

本节主题



控制信号的集成

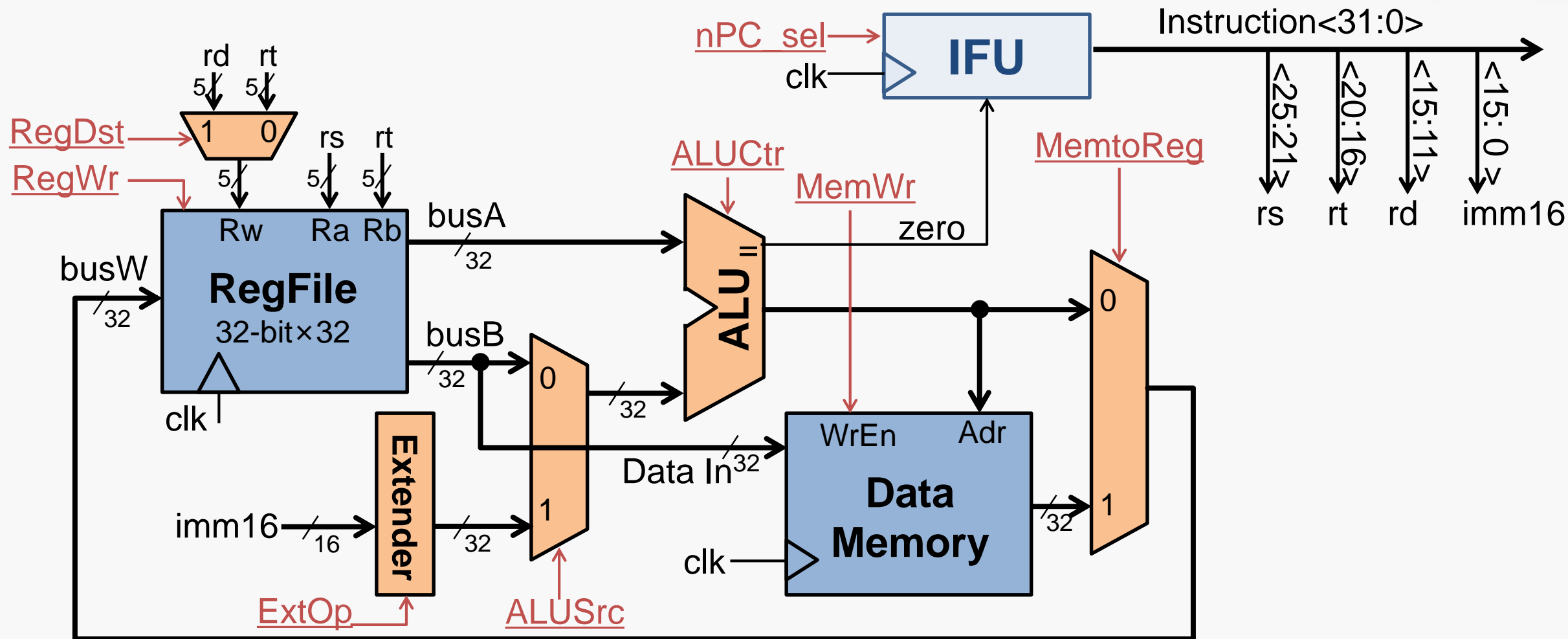
北京大学·慕课

计算机组成

制作人：陆俊林



现有指令所需的控制信号

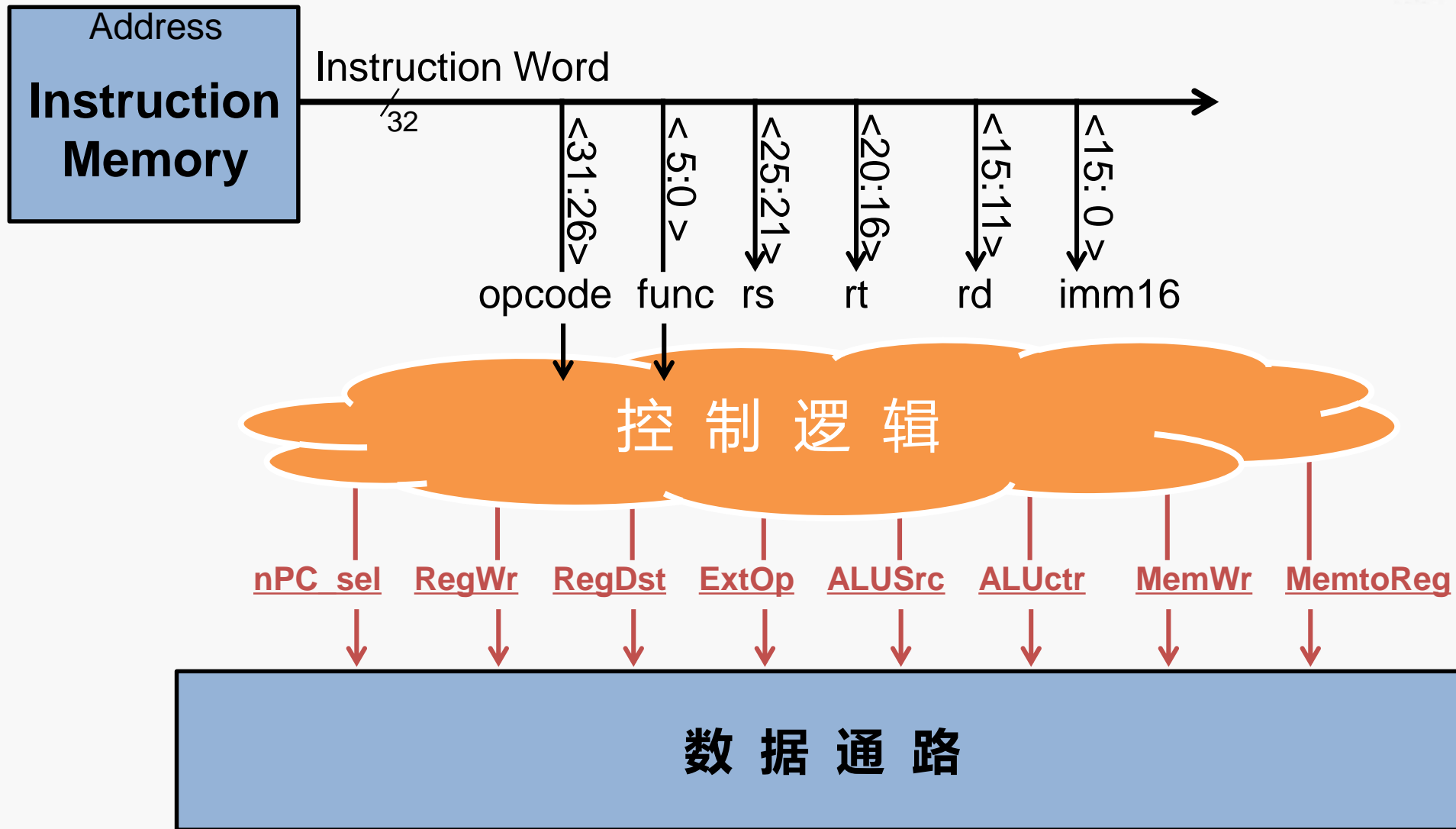


处理器的设计步骤



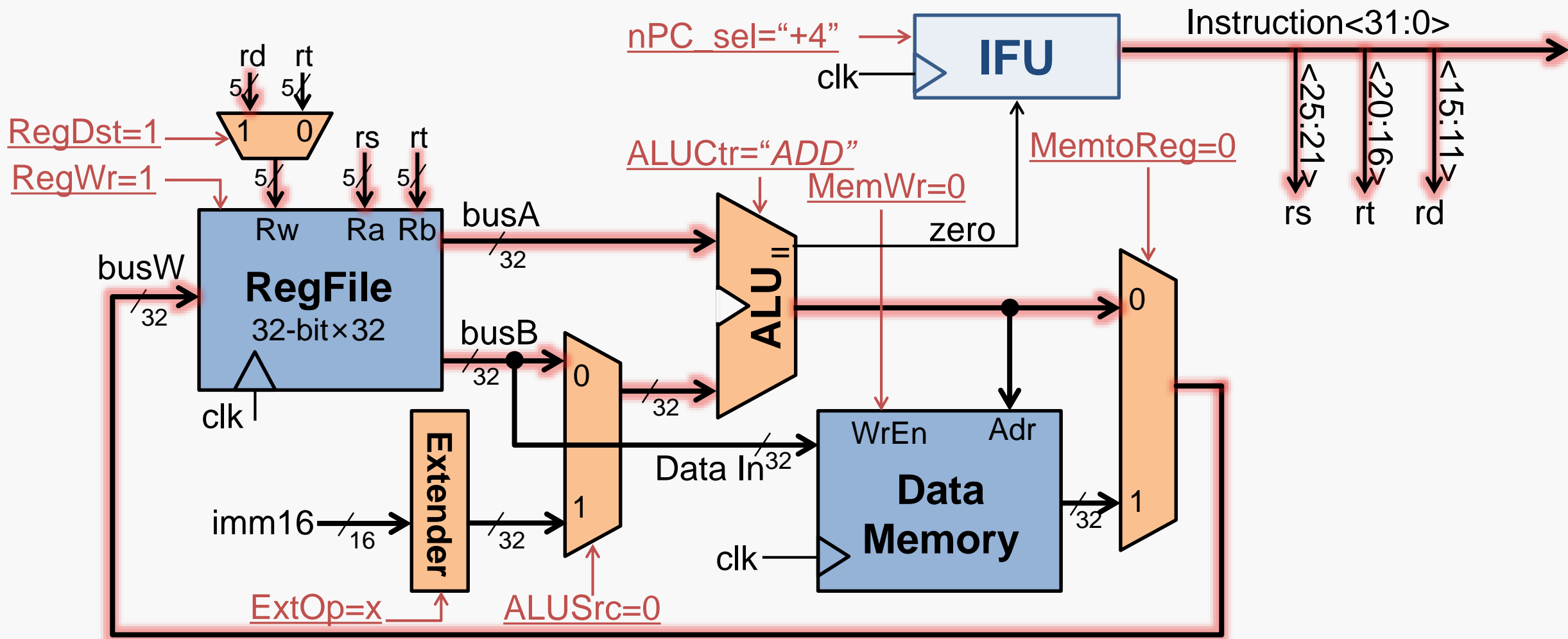
- ① 分析指令系统，得出对数据通路的需求 ✓
- ② 为数据通路选择合适的组件 ✓
- ③ 连接组件建立数据通路 ✓
- ④ 分析每条指令的实现，以确定控制信号 ✓
- ⑤ 集成控制信号，形成完整的控制逻辑

控制逻辑与数据通路



控制信号的汇总（以add指令为例）

add: $R[rd] \leftarrow R[rs] + R[rt]; PC \leftarrow PC + 4$



控制信号的逻辑表达式

func opcode (op)	100000	100010	/			
	000000	000000	001101	100011	101011	000100
	add	sub	ori	lw	sw	beq
RegDst	1	1	0	0	x	x
ALUSrc	0	0	1	1	1	0
MemtoReg	0	0	0	1	x	x
RegWr	1	1	1	1	0	0
MemWr	0	0	0	0	1	0
nPC_sel	0	0	0	0	0	1
ExtOp	x	x	0	1	1	x
ALUctr<1:0>	00 (ADD)	01 (SUB)	10 (OR)	00 (ADD)	00 (ADD)	01 (SUB)

控制信号的逻辑表达式

func	100000	100010	/			
opcode (op)	000000	000000	001101	100011	101011	000100
	add	sub	ori	lw	sw	beq
RegDst	1	1	0	0	x	x

RegDst = add + sub

add = **rtype** · func5 · ~func4 · ~func3 · ~func2 · ~func1 · ~func0

sub = **rtype** · func5 · ~func4 · ~func3 · ~func2 · func1 · ~func0

rtype = ~op5 · ~op4 · ~op3 · ~op2 · ~op1 · ~op0

R	opcode	rs	rt	rd	shamt	funct	add, sub
I	opcode	rs	rt	immediate			ori, lw, sw, beq

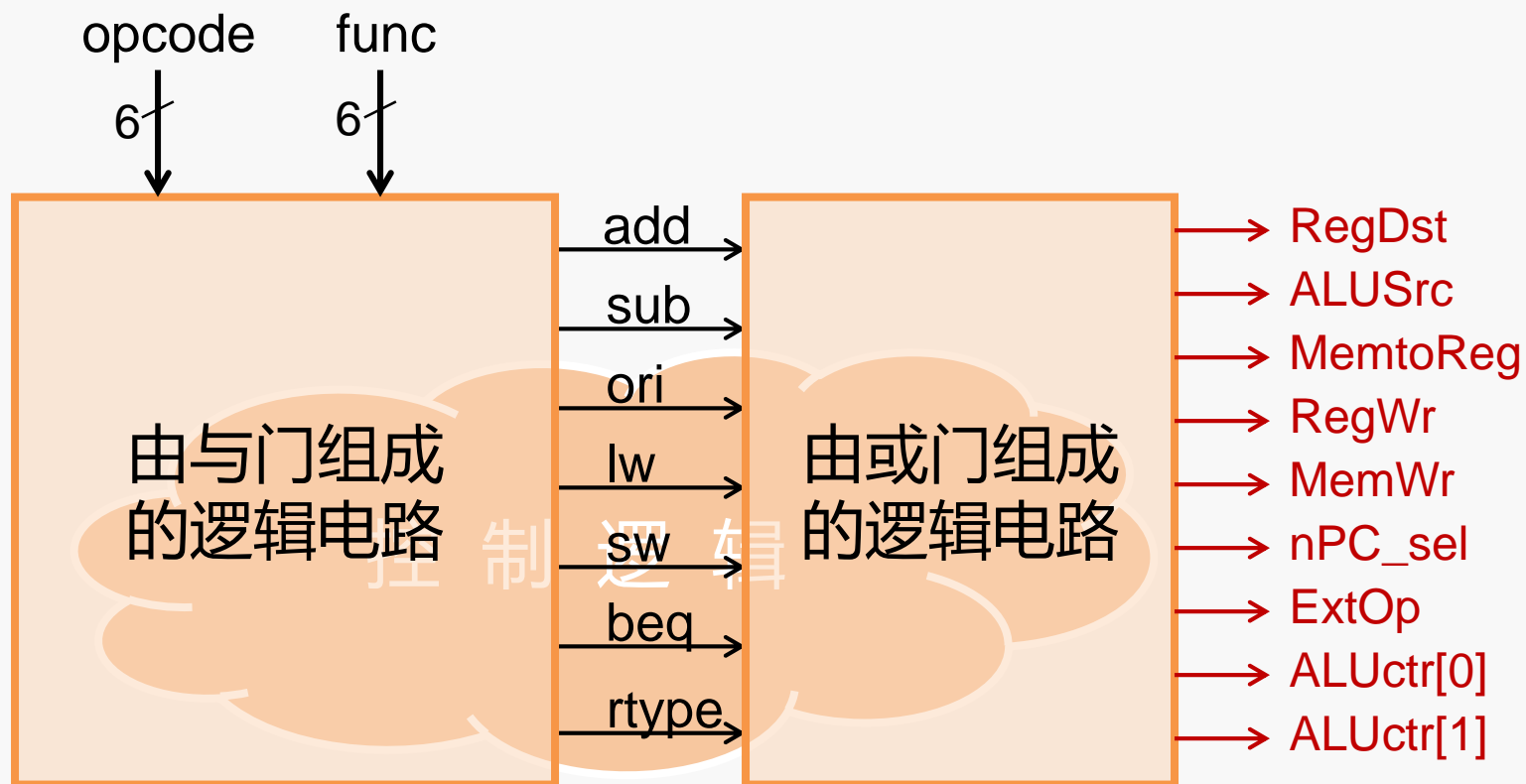
控制器的逻辑表达式



```
RegDst      = add + sub
ALUSrc      = ori + lw + sw
MemtoReg    = lw
RegWr       = add + sub + ori + lw
MemWr       = sw
nPC_sel     = beq
ExtOp       = lw + sw
ALUctr[0]   = sub + beq
ALUctr[1]   = or
```

```
add    = rtype · func5 · ~func4 · ~func3 · ~func2 · ~func1 · ~func0
sub    = rtype · func5 · ~func4 · ~func3 · ~func2 · func1 · ~func0
rtype  = ~op5 · ~op4 · ~op3 · ~op2 · ~op1 · ~op0,
ori    = ~op5 · ~op4 · op3 · op2 · ~op1 · op0
lw     = op5 · ~op4 · ~op3 · ~op2 · op1 · op0
sw     = op5 · ~op4 · op3 · ~op2 · op1 · op0
beq    = ~op5 · ~op4 · ~op3 · op2 · ~op1 · ~op0
```


控制器的实现示意图



处理器的设计步骤



- ① 分析指令，得出对数据通路的需求 ✓
- ② 为数据通路选择合适的组件 ✓
- ③ 连接组件建立数据通路 ✓
- ④ 分析每条指令的实现，以确定控制信号 ✓
- ⑤ 集成控制信号，形成完整的控制逻辑 ✓

本节小结



控制信号的集成

北京大学·慕课

计算机组成

制作人：陆俊林

