Familiarity with gcc

Gnu C compiler (gcc) provides a large number of options (or switches) to the user for controlling the compilation process. This document is a brief introduction to some of the options, particularly to those related to vectorization and parallelization capabilities of gcc.

Outline of Compilation Options in gcc

S N	Option	Effect
1	**-c**	Compile or assemble the source files, but do not link. The linking stage is not performed. The ultimate output is in the form of an object file (.o) for each source file.
2	**-S**	Stop after the stage of compilation and generate assembly code but do not assemble it to produce machine code. The output is in the form of an assembler code file for each non- assembler input file specified. By default, the assembled file name for a source file is made by replacing the suffix .c with .s
3	**-E**	Stop after the preprocessing stage; do not run the compiler proper. The o/p is in the form of preprocessed source code, which is sent to the stdout. Useful if we wish to the effect of the c-preprocessor on our c-source.
4	**-o filename**	This option results in the output being produced in "filename" and applies all output produced by gcc, be an executable file, an object file, an assembler file or preprocessed C code. If -o is not specified, the default is to put an executable file in "a.out", the object file for source.suffix in source.o, its assembler file in source.s, a precompiled header file in source.suffix.gch, and all preprocessed C source on standard output.
5	**-v**	Verbose option : Print (on standard error output) the commands executed to run the stages of compilation. Also print the version number of the compiler driver program and of the preprocessor and the compiler proper.
6	**@file**	Read command-line options from file. The options read are inserted in place of the original @file option. If file does not exist, or cannot be read, then the option will be treated literally, and not removed. Options in file are separated by whitespace. A whitespace character may be included in an option by surrounding the entire option in either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional @file options; any such options will be processed recursively.
7	**-Oc**	Enables optimization to be performed by gcc. The character 'c' may be one of {null, 0, 1, 2, 3, s}. The option -O0, where c is null is the default and informs gcc not to enable optimizations. The option -O1 or -O are identical and few optimizations are enabled with this switch. The option -O2 leads to more vigorous optimizations to be perofrmed. The option -O3 is the highest level of optimization permissible. The option -Os enables optimizations for space. Use man gcc or gcc -v option to get details of optimizations that are enabled with each switch.
8	**-fdump-tree-all**	Shows the dump of compilation passes. The number of dumps vary depending on the other options such as optimization levels. Without optimization, this option displays the dumps of a few passes. Among the ones listed, the following 5 are often useful, i) source.c.004t.gimple which gives the equivalent gimple code after the 4th pass, gimple is the Intermedaie Representation (IR) used by gcc, ii) source.c.014t.cfg which is the control flow graph constructed from gimple code in the 14th pass, iii) source.c.018t.ssa which gives ssa code after the 18th pass, iv) source.c.149t.optimized which is the optimized code produced after the 149th pass, and v) source.c.232t.cfg which gives statistics about the optimizations performed by gcc. With -O2 the compiler produces a large number of dumps as more passes are enabled in this option.

Data Dependence Analysis	Pass Variable name	pass_check_data_deps
	Enabling switch	-fcheck-data-deps
	Dump switch	-fdump-tree-ckdd
	Dump file extension	.ckdd

Vector Code Generation	Pass Variable name	pass_vectorize
	Enabling switch	-ftree-vectorize
	Dump switch	-fdump-tree-vect
	Dump file extension	.vect

Parallel Code Generation	Pass Variable name	pass_parallelize_loops
	Enabling switch	-ftree-parallelize-loops=n
	Dump switch	-fdump-tree-parloops
	Dump file extension	.parloops

Loop Distribution for Parallel loop Generation	Pass Variable name	pass_loop_distribution
	Enabling switch	-ftree-loop-distribution
	Dump switch	-fdump-tree-ldist
	Dump file extension	.ldist

S. N.	**Useful Combination of gcc options**	**Output(s) produced by gcc**
1.	-O2 -fverbose-asm -S	Assembled code in verbose mode with optimizaion level O2 enabled. The executable is not created.
2.	-O2 -fverbose-asm -S -fdump-tree-all	Dumps of various passes alongwith the dumps given out by option 1 above.
3.	-O2 -fcheck-data-deps -fdump-tree-ckdd-all	The data dependence analysis performed by gcc and the result is saved in source.c.103t.ckdd
4.	-O2 -fcheck-data-deps -fdump-tree-ckdd-all -ftree-vectorize -fdump-tree-vect-all -msse4 -fverbose-asm -S	Generate pseudo vector code in source.c.113t.vect along with data dependence analysis. Also generates the vector assembly code for the pentium machine sse4 in the file, source.s. While vectorize enables the data dependence pass implicitly, to generate the dump file source.c.103t.ckdd, both the options for data dependence analysis are required.
5.	-O2 -fcheck-data-deps -fdump-tree-ckdd-all -ftree-parallelize-loops=4 -fdump-tree-parloops -ftree-loop-distribution -fdump-tree-ldist -fverbose-asm -S	Generate omp parallel intermediate code in source.c.118t.parloops Also generates the assembly code with omp parallel threads in file source.s. While parallelize loop option enables the data dependence pass implicitly, to generate the dump file source.c.103t.ckdd, both the options for data dependence analysis are required. The loop distribution option attempts to parallelize loops using loop

S. N.	Useful Combination of gcc options	Output(s) produced by gcc
		distribution transformation and the result of its application is saved in file, source.c.104t.ldist
6.	-O2 -ftree-vectorize -fdump-tree-vect-all -msse4	To generate executable vector code with minimal dumps.
7.	-O2 -ftree-parallelize-loops=4 -fdump-tree-parloops -ftree-loop-distribution -fdump-tree-ldist	To produce executable parallel OMP threads with minimal dumps.

Illustrations : Done in the class with simple loop in a C program