

# CHAPTER 4

---

## Component Programs

Paralogic's *BERT77* application consists of several component programs. Command line options for *BERT* are given below. A complete list of options can be displayed for *BERT* by typing `bert` at the command line. See the *BERT 77 Users Guide* for more information.

### BERT 77 Toolset Options

#### *BERT77* Command Line Syntax

Each of the *BERT* tools accept a common set of options. The syntax for the command lines of all of the *BERT* tools is as follows:

```
bert filename [-options]...
```

When executing `bert`, all the component analysis programs are called automatically. *BERT 77* creates a new directory called `par`. In addition to other information files, the new FORTRAN source files and a makefile are placed in this directory.

#### *BERT* Command Line Options

```
bert. Version 1.03
Copyright (C) 1991-1997 Paralogic, Inc., BERT Group
Usage : bert [ <project> <options> ]
```

Available options:

```
<project>      = <name_of_FORTRAN_file> or \&<name_of_project_file>.
                  Project file contains names of FORTRAN files (one
                  per line);
-m<path>        = directory for BERT database files (bert.tmp by default);
-tg<path>       = directory for generated files (par by default);
-np<N>          = number of processors used in network (8 by default);
```

```
-npmN      = max number of processors in network (default 1024);
-Fc<comp>  = FORTRAN compiler to be used for parallel program
             compilation (f77 by default);
-FS<path>  = get path to time profiling file fun.std
             ($BERT/fun.std by default);
-BL<path>  = get path to BERT low-level library
             ($BERT/libBERT.a by default);

-an        = analyze only and generate full information;

-full      = generate all possible information (three options below);
-info      = generate information about parallel constructions in file
             <project>.PAR;
-ls[e]     = generate scheduling information file in file <project>.SCH
             [extended];
-lr[e]     = generate information about parallel program in file
             <project>.RES [extended];

-pio[e][f] = generate a copy of the user's program with pragma patterns
             [extended][in frame];
-Pn<n>     = generate user's program with pragmas for <n> heaviest
```

---

```

                                levels;
-Pp<p>                        = generate user's program with pragmas for levels having
                                weight at least <p> percentages ( -Pp20 or -Pp0.7 );

-cont<N>                      = change the limit on continuation lines to N;
-extend                      = extend the length of statement fields to column 132;

-autosave                   = all arrays are SAVE;
-call                       = absent modules use and modify only arguments;
-dbnd                      = DO loops can have zero number of iterations;
-op<n>                      = optimization level (0 - 1, default=1);
-subrg                     = subscript ranges may be violated, but not array ranges;
-nosubrg                   = subscript ranges never violated (default);
-round                     = roundoff is important;
-frm                       = parallelize only blocks;
-loop                     = parallelize only DO loops;
-par                       = automatic parallelization on (default);
-nopar                    = automatic parallelization off;
-rw                       = read/write statement may be reordered;
-norw                     = read/write statement may not be reordered (default);

-fs                       = fast scheduling;
-tm[p][s][i]              = insert time profiling statements in text of parallel
                                programs;
                                p = for print on stdout;
                                s = save in file <project>.PRF;
                                i = print integral time. Set default for p and s;
-cb                       = insert BERT comments in text of parallel programs;
-cu                       = save user comments in text of parallel programs;
-vm                       = run-time visualisation of module processing;
-vw                       = run-time visualisation of workers processing;
-ds[-]                   = turn on [turn off] dynamic split of parallel loops
                                (turn on by default);
-ps<size>                 = maximum packet size of transfered data in Kb
                                (default is 4 Kbytes);
-ios<size>                = size of I/O buffer for data transfer in Kb (default
                                is 65 Kbytes);
-im                       = insert IMPLICIT NONE statement in each module.

-savetmp                   = don't remove BERT database directory.

```

## Analysis Programs

The following sections provide details of the various programs in the *BERT77* Toolset. With the exception of BERT.INF, and LOADpvm, all of the programs are called by *BERT77* automatically. Normally there will be no need to execute these programs individually.

BERT.SAN performs syntax analysis of FORTRAN programs.

BERT.DAN constructs IN/OUT sets, searches reductions and estimates times.

BERT.PAN checks parallelism inhibitors and defines parallel segments.

BERT.TAN chooses which parallel segments will be parallelized.

BERT.GEN generates the parallel program source code.

## Building Programs

BERT.MAK generates the a Makefile for new FORTRAN code produced by BERT.GEN. Two executable file will be created by the Makefile; `master` and `worker`.

## Information Gathering Program

### BERT.INF

BERT.INF generates a summary file from the information files generated by *BERT 77*. The summary file is named *filename.f*.INF. Where *filename.f* is the name of the converted file. If a project file was used then the file will begin with *filename.prj*. The informational files generated by *BERT77* are:

*filename.f*.SCH - a description of scheduling information.

*filename.f*.RES - a very detailed summary of the conversion.

*filename.f*.PAR - a summary of the parallel constructs in the file.

NOTE: The "-full" option must be used with `bert` to generate all the required informational files. The available options for BERT.INF are as follows:

```
bert.inf version 1.03.SVR3
Copyright (C) 1991-1993 Paralogic, Inc., BERT Group
Usage: bert.inf [<options>]
Available options:
  -? or h    = for help
  -p <path>  = a place where info-files are placed and
               file *.INF would be placed (. by default).
  -n <#>     = number of heaviest levels
```

## BERT.SUM

BERT.SUM provides a summary of the BERT analysis. The available options are:

```
BERT Analysis Summary version 1.0
Use command line arguments:
  bert.sum <file_name> [<speed-up threshold> <execution time
threshold>]
  <file_name> must be a BERT .PAR file
```

A \*.PAR must be present for BERT.SUM to work. The speed-up threshold is the present speedup above which you want BERT.SUM to report (default is 10%). execution time threshold is the present execution time above which you want BERT.SUM to report. (default is 10%) For instance:

```
bert.sum project.f.PAR 20 20
```

would only examine those blocks of code that contribute more than 20% to the speed-up or 20% to the overall amount of calculations performed.

## Loading and Running a Program

### LOADpvm

LOADpvm loads the host parallel computer with the master and worker programs. One master will be loaded on the root node and n-1 worker programs will be loaded.

The master and worker programs are generated by issuing a make command in the par directory created by *BERT77*. The available options are as follows:

Usage: LOAD [<options>]

Available options:

```
-h           = for help
-m <master> = name of MASTER program (master by default)
-w <worker> = name of WORKER program (worker by default)
-n <#>      = number of nodes (4 by default)
```