

### Instruction scheduling

1. Assume that your machine has two ALUs, which can execute ADDs in a single cycle, SUBs in two cycles, and *one* of the two ALUs can execute MULs in two cycles. There is also a single LD/ST unit. LOADs take up the LD/ST for two cycles, while STs take up the LD/ST for one cycle. The LD/ST unit can also perform ADD instructions in a single cycle. Draw the reservation tables for this machine.
2. Draw the scheduling DAG for the following program:
  1. LOAD(A) R1
  2. LOAD(B) R2
  3. LOAD(C) R3
  4. LOAD(D) R4
  5. R5 = R1 + R2
  6. R6 = R5 \* R3
  7. R7 = R1 + R6
  8. R8 = R6 + R4
  9. R9 = R5 + R7
  10. R10 = R9 + R8
  11. ST(R10) E;
3. Give the schedule you obtain after performing height-based list scheduling on the above code.

### Loop transformations

For the following problems, consider the code below:

1. X = 2;
2. Y = 10;
3. if (X < Y) goto 13
4. A = Y + 10;
5. B = X \* 4 + A;
6. Z = 10;
7. if (B < Z) goto 11
8. D = Y + Z \* 2;

```
9.      Z = Z - 1;
10.     goto 7;
10.    X = X + 2;
12.     goto 3;
13.    Y = D;
14.    halt;
```

1. Draw the CFG for the code above. Identify the loops in the code.
2. Which statements are loop invariant? Can they be moved outside their enclosing loop? Show the code that results after hoisting any loop invariant code outside the loop.
3. Identify the induction variables in this code. Show the code that results after performing any possible strength reduction.
4. Show the code after performing any possible linear test replacement.