Module 7

Parallelism

How fast can a program be run on a processor with instruction-level parallelism?

- 1. The potential parallelism in the program.
- 2. The available parallelism on the processor.
- 3. Our ability to extract parallelism from the original sequential program.
- 4. Our ability to find the best parallel schedule given scheduling constraints.

Pipelined Execution

	i	$i+1$	$i+2$	$i+3$	$i+4$
1.	IF				
2.	ID	IF			
3.	EX	ID	IF		
4.	MEM	EX	ID	IF	
5.	WB	MEM	EX	ID	IF
6.		WB	MEM	EX	ID
7.			WB	MEM	EX
8.				WB	MEM
9.					WB

Five consecutive instructions in a 5-stage instruction pipeline

With an instruction pipeline, a new instruction can be fetched every clock while preceding instructions are still going through the pipeline.

Code-Scheduling Constraints

- Code scheduling is a form of program optimization that applies to the machine code that is produced by the code generator.
- Code scheduling is subject to three kinds of constraints:
  - Control-dependence constraints. All the operations executed in the original program must be executed in the optimized one.
  - Data-dependence constraints. The operations in the optimized program must produce the same results as the corresponding ones in the original program.
  - Resource constraints. The schedule must not oversubscribe the resources on the machine.

Tradeoff Between Register Usage and Parallelism

```
LD t l , a        // tl = a
ST b, tl          // b = t1
LD t2, c          // t2 = c
ST d, t2          //d =t2
```

The code below copies the values of variables in locations a and c to variables in locations b and d, respectively, using pseudoregisters t1 and t2.

If all the memory locations accessed are known to be distinct from each other, then the copies can proceed in parallel. However, if tl and t2 are assigned the same register so as to minimize the number of registers used, the copies are necessarily serialized.

The reuse of registers, however, serializes the computation.

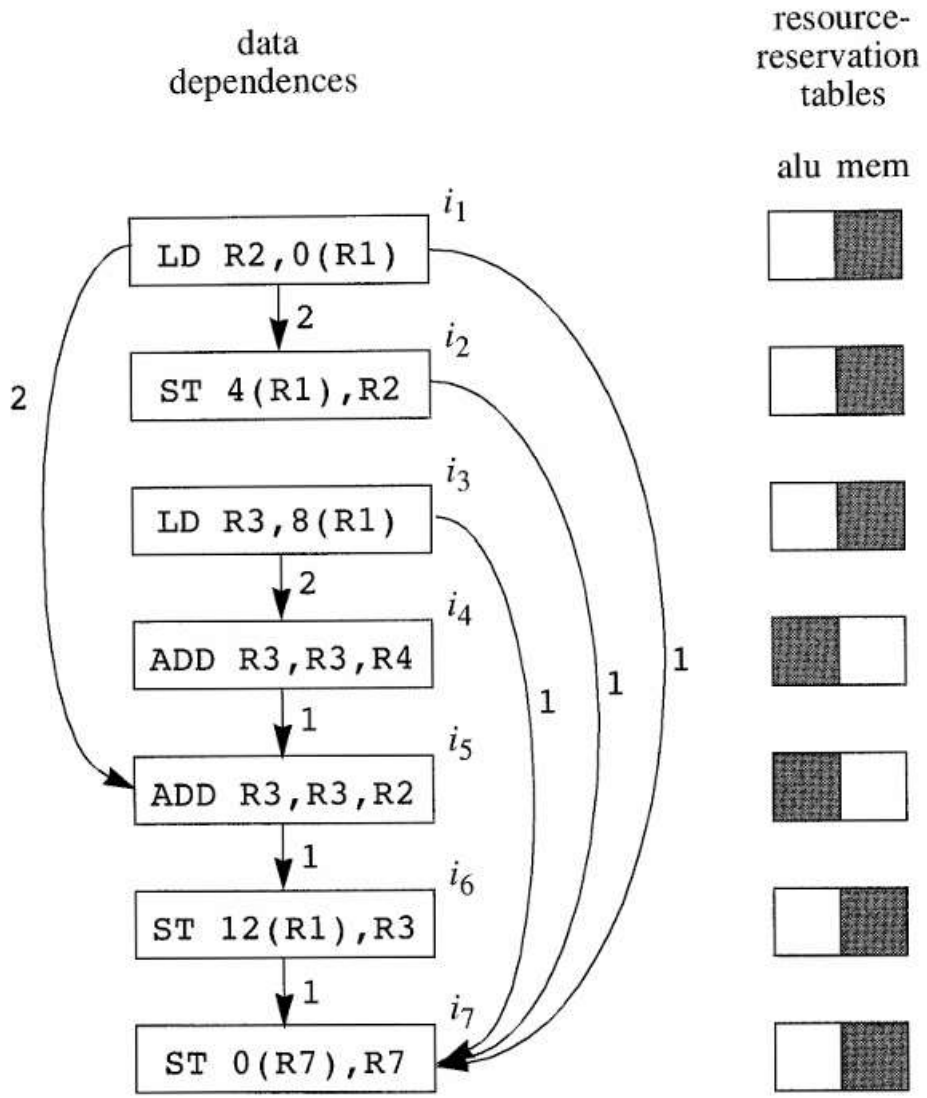
A Basic Machine Model

Many machines can be represented using the following simple model. A machine $M = \langle R, T \rangle$, consists of:

1. A set of operation types T, such as loads, stores, arithmetic operations, and so on.

2. A vector $R = [r_1, r_2, \ldots]$ representing hardware resources, where r_i is the number of units available of the ith kind of resource. Examples of typical resource types include: memory access units, ALU's, and floating-point functional units.

Data-Dependence Graphs



Prepare the problem discussed in class