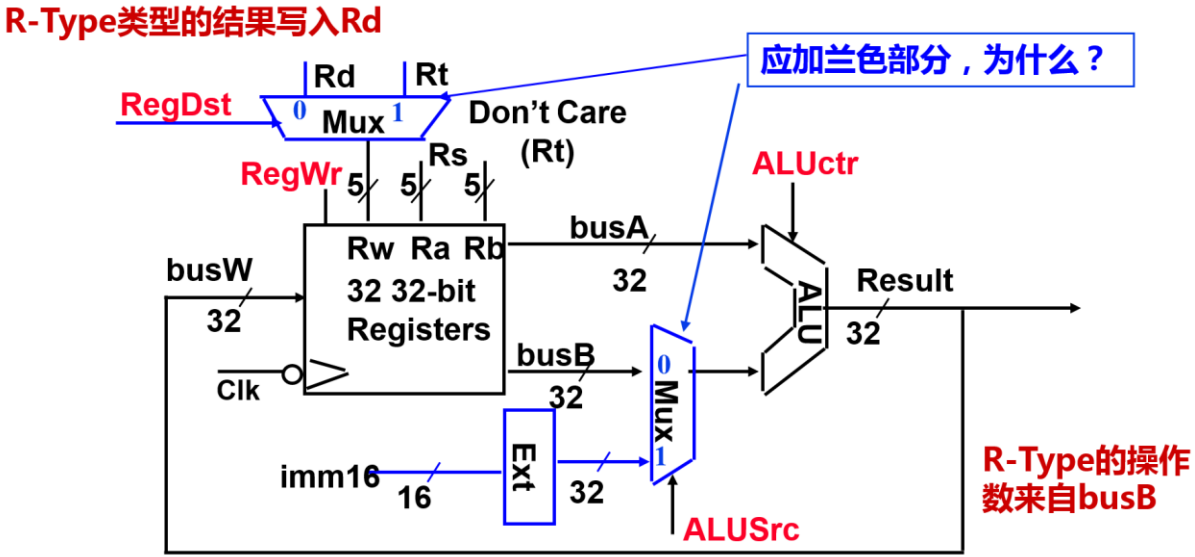
零扩展 ZeroExt(imm16)



立即数逻辑指令的数据通路



- R[rt] ← R[rs] op ZeroExt[imm16]]　　Example: ori　rt, rs, imm16



Ori指令的控制信号：

RegDst=1；

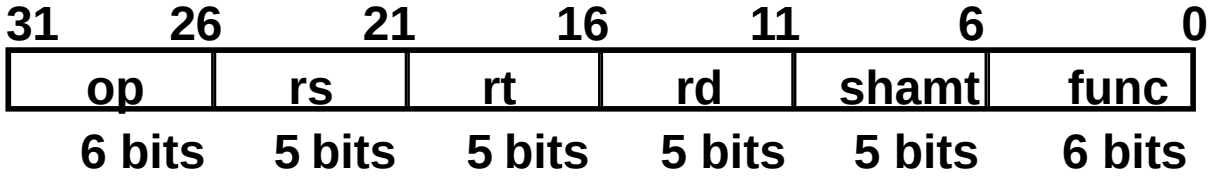
RegWr=1；

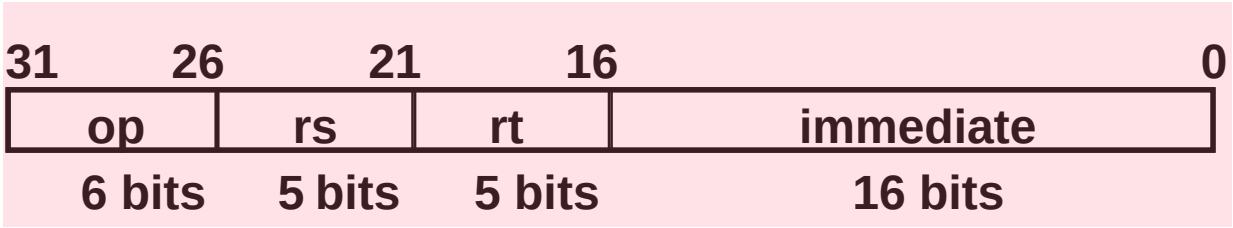
ALUSrc=1；

ALUctr=or

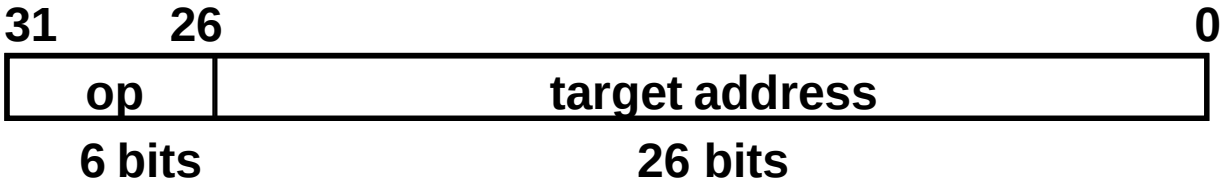
访存的数据装入指令 (lw)

- 实现目标（7条指令）：
- ADD and SUBSTRACT
  - add rd, rs, rt
  - sub rd, rs, rt
- OR Immediate
  - ori rt, rs, imm16
- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16
- BRANCH
  - beq rs, rt, imm16
- JUMP
  - j target

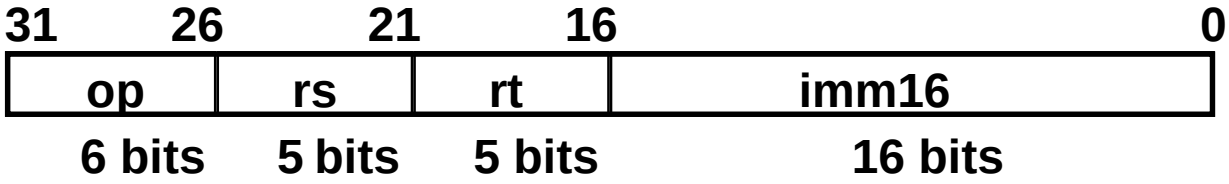




考虑lw 指令（访存指令的代表）

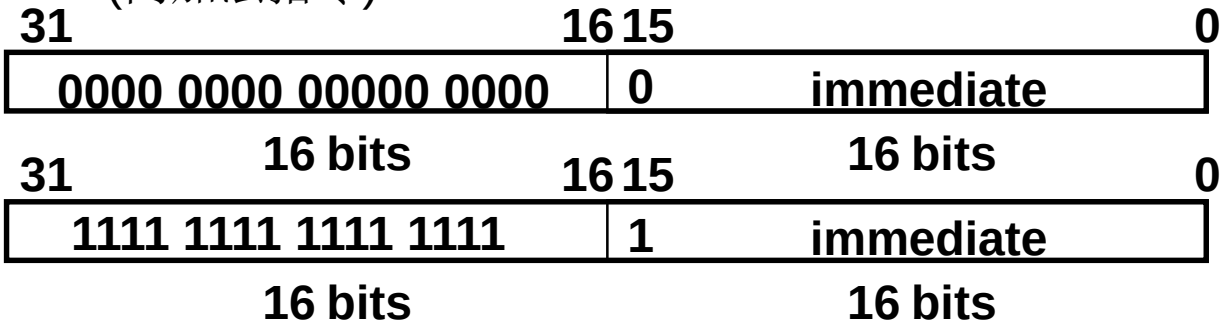


装入指令（Load）

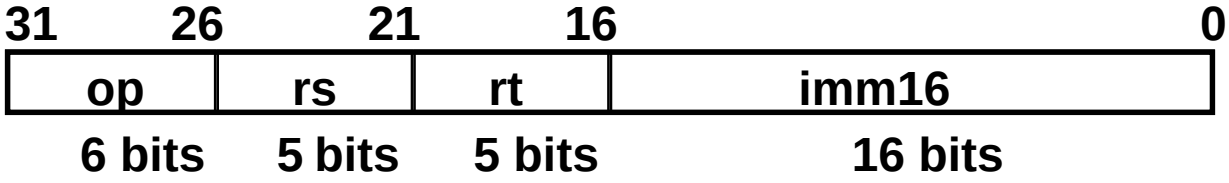


- lw rt, rs, imm16
  - M[PC] (同加法指令)
  - Addr ← R[rs] + SignExt(imm16) 计算数据地址 (立即数要进行符号扩展)
  - R[rt] ← M[Addr] 从存储器中取出数据，装入到寄存器中
  - PC ← PC + 4 (同加法指令)

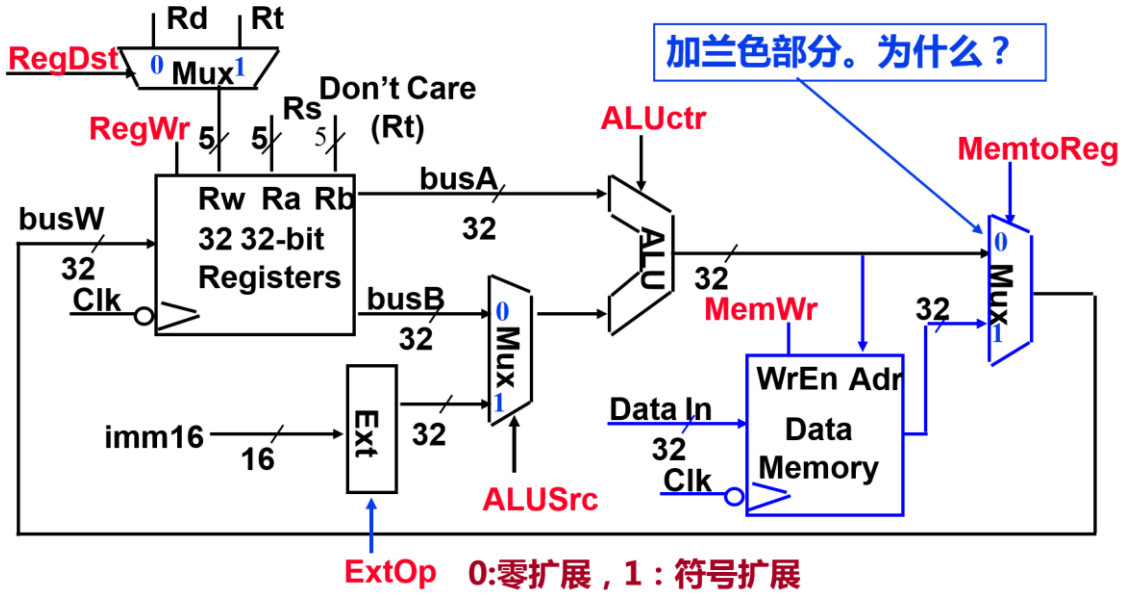
- 符号扩展
  - 为什么不是零扩展?



装入(lw)指令的数据通路



- R[rt] ← M[R[rs] + SignExt[imm16]]　　lw　rt, rs, imm16



RegDst=1,

RegWr=1,

ALUctr=addu,

ExtOp=1,

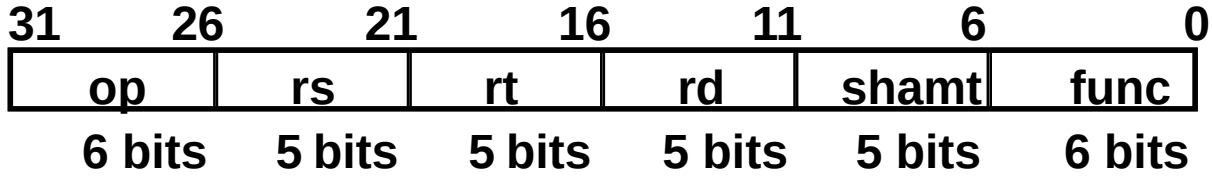
ALUSrc=1,

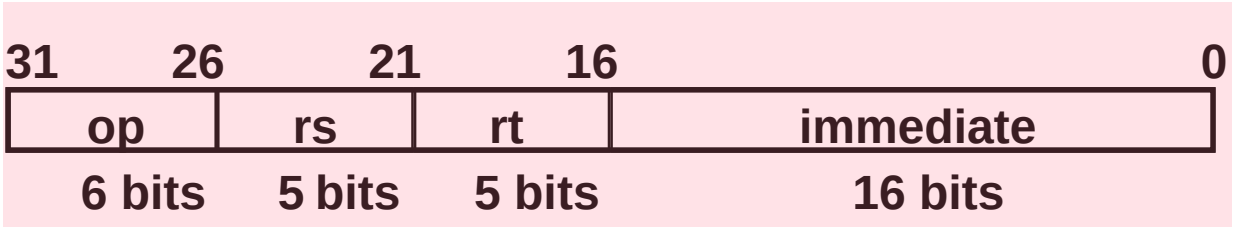
MemWr=0,

MemtoReg=1

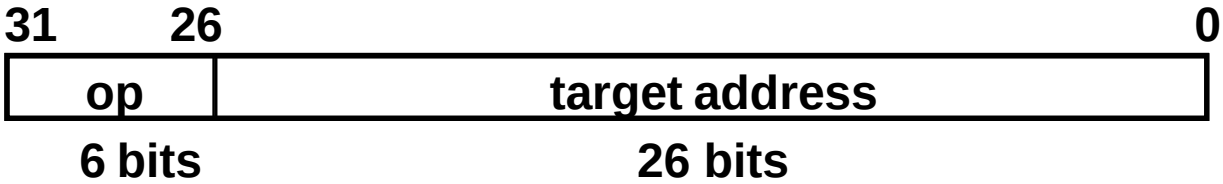
访存的存数指令 (sw)

- 实现目标（7条指令）：
- ADD and SUBSTRACT
  - add rd, rs, rt
  - sub rd, rs, rt
- OR Immediate
  - ori rt, rs, imm16
- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16
- BRANCH
  - beq rs, rt, imm16
- JUMP
  - j target

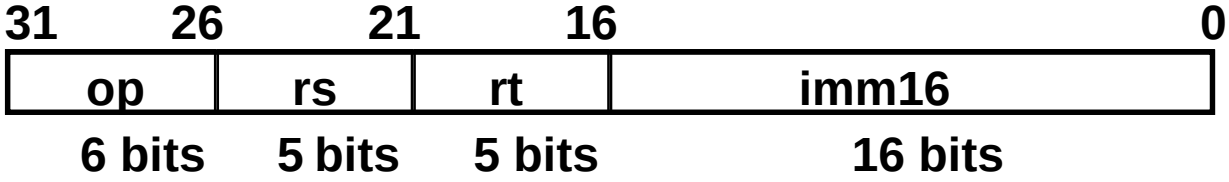




考虑sw 指令（访存指令的代表）

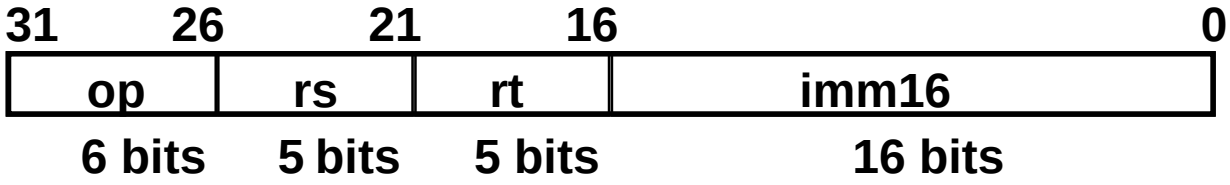


Store Instruction

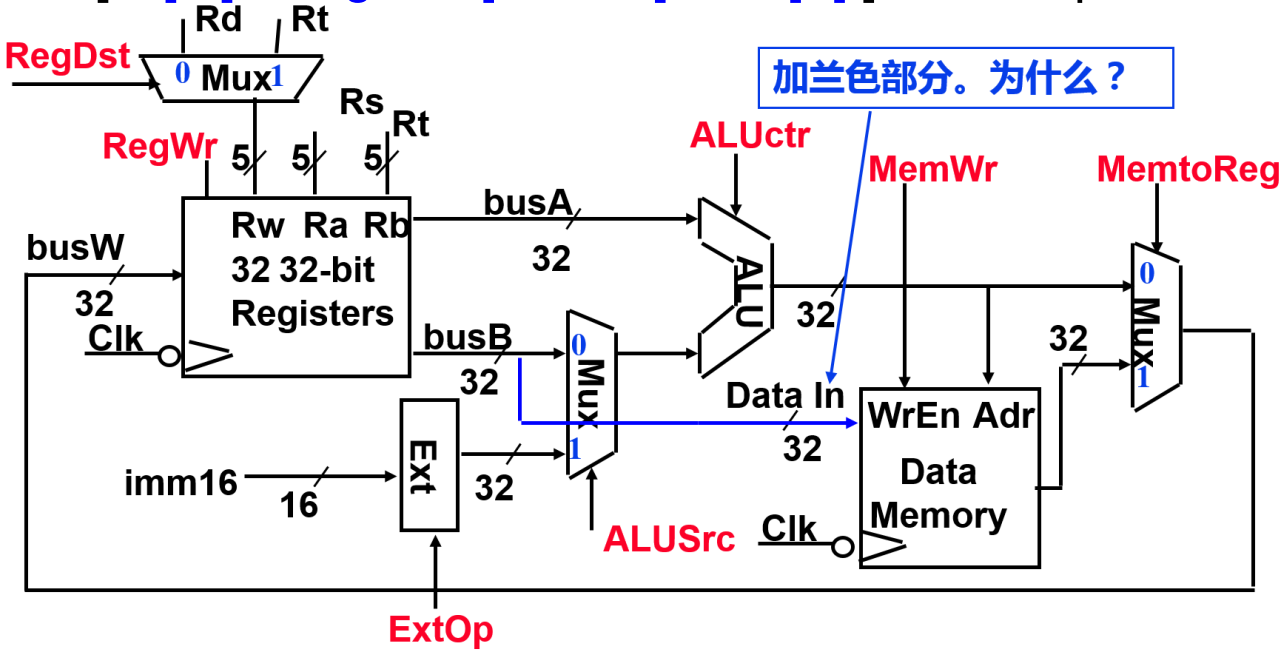


- sw rt, rs, imm16
  - M[PC]　　取指令（公共操作，取指部件完成）
  - Addr ← R[rs] + SignExt(imm16)　计算存储单元地址（符号扩展！）
  - Mem[Addr] ← R[rt]　寄存器rt中的内容存到内存单元中
  - PC ← PC + 4　计算下地址（公共操作，取指部件完成）

存数(sw)指令的数据通路



- M[R[rs] + SignExt[imm16] ← R[rt]] Example: sw rt, rs, imm16



RegDst=x,

RegWr=0,

ALUctr=addu,

ExtOp=1,

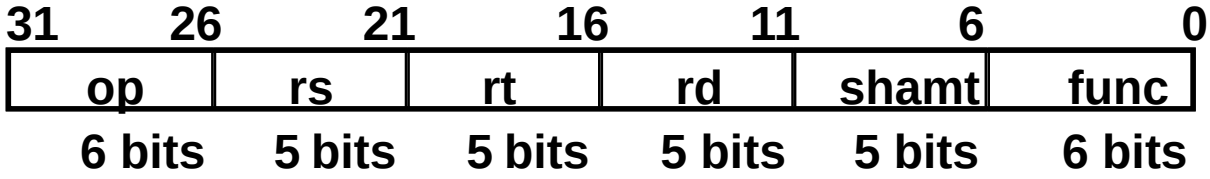
ALUSrc=1,

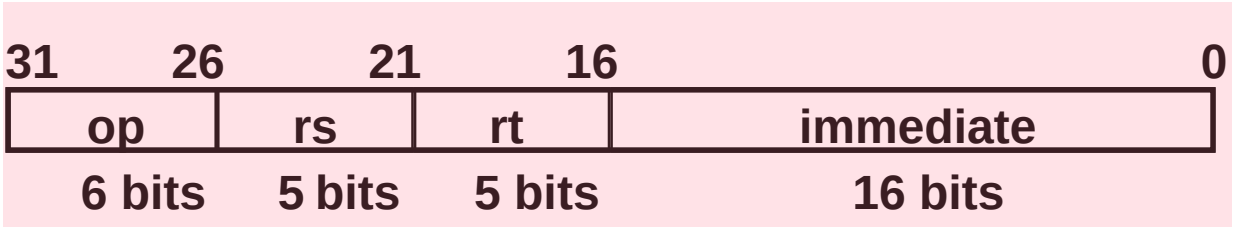
MemWr=1,

MemtoReg=x

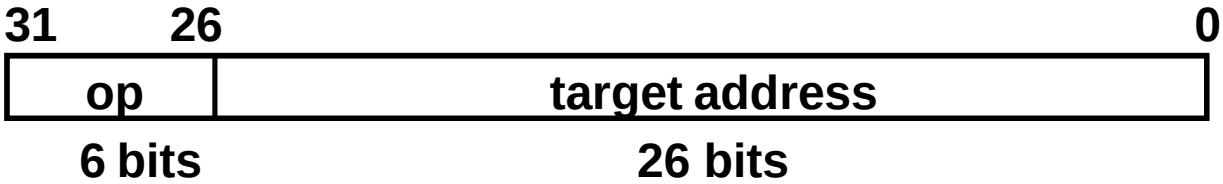
分支（条件转移）指令

- 实现目标（7条指令）：
- ADD and SUBSTRACT
  - add rd, rs, rt
  - sub rd, rs, rt
- OR Immediate
  - ori rt, rs, imm16
- LOAD and STORE
  - lw rt, rs, imm16
  - sw rt, rs, imm16
- BRANCH
  - beq rs, rt, imm16
- JUMP
  - j target

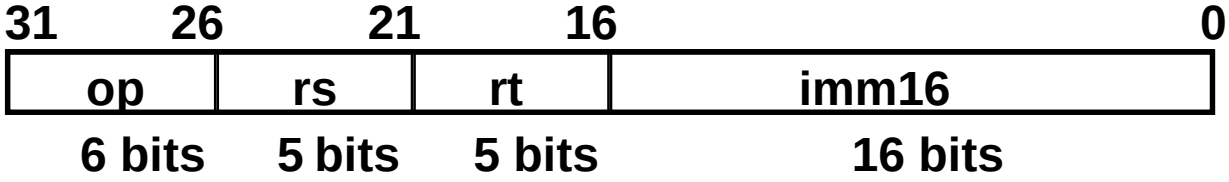




考虑beq指令（条件转移指令的代表）



Branch Instruction



- beq rs, rt, imm16
  - M[PC] 取指令（公共操作，取指部件完成）
  - Cond ← R[rs] - R[rt] 做减法比较rs和rt中的内容
  - if (COND eq 0) 计算下地址（根据比较结果，修改PC）

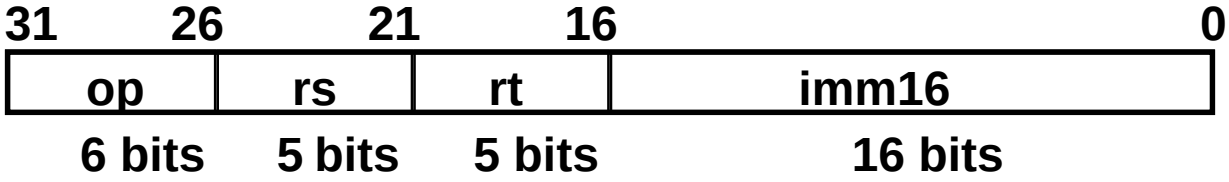
    PC ← PC + 4 + (SignExt(imm16) x 4)

    else

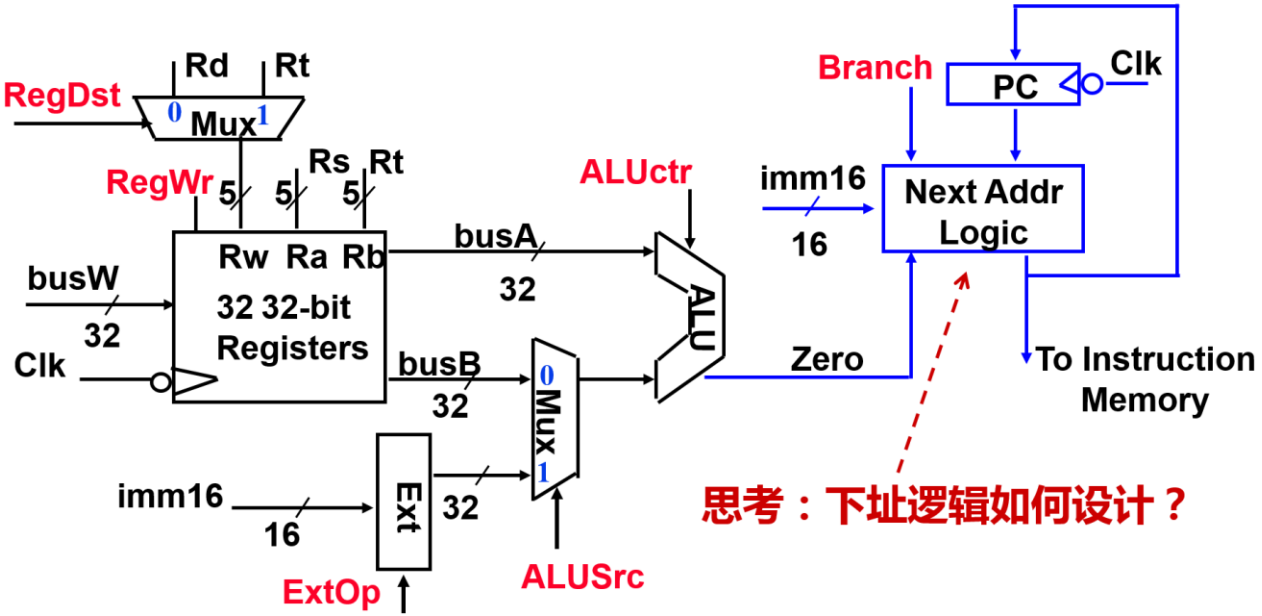
    PC ← PC + 4

立即数的含义是什么？是相对指令数还是相对单元数？

条件转移指令的数据通路



- beq rs, rt, imm16 比较Rs 与 Rt !



RegDst=x,

RegWr=0,

ALUctr=subu,

ExtOp=x,

ALUSrc=0,

MemWr=0,

MemtoReg=x,

Branch=1

Q & A

THANKS