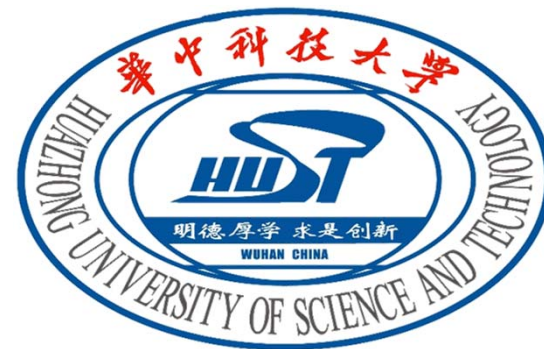


微机原理与接口技术

MIPS数据传输指令

华中科技大学 左冬红



数据传输指令

- 寄存器与存储器之间数据传输
- 通用寄存器与特殊寄存器之间数据传输
- 通用寄存器赋值指令

表述约定

寄存器编号或寄存器名称

\$17,\$s1

寄存器的值

RF[\$s1]

存储单元地址

Imm(\$17),Imm(\$s1)

存储的数据

mem[Imm+RF[\$s1]]

立即数

4, -12

Imm

存储器到寄存器数据传送

装载: load 简写l

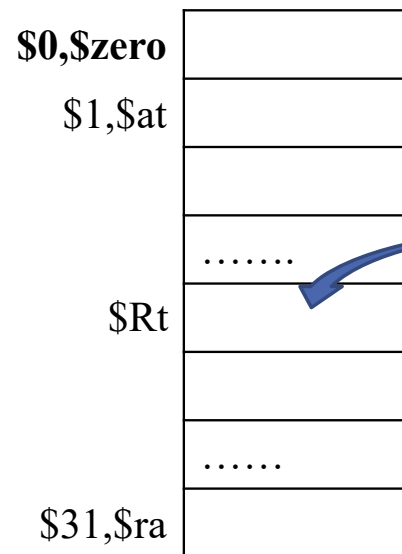
指令格式

lx \$Rt, imm(\$Rs)

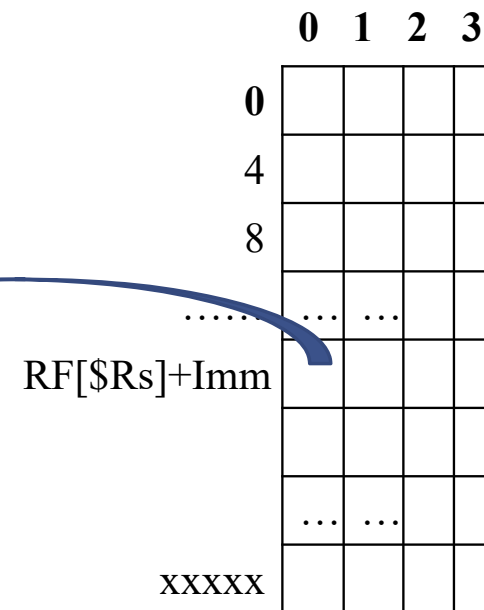
x——w, h, hu, b, bu

指令功能

1个寄存器32位



CPU



存储器

1个单元1B

存储器到寄存器数据传送

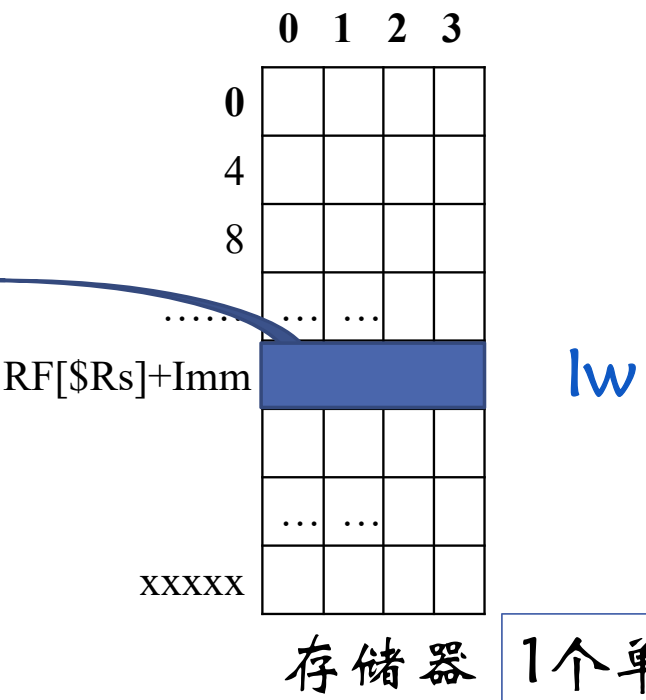
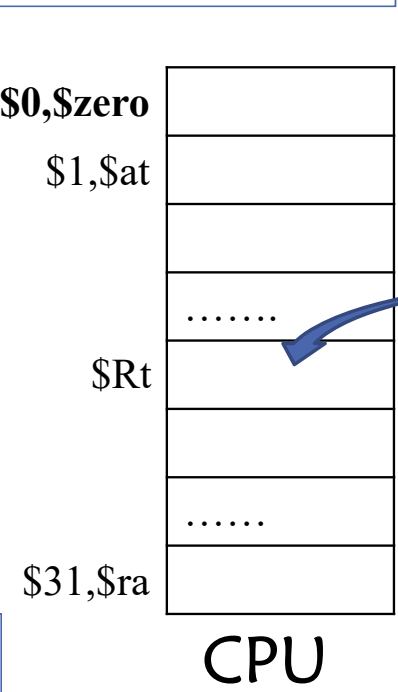
装载: load 简写l

指令格式

lw \$Rt, imm(\$Rs)

指令功能

1个寄存器32位



存储器到寄存器数据传送

装载: load 简写l

指令格式

$l\ x\ \$Rt,\ imm(\$Rs)$

指令功能

1个寄存器32位

CPU

扩展

符号扩展

lh

扩展符号位

无符号扩展

lhu

扩展0

0 1 2 3

$\$0,\$zero$

$\$1,\at

$\$Rt$

$\$31,\ra

$RF[\$Rs]+Imm$

XXXXX

	0	1	2	3
0				
4				
8				
...		
...				
...		
XXXXX				

lh,lhu

存储器

1个单元1B

存储器到寄存器数据传送

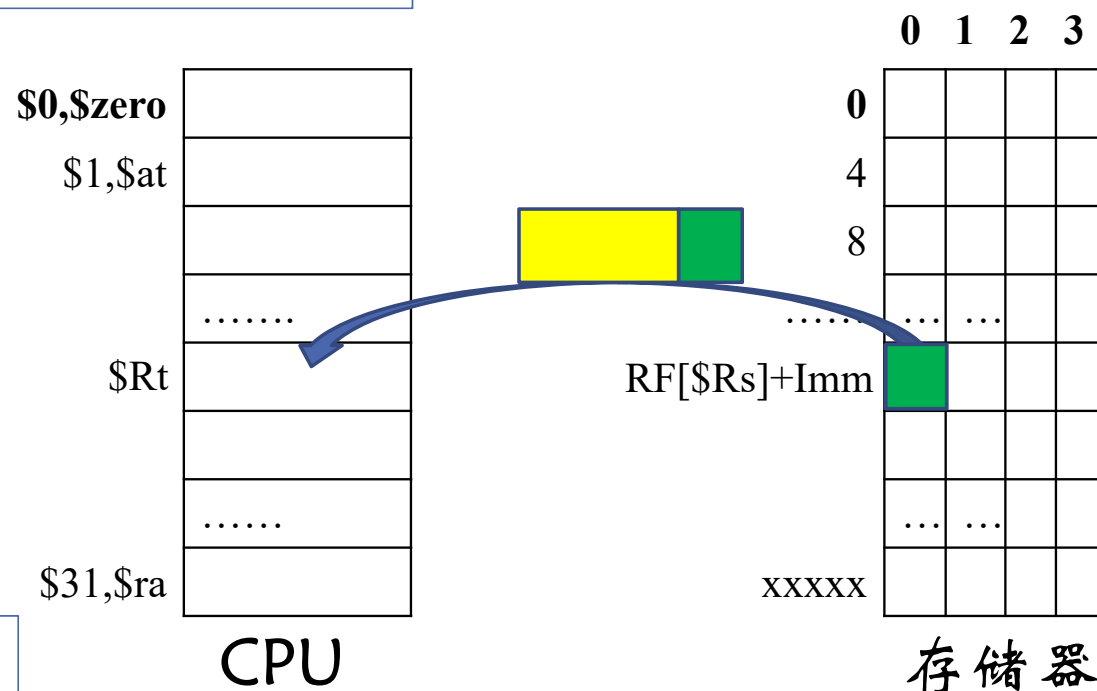
装载: load 简写l

指令格式

lx \$Rt, imm(\$Rs)

指令功能

1个寄存器32位



1个单元1B

存储器到寄存器数据传送

装载: load 简写l

指令格式

$l_x \$Rt, imm(\$Rs)$

指令功能

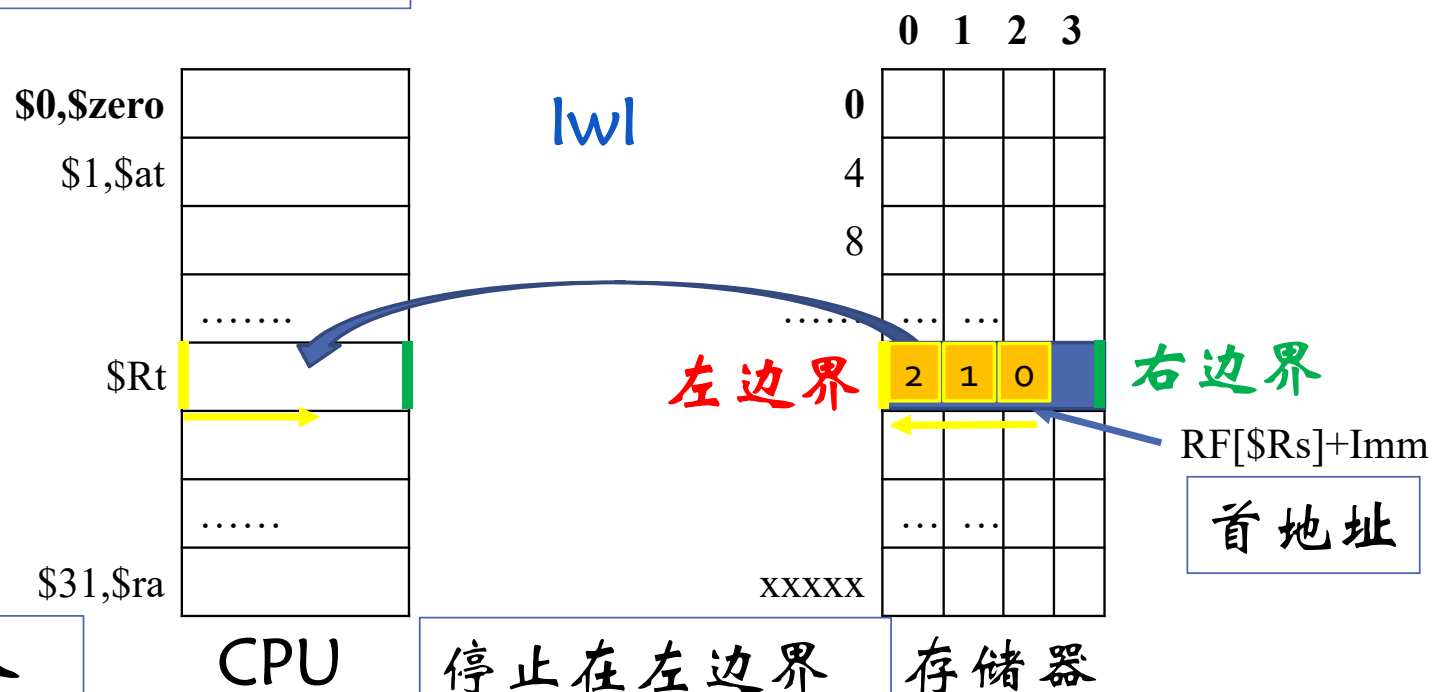
从左边开始写入

CPU

停止在左边界

存储器

lwl , 左边界对齐非规则字访问
 lwr , 右边界对齐非规则字访问



存储器到寄存器数据传送

装载: load 简写l

指令格式

$l_x \ \$Rt, imm(\$Rs)$

lwr , 右边界对齐非规则字访问

指令功能

从右边开始写入

$\$0, \$zero$

$\$1, \at

$\$Rt$

$\$31, \ra

CPU

0 1 2 3

0				
4				
8				
...		
			0	1
...	...			

右边界

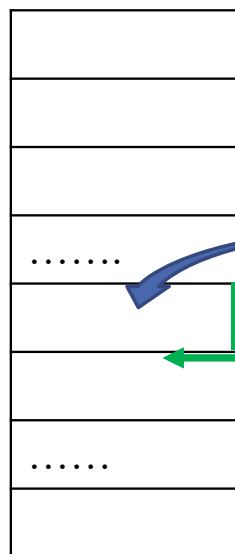
$RF[\$Rs] + Imm$

首地址

XXXXXX

停止在右边界

存储器



...

...

...

存储器到寄存器数据传输示例

lw \$t0,0(\$s0)

存储器地址

$$\begin{aligned} & \text{RF}[\$Rs] + \text{Imm} \\ &= 0x10040000 + 0 \\ &= 0x10040000 \end{aligned}$$

\$0,\$zero	

\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000

\$s0	0x10040000

\$31,\$ra	

CPU

MIPS大字节序

0x8078ab34

	0	1	2	3
0				
4				
8				
.....		
0x10040000	0x80	0x78	0xab	0x34
		
xxxxxx				

存储器

存储器到寄存器数据传输示例

lh \$t0,0(\$s0)

lhu \$t0,0(\$s0)

\$0,\$zero	
.....	
\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000
.....	
\$s0	0x10040000
.....	
\$31,\$ra	

CPU

0xffff8078

	0	1	2	3
0				
4				
8				
.....		
0x10040000	0x80	0x78	0xab	0x34
		
XXXXXX				

0x00008078

存储器

存储器到寄存器数据传输示例

```
lb $t0,0($s0)
```

```
lbu $t0,0($s0)
```

\$0,\$zero	
.....	
\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000
.....	
\$s0	0x10040000
.....	
\$31,\$ra	

CPU

0xffffffff80

	0	1	2	3
0				
4				
8				
.....		
0x10040000	0x80	0x78	0xab	0x34
		
xxxxxx				

0x00000080

存储器

存储器到寄存器数据传输示例

lwl \$t0,2(\$s0)

存储器首地址

$$\begin{aligned} & \text{RF}[\$Rs] + \text{Imm} \\ &= 0x10040000 + 2 \\ &= 0x10040002 \end{aligned}$$

lwr \$t0,-1(\$s0)

存储器首地址

$$\begin{aligned} & \text{RF}[\$Rs] + \text{Imm} \quad \text{-1的补码} \\ &= 0x10040000 + 0xffffffff \\ &= 0x1003ffff \end{aligned}$$

Imm在机器指令中为16位符号数补码，扩展为32位之后再参与运算

\$0,\$zero	

\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000

\$s0	0x10040000

\$31,\$ra	

CPU

	0	1	2	3
0				
4				
8				↓→
0x1003fffc		0x23
0x10040000	0x80	0x78	0xab	0x34
	←		↑	
		
xxxxx				

存储器

存储器到寄存器数据传输示例

lwl \$t0,2(\$s0)

lwr \$t0,-1(\$s0)

首地址是否边界对齐?

什么字节序?

功能：非规则字数据读取

\$0,\$zero	
.....	
\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000
.....	
\$s0	0x10040000
.....	
\$31,\$ra	

CPU

	0	1	2	3
0				
4				
8				
0x1003fffc		0x23
0x10040000	0x80	0x78	0xab	0x34
		
xxxxxx				

存储器

寄存器到存储器数据传输

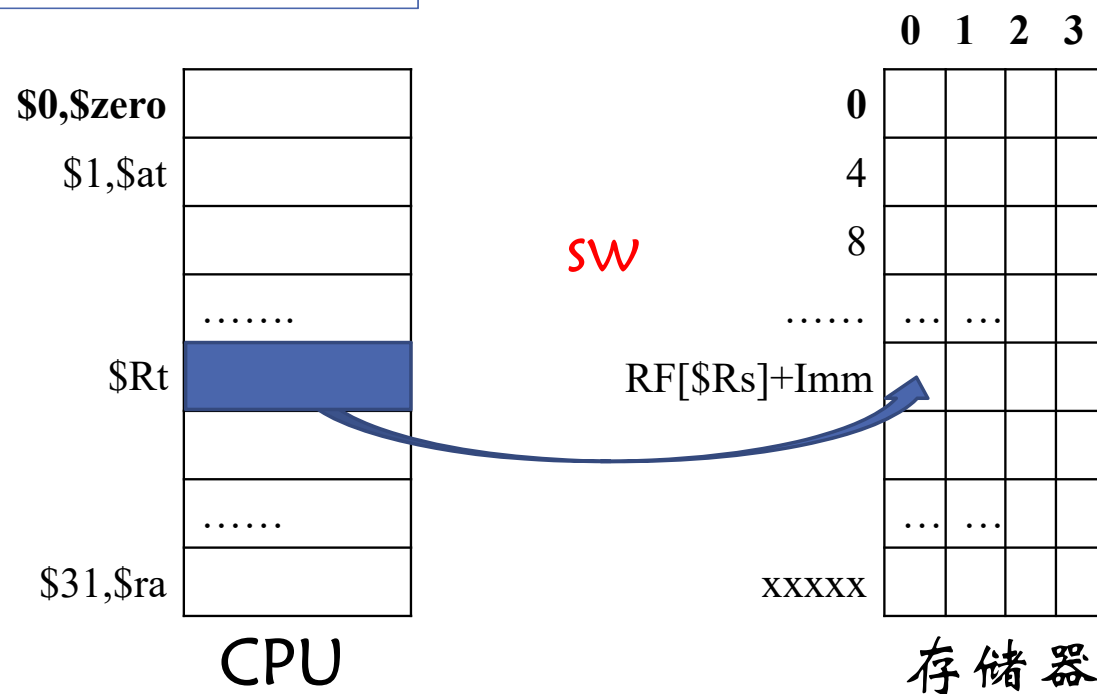
存储: store 简写s

指令格式

sx $\$Rt$, $imm(\$Rs)$

x — w, h, b

指令功能



寄存器到存储器数据传输

存储: store 简写s

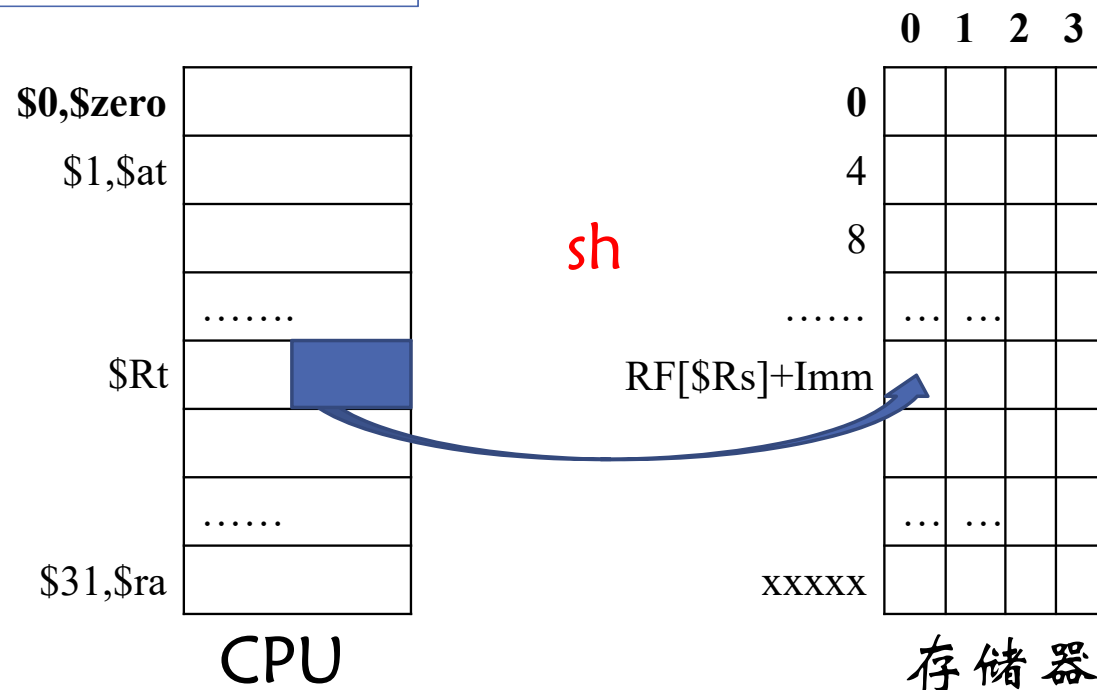
指令格式

$sx \ \$Rt, \text{imm}(\$Rs)$

x — w, h, b

指令功能

截断, 存储低半字



寄存器到存储器数据传输

存储: store 简写s

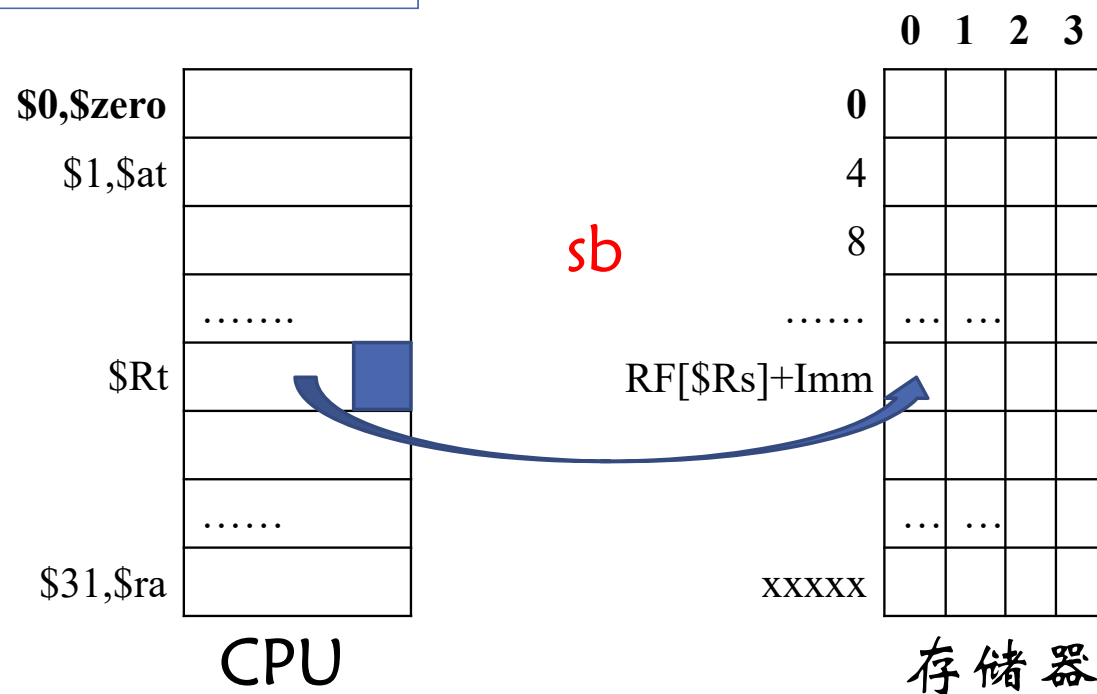
指令格式

$sx \ \$Rt, imm(\$Rs)$

x — w, h, b

指令功能

截断, 存储低字节



寄存器到存储器数据传输

存储: store 简写s

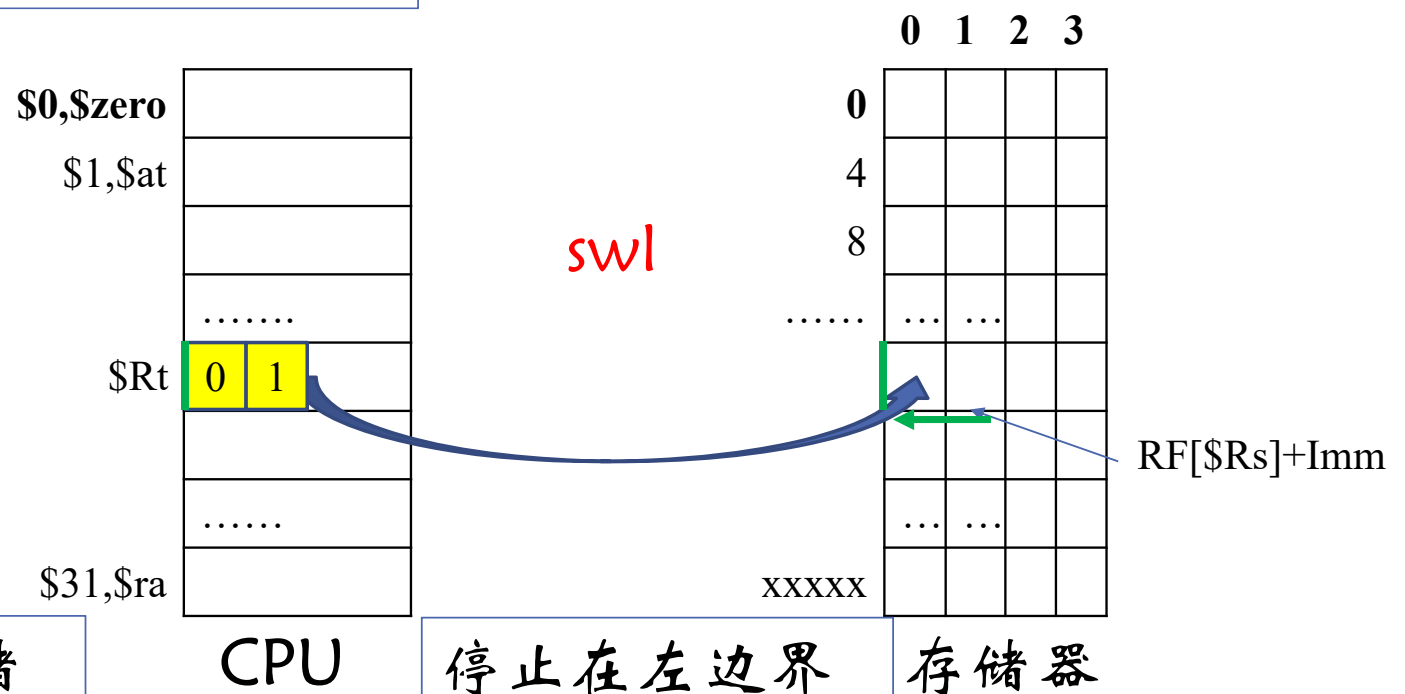
指令格式

$sx \ \$Rt, \text{imm}(\$Rs)$

x — w, h, b

指令功能

从左边开始存储



停止在左边界

存储器

寄存器到存储器数据传输

存储: store 简写s

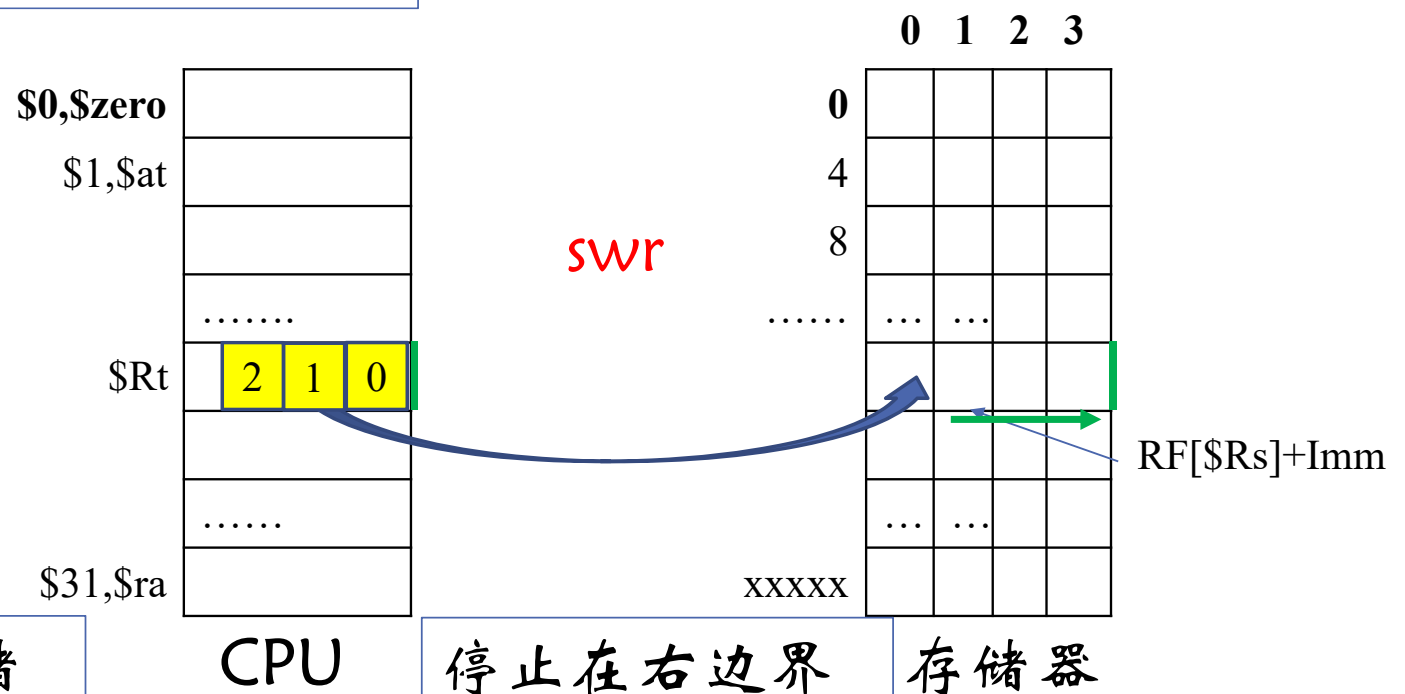
指令格式

$sx \ \$Rt, \text{imm}(\$Rs)$

$x \text{---} w, h, b$

指令功能

从右边开始存储



寄存器到存储器数据传输示例

sw \$t0,0(\$s0)

存储器地址

$$\begin{aligned} & \text{RF}[\$Rs] + \text{Imm} \\ &= 0x10040000 + 0 \\ &= 0x10040000 \end{aligned}$$

\$0,\$zero	

\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000

\$s0	0x10040000

\$31,\$ra	

CPU

	0	1	2	3
0				
4				
8				
.....		
0x10040000	0x80	0x78	0xab	0x34
		
xxxxxx				

存储器

寄存器到存储器数据传输示例

sh \$t0,0(\$s0)

存储器地址

$$\begin{aligned} & \text{RF}[\$Rs] + \text{Imm} \\ &= 0x10040000 + 0 \\ &= 0x10040000 \end{aligned}$$

\$0,\$zero	

\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000

\$s0	0x10040000

\$31,\$ra	

CPU

	0	1	2	3
0				
4				
8				
.....		
0x10040000	0x80	0x78	0xab	0x34
		
xxxxxx				

存储器

寄存器到存储器数据传输示例

sb \$t0,0(\$s0)

存储器地址

$$\begin{aligned} & \text{RF}[\$Rs] + \text{Imm} \\ &= 0x10040000 + 0 \\ &= 0x10040000 \end{aligned}$$

\$0,\$zero	

\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000

\$s0	0x10040000

\$31,\$ra	

CPU

	0	1	2	3
0				
4				
8				
.....		
0x10040000	0x80	0x78	0xab	0x34
		
xxxxxx				

存储器

寄存器到存储器数据传输示例

swl \$t0,1(\$s0)

存储器地址

$$\begin{aligned} & \text{RF}[\$R_s] + \text{Imm} \\ &= 0x10040000 + 1 \\ &= 0x10040001 \end{aligned}$$

swr \$t0,-2(\$s0)

存储器地址

-2的补码

$$\begin{aligned} & \text{RF}[\$R_s] + \text{Imm} \\ &= 0x10040000 + 0xffffffffe \\ &= 0x1003fffe \end{aligned}$$

\$0,\$zero	
.....	
\$t0	0x40806070
\$t1	0x10010000
\$t2	0x10018000
.....	
\$s0	0x10040000
.....	
\$31,\$ra	

CPU

	0	1	2	3
0				
4				
8				
0x1003fffc	0x10	0x20
0x10040000	0x80	0x78	0xab	0x34
.....		
xxxxxx				

存储器

特殊寄存器与通用寄存器之间数据互传

特殊寄存器:hi、lo

保存乘除法运算结果:

乘法: hi保存高32位、lo保存低32位

除法: hi保存余数、lo保存商

指令格式

mfhi \$Rd

mflo \$Rd

mthi \$Rs

mtlo \$Rs

指令功能

$RF[\$Rd] = RF[hi]$

$RF[\$Rd] = RF[lo]$

$RF[hi] = RF[\$Rs]$

$RF[lo] = RF[\$Rs]$

寄存器赋初值

指令格式

lui \$Rt, Imm

指令功能

$RF(\$Rt) = \{\text{Imm}, 16'h0\}$

通常与逻辑运算指令ori一起实现给寄存器赋32位初值

小结

- 数据传送指令
 - 存储器与寄存器之间数据传输有多种类型
 - 存储器数据传输到寄存器若需位扩展
 - 符号扩展
 - 无符号扩展
 - 寄存器传输到存储器只需截断
 - 非规则字存\取指令必须成对使用
 - hi\lo寄存器与通用寄存器之间的数据传输
 - 指令中仅一个操作数\$Rd或\$Rs, hi,lo隐含在操作码中
 - 寄存器赋初值，数据传送lui指令仅实现高16位赋值

下一讲：运算类指令