Computer Organization
605.204

Notes from Compiler Design and Construction by Arthur B. Pyster
Van Norstrand Reinhold Company

COMPILER OVERVIEW

TRANSLATOR - a program which takes text written in one language (the Source language), and converts it into equivalent text in a second language (the Target or Object language.)

If the Source is an abstract high-level language, and the Target is a low-level or machine language, then the translator is a COMPILER.

High-level languages such as FORTRAN, Pascal, SNOBOL, and PL/1, and C++ and JAVA give the programmer the ability to express solutions in a notation suited to the problem space much better than in a low level language.

High-level languages custom designed to facilitate solutions in particular applications. Examples:

SNOBOL4 - string processing
GPSS - discrete simulation
FORTRAN - numerical computation

INSIDE THE BLACK BOX

SCANNER

PARSER

INTERMEDIATE CODE GENERATION

SEMANTIC PROCESSOR

OPTIMIZER

CODE GENERATOR

TABLES

Computer Organization
605.204

SCANNER

Converts the stream of characters from the input medium into an internal format suited for use by the Compiler

Identify the TOKENS of the language
Remove extraneous blanks and other white space
Remove comments
Report errors discovered

The SCANNER examines the text, character by character, identifying the TOKENS for the Parser. The SCANNER determines whether the symbol is part of the current TOKEN, extraneous, or the beginning of a new TOKEN.

Depending on the implementation, the SCANNER returns to the Parser
One TOKEN at a time,
All TOKENS in a string separated by a special character, or
All TOKENS as elements in a table.

PARSER

Analyzes the Tokens output by the Scanner, and verifies that the source program satisfies the Rules of the Language.

The Rules of the Language are given in the grammar.

The PARSER verifies the syntax of the language. Is the structure correct ?

Create the Parse tree.

Report errors in structure.

INTERMEDIATE CODE GENERATOR

Takes the output of the Parser and puts into an Intermediate Form so that it can be processed by the remaining parts of the Compiler.

Looks somewhat like assembly language.

QUADRUPLES is one example.

SEMANTIC PROCESSOR

Checks the validity of the code.

Is the MEANING of the statements correct ?

Are the elements of the code correct in the context in which they are found ?

For example:

- Are the variables declared exactly once?
- Are the correct data types specified for each operand ?

Reports errors or
perhaps fixes errors using ‘intelligent’ guesses at what the programmer intended.

OPTIMIZER

Module specifically designed to improve some combination of time and space characteristics of the code.

Register allocation to reduce the number of memory loads and stores.

Move loop invariant computations

Compile-time Arithmetic

CODE GENERATION

Takes the intermediate code and generates Assembly or machine language code.

Highly target machine dependent.

Computer Organization
605.204

TABLES

The algorithms which make up most compilers are largely driven by TABLES which summarize information about the program being compiled and about the source and target languages in general.

SCANNER builds the SYMBOL TABLE

PARSER and SEMANTIC PROCESSOR add additional information.

DIAGNOSTIC TOOLS

No commercial COMPILER is marketable unless it generates informative commentary about each program that it processes.

The COMPILER must give some indication when it encounters an error.

The COMPILER must also make extensive checks for errors

Cross reference symbol map

Profile option to count the times each statement is executed.

Not required by the language definition.

MULTI-PASS COMPILER

Single pass possible.

MULTI-PASS compilers process the source code completely with in each module and then pass it on to the next module.

Allows increased modularity

Improved ability to perform global analysis of the program.

Decreases the space required by the compiler execution

Disadvantage is considerable amount of time used to read and write the input and output of each pass.