Code to Circuits

How computer programs become electrical signals

C Code

```
size_t strlen(const char *string)
{
        size_t  len = 0;

        while(*string++ != 0)
        {
                len ++;
        }
        return len;
}
```

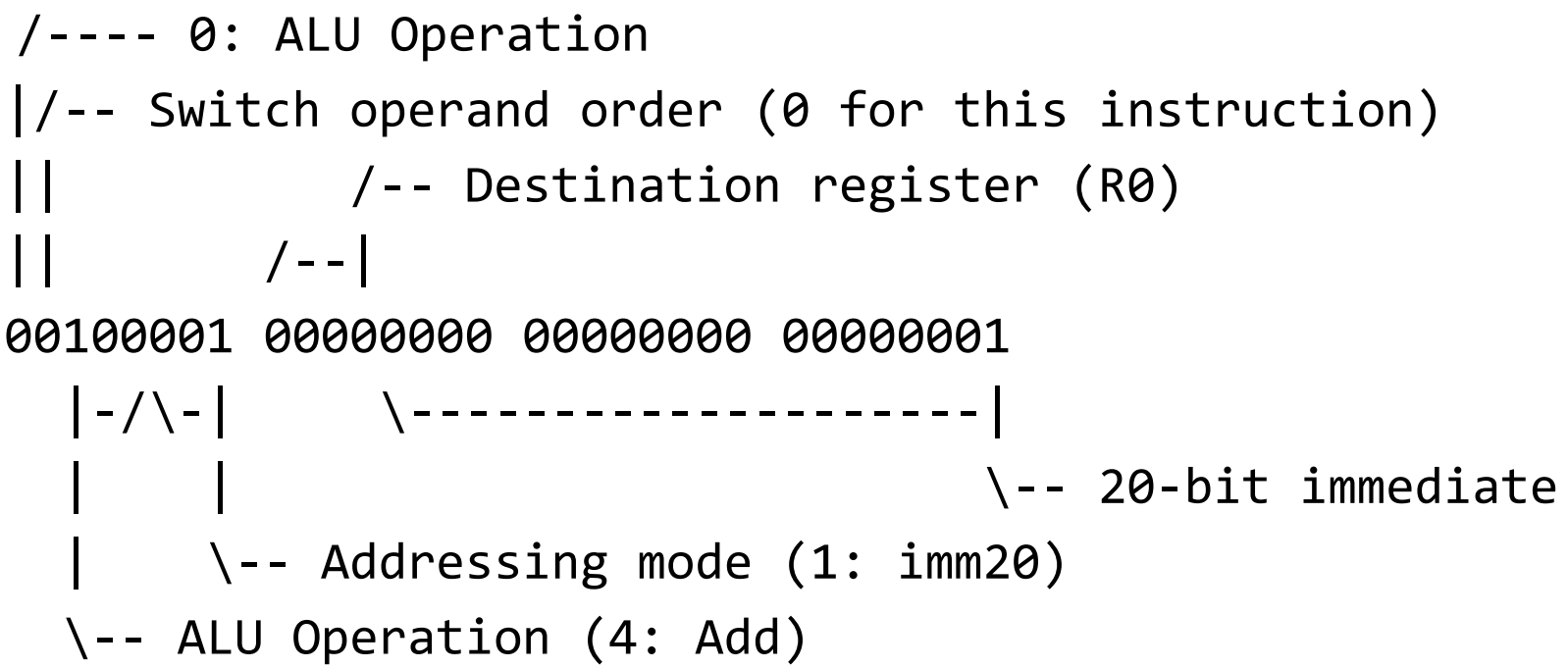
Assembly

```
; R0: string
strlen:
      mov r1, 0   ; len = 0
.loop
      ldrbz r2, [r0]
      and r2, r2
      jz .end
      add r0, 1   ; string++
      add r1, 1   ; len++
      jmp .loop
.end:
      mov r0, r1  ; return len
      jmp lr
```

Binary

```
strlen:
      01100000   ; MOV R1, 0x0
.loop:
      C6200000   ; LRDBZ R2, [R0+0x0]
      38220000   ; TEST R2, R2
      80000003   ; JMP.Z .+3*4 (to .end)
      21000001   ; ADD R0, 1
      21100001   ; ADD R1, 1
      8EFFFFFA   ; JMP.AL .-6 (to .loop)
.end: 00010000   ; MOV R0, R1
      A0EE0000   ; JMP.AL R14 (Always, LR)
```

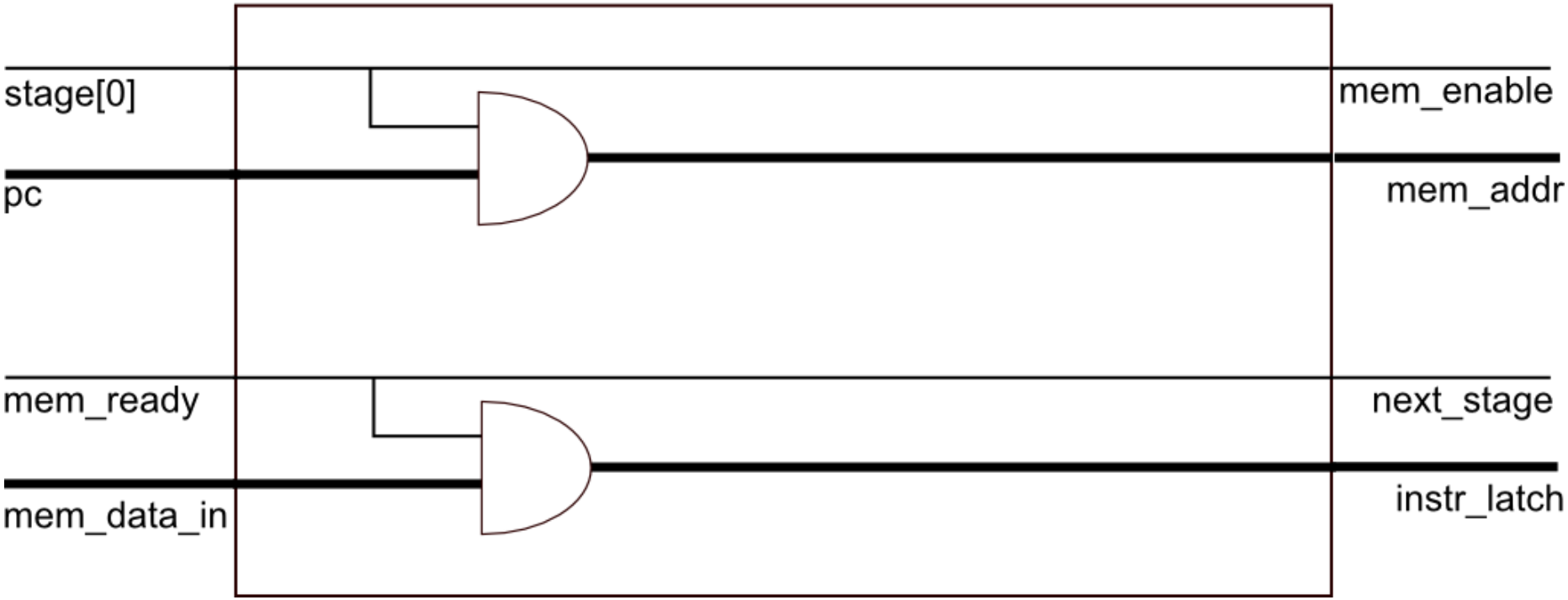
Instruction Format (ALU)



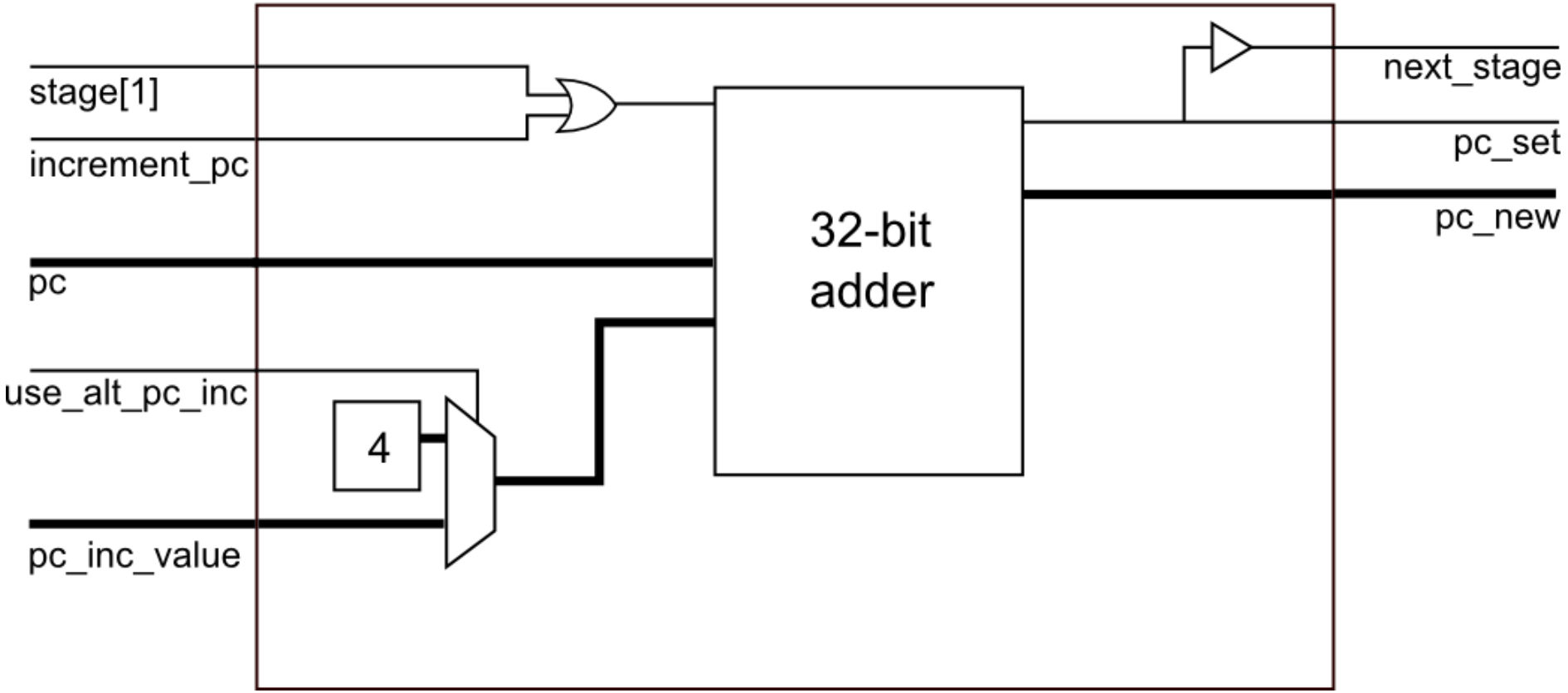
CPU Stages

- Instruction load
- Increment the instruction pointer
- Instruction decode and operand loading
- Actual calculation
- Operand writeback

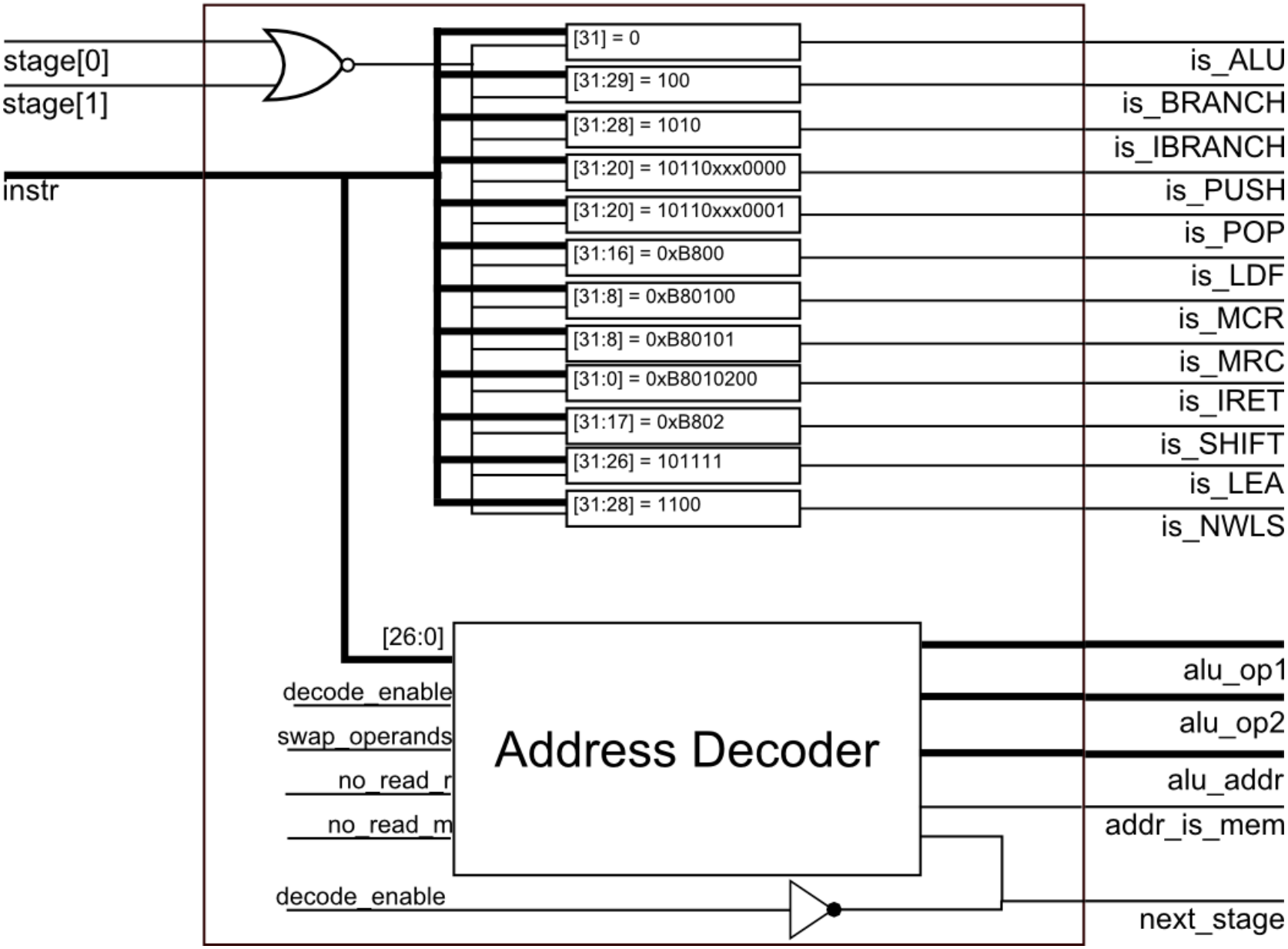
Stage #1: Instruction Read



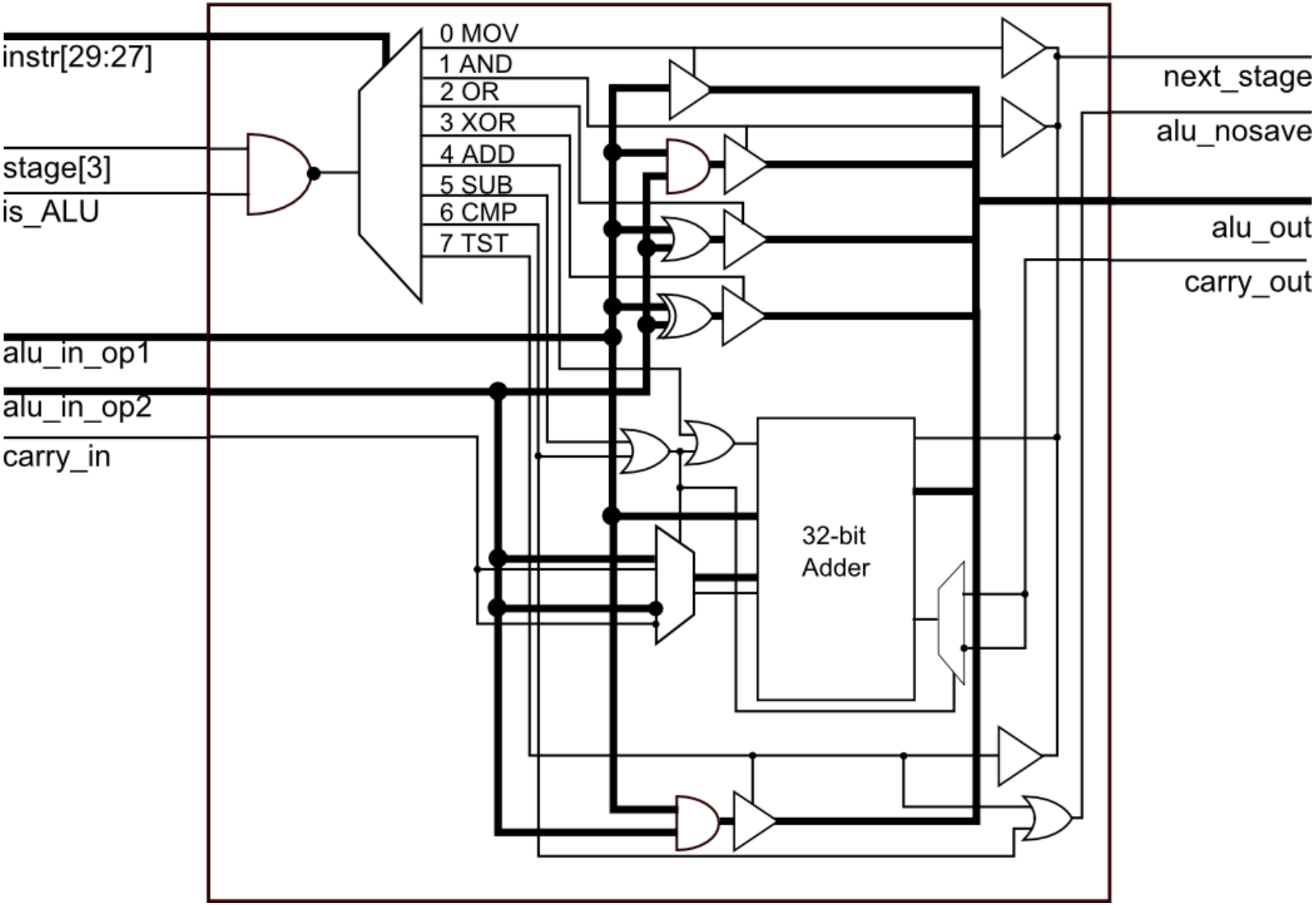
Stage #2: Increment PC



Stage #3: Decode



Stage #4: ALU Operate



Stage #5: ALU Save