

responding diagnostic tracing functions, said macros including expansion enabling instructions and tracing instructions, associating a corresponding name string with each macro, selecting by command line arguments particular macros to be enabled and disabled and particular functions to be traced, turning selected macros on and off using its corresponding associated name string at run time and running the program.

2. A method of debugging a program comprising the steps of selectively tracing the program at run time by means of command line arguments without recompiling, including the steps of:

defining each of a plurality of macros to perform selected debugging functions, providing a sub definition in the form of a name string corresponding to each defined macro, and locating the macros in a corresponding file of the program;

providing trace instructions and locating the instructions in a corresponding macro;

defining a series of argument processing instructions and locating them in a corresponding instruction file;

compiling the program with the macros and linking the argument processing instructions therewith to produce a runnable program under test and selectively enabling or disabling the code produced by the compiler from the macros using the name string for each corresponding macro to trace selected portions of program under test by means of selective use of command line arguments.

3. The method of claim 2 wherein the macros occupy unchanging memory locations and effect only execution speed of the program.

4. The method of claim 2 wherein the macros do not effect other memory locations in the program by virtue of their being selectively enabled or disabled.

5. The method of claim 2 further comprising the steps of selectively expanding the macros at compile time in accordance with the sub definitions by selectively defining the macros in a master definition in a header file, the presence or absence of the master definition selectively enables or disables the expansion of all the macros accordingly and compiling the program.

6. A method of debugging a program at run time without recompiling the program comprising the steps of:

defining macros by name and function;  
providing at least two sub definitions of each defined macro corresponding to when a symbol corresponding to the macro name is defined or not;  
incorporating call macro instructions in the program to be debugged;

providing instructions for enabling and disabling the macros by means of command line arguments;  
selectively enabling and disabling the macros from the command line arguments by using the macro name, and running the program.

7. The method of claim 6 comprising expanding the macros in accordance with whether a symbol is defined or undefined to produce respective operation code and empty code and thereafter compiling the program.

8. The method of claim 6 further including providing select and deselect instructions in the form of a definition for each macro in a header file and providing in-

structions to permanently but reversible deselect all the macros from the program to be debugged so that when called the macros expand to empty code and thereby

have no effect on the execution of the program to be debugged.

9. The method of claim 6 wherein the macros include instructions to trace selected routines.

10. The method of claim 6 wherein the macros include instructions to count the execution of selected routines.

11. A computer program for debugging at run time a program to be tested comprising:

macro means for defining in abbreviated form corresponding names for a plurality of sets of corresponding instructions;

name string means associated with each set of instructions;

debugging routine means corresponding to each set of instructions, each debugging routine means performing a selected debugging function of a corresponding macro;

selecting means for selecting and deselecting the debugging routine means by defining and undefining the corresponding macros by means of the name string associated with the corresponding set of instructions;

processing means for processing arguments in the form of the name strings from a command line in the program;

compiler means for compiling the macros, the selecting means and the processing means for producing compiled output; and

means for linking the debugging routine means and the compiled output.

12. The apparatus of claim 11 further including counter means for counting the execution of routines.

13. In a process for debugging a computer program from the command line at run time using a debugging program comprising the steps of

preparing source files for the debugging program including inserting macros for tracing and counting routines, argument processing code, a symbol table code and a master definition for enabling the macros;

naming the macros by means of a name string;  
creating a program under test by compiling the computer program with the argument processing code and macros to form a program link module and linking the program link module with the symbol table;

selectively enabling and disabling macros by name from the command file; and

running the program under test with selected macros enabled and disabled and examining the results of the macros for errors and debugging the program where appropriate.

14. The method of claim 13 further comprising selectively enabling and disabling such named macros by name prior to recompiling the program to enable and disable all such named macros throughout the program.

\* \* \* \* \*