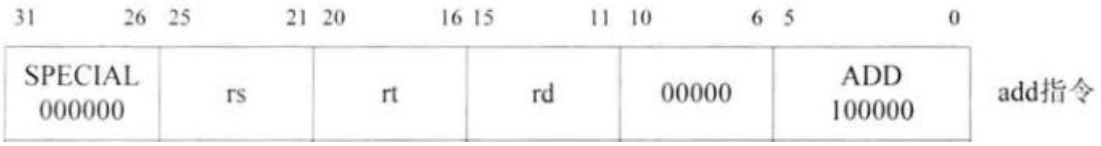
1、add rd,rs,rt

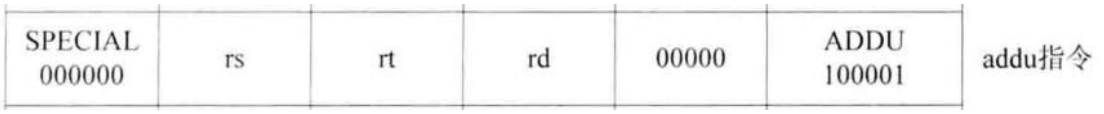


- 当功能码是 6'b100000 时，表示 add 指令，加法运算。

指令用法为：add rd, rs, rt。

指令作用为：rd <- rs + rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行加法运算，结果保存到地址为 rd 的通用寄存器中。但是有一种特殊情况：如果加法运算溢出，那么会产生溢出异常，同时不保存结果。

2、addu rd,rs,rt



- 当功能码是 6'b100001 时，表示 addu 指令，加法运算。

指令用法为：addu rd, rs, rt。

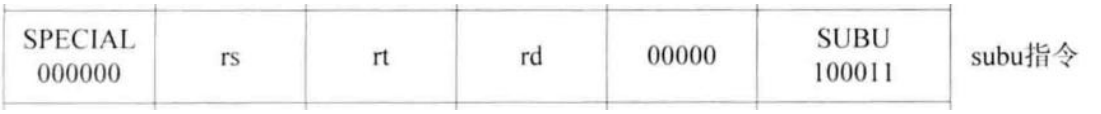
指令作用为：rd <- rs + rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行加法运算，结果保存到地址为 rd 的通用寄存器中。与 add 指令的不同之处在于 addu 指令不进行溢出检查，总是将结果保存到目的寄存器。

3、sub rd,rs,rt

指令用法为：sub rd, rs, rt。

指令作用为：rd <- rs - rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行减法运算，结果保存到地址为 rd 的通用寄存器中。但是有一种特殊情况：如果减法运算

4、subu rd,rs,rt

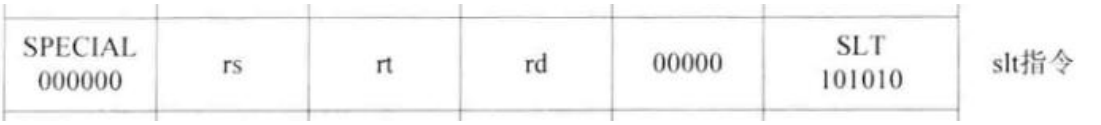


- 当功能码是 6'b100011 时，表示 subu 指令，减法运算。

指令用法为：subu rd, rs, rt。

指令作用为：rd <- rs - rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行减法运算，结果保存到地址为 rd 的通用寄存器中。与 sub 指令的不同之处在于：subu 指令不进行溢出检查，总是将结果保存到目的寄存器。

5、slt rd,rs,rt



- 当功能码是 6'b101010 时，表示 slt 指令，比较运算。

指令用法为：slt rd, rs, rt。

指令作用为：rd <- (rs < rt)，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值按照有符号数进行比较，如果前者小于后者，那么将 1 保存到地址为 rd 的通用寄存器中；反之，将 0 保存到地址为 rd 的通用寄存器中。

6、sltu rd,rs,rt

SPECIAL 000000	rs	rt	rd	00000	SLTU 101011	sltu指令

- 当功能码是 6'b101011 时，表示 sltu 指令，比较运算。

指令用法为：sltu rd, rs, rt。

指令作用为：rd <- (rs < rt)，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值按照无符号数进行比较，如果前者小于后者，那么将 1 保存到地址为 rd 的通用寄存器中；反之，将 0 保存到地址为 rd 的通用寄存器中。

7、and rd,rs,rt

SPECIAL 000000	rs	rt	rd	00000	AND 100100	and指令

- 当功能码是 6'b100100 时，表示是 and 指令，逻辑“与”运算。

指令用法为：and rd, rs, rt。

指令作用为：rd <- rs AND rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行逻辑“与”运算，运算结果保存到地址为 rd 的通用寄存器中。

8、or rd,rs,rt

SPECIAL 000000	rs	rt	rd	00000	OR 100101	or指令

- 当功能码是 6'b100101 时，表示是 or 指令，逻辑“或”运算。

指令用法为：or rd, rs, rt。

指令作用为：rd <- rs OR rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行逻辑“或”运算，运算结果保存到地址为 rd 的通用寄存器中。

9、xor rd,rs,rt

SPECIAL 000000	rs	rt	rd	00000	XOR 100110	xor指令

- 当功能码是 6'b100110 时，表示是 xor 指令，异或运算。

指令用法为：xor rd, rs, rt。

指令作用为：rd <- rs XOR rt，将地址为 rs 的通用寄存器的值与地址为 rt 的通用寄存器的值进行逻辑“异或”运算，运算结果保存到地址为 rd 的通用寄存器中。

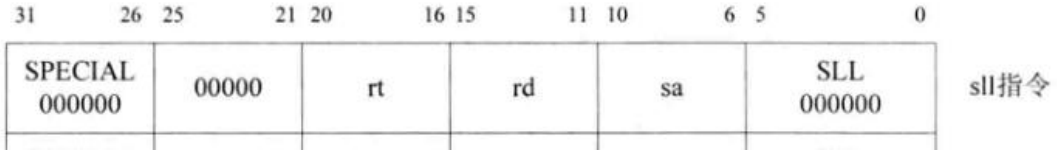
10、nor rd,rs,rt

SPECIAL 000000	rs	rt	rd	00000	NOR 100111	nor指令

指令用法为：nor rd, rs, rt。

指令作用为：rd <- rs NOR rt，将地址为 rs 的通用寄存器的值，与地址为 rt 的通用寄存器的值进行逻辑“或非”运算，运算结果保存到地址为 rd 的通用寄存器中。

11、sll rd,rt,sa

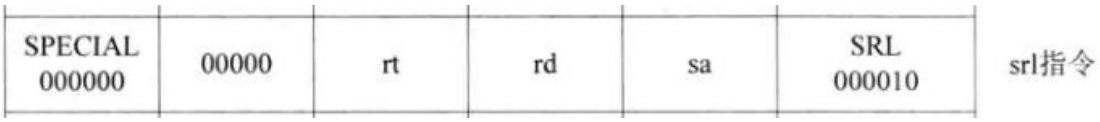


- 当功能码是 6'b000000，表示是 sll 指令，逻辑左移。

指令用法为：sll rd, rt, sa。

指令作用为：rd <- rt << sa (logic)，将地址为 rt 的通用寄存器的值向左移 sa 位，空出来的位置使用 0 填充，结果保存到地址为 rd 的通用寄存器中。

12、srl rd,rt,sa

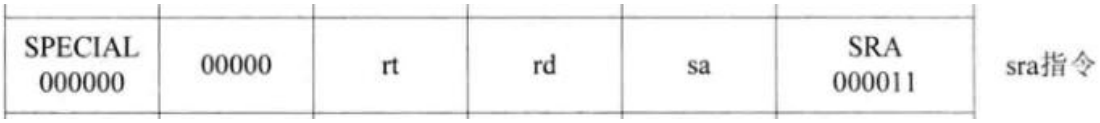


- 当功能码是 6'b000010，表示是 srl 指令，逻辑右移。

指令用法为：srl rd, rt, sa。

指令作用为：rd <- rt >> sa (logic)，将地址为 rt 的通用寄存器的值向右移 sa 位，空出来的位置使用 0 填充，结果保存到地址为 rd 的通用寄存器中。

13、sra rd,rt,sa

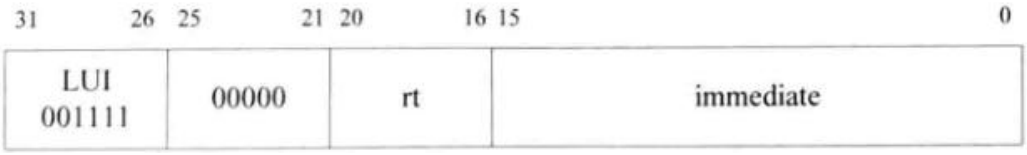


- 当功能码是 6'b000011，表示是 sra 指令，算术右移。

指令用法为：sra rd, rt, sa。

指令作用为：rd <- rt >> sa (arithmetic)，将地址为 rt 的通用寄存器的值向右移 sa 位，空出来的位置使用 rt[31]的值填充，结果保存到地址为 rd 的通用寄存器中。

14、lui rt,immediate



指令用法为：lui rt, immediate。

指令作用为：rt <- immediate || 0^{16}，将指令中的 16bit 立即数保存到地址为 rt 的通用寄存器的高 16 位。另外，地址为 rt 的通用寄存器的低 16 位使用 0 填充。