Concepts Introduced in Chapter 8

- register assignment
- instruction selection
- run-time stack management
- code-generator generators

Tasks Performed by a Code Generator

- Register assignment
- Instruction selection
- Management of the run-time stack

Target Program

The target program can be
- Absolute machine language
- Relocatable machine language
- Assembly language

Register Assignment

- Register assignment is the assignment of temporaries to hardware registers.

Challenges
- Use of register pairs
- Overlapping of registers
- Operations in specific registers
- Spills
Instruction Selection

- Instruction selection is the mapping of the intermediate language operations to machine instructions.
- Have to choose not only the appropriate instructions, but also the addressing mode of each operand.

Implementation of Conditional Branches

- Use condition codes that reflect the result of the last arithmetic operation.
- Use comparison instructions to set a register to zero or a nonzero value based on two values and a relational operator.
- Use comparison instructions to set a predicate register, which contains only a single bit.
- Compare and branch in a single instruction.

SPARC Addressing Modes

<table>
<thead>
<tr>
<th>Name</th>
<th>Assembly</th>
<th>RTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>immediate</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>register</td>
<td>%n</td>
<td>r[n]</td>
</tr>
<tr>
<td>register deferred</td>
<td>[%n]</td>
<td>M[r[n]]</td>
</tr>
<tr>
<td>displacement</td>
<td>[%n+m]</td>
<td>M[r[n]+m]</td>
</tr>
<tr>
<td>indexed</td>
<td>[%n+%m]</td>
<td>M[r[n]+r[m]]</td>
</tr>
</tbody>
</table>

Run-Time Stack Management

- Allocating and deallocating space on the run-time stack when entering and leaving a function.
- Partitioning the register set into scratch and nonscratch sets.
- Saving and restoring nonscratch registers that are used in the function.
- Dedicating registers for managing the run-time stack.
- Passing arguments.
- Responsibility of callee versus caller.
### Evaluation Order of Arguments

• What is printed by the following program?

```c
int g = 0;
int f0() { return g; }
int f1() { return ++g; }
main()
{
    printf("%d %d\n", f0(), f1());
}
```

### Code-Generator Generators

• Generates a code generator from a specification.
• Types of pattern matching
  - Tree rewriting
  - Parsing

### Code Generation by Tree Rewriting

• Each rule is of the form
  
  replacement ← template  { action }

  where replacement is a node
  template is a tree
  action is a code fragment
• Accomplished by a depth-first translation.

### Code Generation by Parsing

• Use an LR parser to accomplish the pattern matching.
• Input tree is represented as a string.