In More Depth: Jump Tables

One way to implement *switch* or *case* statements is via a sequence of conditional tests, turning the *switch* statement into a chain of *if-then-else* statements. But sometimes the alternatives may be efficiently encoded as a table of addresses of alternative instruction sequences, called a *jump address table*, and the program needs only to index into the table and then jump to the appropriate sequence. The jump table is then just an array of words containing addresses that correspond to labels in the code. The program loads the appropriate entry from the jump table into a register, and then it jumps to the proper address using a jump register

2.10 [20] <§2.5> This C version of a *case* statement is called a *switch* statement. The following C code chooses among four alternatives depending on whether `k` has the value 0, 1, 2, or 3.

```
switch (k) {
  case 0:f = i + j; break; /* k = 0 */
  case 1:f = g + h; break; /* k = 1 */
  case 2:f = g - h; break; /* k = 2 */
  case 3:f = i - j; break; /* k = 3 */
}
```

Assume the six variables `f` through `k` correspond to six registers `$s0` through `$s5` and that register `$t2` contains 4. What is the corresponding MIPS code? [Hint: use the *switch* variable `k` to index a jump address table, and then jump via the value loaded. We first test `k` to be sure it matches one of the cases ($0 \leq k \leq 3$); if not, the code exits the *switch* statement.]

2.11 [20] <§2.5> For the *switch* statement shown in Exercise 2.10, do the following:

a. Implement the switch in C using *if-else* statements.

b. Implement this C code in MIPS. Make the same assumptions about register placement as in Exercise 2.10.

c. Assume the CPI for MIPS instructions as shown in Exercise 2.50. Given your MIPS code for Exercise 2.10 and your code for this exercise, which implementation of the *switch* statement is faster? Assume a worst-case scenario where the last *case* statement is the valid one.

2.12 [20] <§2.5> Assume the CPI for MIPS instructions as shown in Exercise 2.51. Given a case statement with *N* cases, when is the use of jump tables faster than chained conditional jumps? Assume a worst-case scenario where the last *case* statement is the valid one, and assume that each case contains only one arithmetic operation.