DOMINO

User's Guide

Konrad Doll and Bernhard M. Riess

July 12, 1995

Institute of Electronic Design Automation
Technical University of Munich
Professor K. Antreich
Arcisstr. 21, 8000 Munich 2
Tel.: (+49 89) 55 17 43 - 0

Contents

1 Introduction

The program DOMINO [2,3] is a placement tool for integrated circuits. It has been developed at the Institute of Electronic Design Automation, Technical University of Munich. DOMINO is an iterative improvement algorithm that aims at improving solutions produced for example by GORDIAN. It calculates the placement of cells for row-oriented layout styles.

The placement program DOMINO is organized as follows:

- Input of the circuit data and the initial placement
  The input data (net list, cell library, chip geometry) and the initial placement is read from a SPACE database or from a special description of the circuit in Proud format [4].

- Generations
  The iterative process generates a sequence of intermediate placements. Each generation problem is geometrically divided into similar local subproblems. Each subproblem consists of a subset of cells to be rearranged. For that purpose the chip area is represented by an array of overlapping regions. A new placement is generated by meandering through the regions from bottom to top. In each region all cells currently placed inside of it are rearranged.

- Evening the rows
  After the generations the rows can optionally be evened by interchanging cells between adjacent rows.

- Swapping cells in a row
  In this part the sizes of all regions are chosen to contain a part of one row. Therefore in each step only cells in one row are rearranged.

- Output of the results
  Finally the placement result is written to a file and/or back in the SPACE database.

2 First Time Use

If you are using DOMINO for the first time, you may proceed as follows:

1. To get an initial placement start GORDIAN as described in the GORDIAN manual [4]. For a first run you can choose the control file of Figure 1. Note, that this file causes GORDIAN to write the initial placement in the file `init_place.domino`.

2. Generate a control file in the directory `input`. For a first run you can use the control file of Figure 2. In further runs you can add control words described in Section 3.3. For an improved placement add the control word `COST_WINTCELL`.

3. Start DOMINO as described in Section 3.3. The protocol file can be found in the `output` directory.

4. The placement is written in the file `place.`*controlfile*.

Figure 1: Example of a GORDIAN control file

```
DESIGN_STYLE = STANDARDCELL;
DESIGN_FORMAT = PROUD;
DESIGN_FILE = design.circuit_name;
TARGET_SYSTEM = DOMINO;
DYNNET;
END;
```

Figure 2: Example of a DOMINO control file

```
DESIGN_FORMAT = PROUD;
DESIGN_FILE = design.circuit_name;
ROW_EVENER;
END;
```

3 General Instructions for the User of DOMINO

This section describes the necessary commands and parameters to invoke DOMINO. It follows the following convention: words written in typewriter (i.e. `in this style`) must be typed exactly as they appear here, whereas words written in italics (i.e.*in this style*) must be replaced by the actual name used.

3.1 Program call

The program DOMINO can be invoked using the following command:

`domino` *controlfile*

where *controlfile* consists of the name of the file containing the control parameters without the suffix `.do` (see Section 3.3).

The only allowable command line option is `-h`, which dumps the legal control file words to standard output. Section 3.3 was generated in this way.

At the moment of the call, the directory from which the program is invoked must contain the subdirectories `input`, `output`. The control file named *controlfile*`.do` must be in the directory `input`. Within the SPACE-System a directory `daba` is required additionally.

If one of the graphics output options is used, the program XOUTPIPE must be in the current search path.

3.2 Input and Output

This section describes the input necessary for DOMINO and the output generated by the program. DOMINO needs information about the cells to be placed (the dimensions and pin locations of the cells), the net list, some geometrical data of the available placement area (e.g. the number of standard-cell rows or the basic cell grid of the sea-of-gates circuit), some information about the location of the I/O connectors or pad cells and an initial placement.

The data are either read from files describing the circuit in Proud format or from the SPACE database.

The result of the placement is written by DOMINO in the file `place.`*controlfile* and/or to a new SPACE database.

Furthermore, DOMINO generates a protocol file called *controlfile*`.doo` covering the entire run of the program (e.g. wiring length, processing time) in the subdirectory `output` of the circuit directory.

3.2.1 DOMINO with design description in proud format

When DOMINO is used with proud design description, the keyword `PROUD` must be specified as `DESIGN_FORMAT`. The netlist will then be read from the file specified under `DESIGN_FILE`, e.g. `design.`*xy*. A file (e.g. `celllib.`*xy*, the name is taken from the file `design.`*xy*) describing the cell library must also be present in the current directory. If a file `netweight.`*xy* exists, net weights will be read from this file.

The resulting placement is written in a file `place.`*controlfile* (see also [4]).

3.2.2 DOMINO in the SPACE-environment

When DOMINO is used in the SPACE-environment, the keyword `SPACE` must be specified for all of the control words `DESIGN_FILE` and `DESIGN_FORMAT`.

The `daba` directory must contain the following files specifying the circuit data:

glbea.asc	Number of modules, nets and connected pins
glgeo.asc	grid unit
glktg.asc	part of the cell library used in the circuit
glbpl.asc	integer module coordinates, module rotation and placement mark
glbto.asc	number of pins of each module
glkai.asc	width, height, and type of a module
glsin.asc	information about the nets
glgam.asc	dimensions of the placement area, channel capacities, and pad locations
pnlst.asc	net list
mckoo.asc	initial placement: center coordinates of the modules after each level of partitioning of the GORDIAN procedure

The following files in the directory `daba` will be created or changed by DOMINO:

dokoo.asc	center coordinates of the modules
glbpl.asc	new integer module coordinates, module rotation and placement mark

3.3 The Control file

The control file must be present in the subdirectory `input` of the circuit directory. Moreover, to be recognized by DOMINO, its name must end with the suffix `.do`. This subsection contains an alphabetical list of all the control words. Most invalid set of control parameters will cause the program to stop.

A list of legal control words is output if DOMINO ist invoked with the `-h` option.

DESIGN_FORMAT Specifies the way DOMINO retrieves its circuit information. This is a required parameter. Type: choice, one of PROUD, PROUD_L, PRIZE, SOGSCX, SPACE, BOE, XILINX, KLEIN, KLEIN_T, UNKNOWN_DES_FORMAT. Default value: UNKNOWN_DES_FORMAT

DESIGN_FILE Specifies the name of the file that contains the circuit description. If the description is to be read from a SPACE database, you must specify DESIGN_FILE=SPACE. This is a required parameter. Type: string of length 2000. Default value:

INIT_PLACEMENT For DESIGN_FILE=SPACE, the initial placement will be read from the file XXkoo.asc in the daba/ directory. XX is the string you can specify with this command. It must contain two letters. Type: string of length 2000. Default value: mc

ITSTEP_PLACEMENT For DESIGN_FILE=SPACE, the file XXkoo.asc can consist of several placements steps (e.g. GORDIAN writes the placement after each level of partitioning into this file). This command specifies the placement DOMINO will read from this file. Default: log2(number of cells)+2 Type: integer. Default value: -1

READ_ROTATE Rotates chip and modules while reading netlist. Not possible for all design formats. Type: implicit boolean. Default value: FALSE

TWHUNT_OUTPUT Produces an output file for the TimberWolf Hunt. Type: implicit boolean. Default value: FALSE

WRITE_SPACE_DABA Create a complete database using the SPACE format Type: implicit boolean. Default value: FALSE

WRITE_KLEIN_FORMAT Writes a KLEIN-database. Only possible if the initial data was read from KLEIN, too. Type: implicit boolean. Default value: FALSE

BIGPROT	Produce a detailed protocol file of its different actions. Type: implicit boolean. Default value: FALSE
COST_WINTCELL	This control word activates a net model (net model B) for the calculation of the transportation costs, which takes into account the positions of the cells within a region. This net model usually yields better results than the default net model. Type: implicit boolean. Default value: FALSE
ESTIMATE_FEEDS	Estimate the number of necessary feedthroughs during the placement. The number specifies the width of a feedthrough. Can only be done with DESIGN_FORMAT SPACE. Type: integer. Default value: -1
GRAPHIC	Graphic output in X-Windows. If 1, DOMINO waits on a click after each plot of a picture. If 0, DOMINO does not wait. Default: No graphic Type: integer. Default value: -1
LEGAL_PLACEMENT	This controlword has to be set if the initial placement is not a global placement with overlapping cells but a legal placement with the cells already placed in rows. Type: implicit boolean. Default value: FALSE
MAX_GENERATIONS	Specifies the maximum number of generations. Type: integer. Default value: 10
MAX_SWAPINROWS	Specifies maximum number of rearrangements of cells in rows after the generations. Type: integer. Default value: -1
MAXMOD_REGION	Specifies the maximum numbe modules in a region. Type: integer. Default value: 40
MSPTREES	Optimize Minimum Spanning Trees. Type: implicit boolean. Default value: FALSE
OVERLAP_X	Specifies the overlap of two adjacent regions in x-direction in number of rows. Type: integer. Default value: 1
OVERLAP_Y	Specifies the overlap of two adjacent regions in y-direction in units. Default: Maximum height of a cell Type: integer. Default value: -1
PLTSUBCELL	The subcells are plotted after each rearrangement of the cells. Type: implicit boolean. Default value: FALSE

REGION_ITERATION	Specifies the number of rearrangements of cells in each region. Type: integer. Default value: 1
REGIONS_X	Specifies the number of regions in x-direction; Default: half of the number of rows, but at least 2 and at most 30. Type: integer. Default value: -1
REGIONS_Y	Specifies the number of regions in y-direction; Default: Is chosen that each region contains about 40 cells. Type: integer. Default value: -1
ROW_EVENER	After the generations the rows are evened out. Type: implicit boolean. Default value: FALSE
ROW_INDENT	Controls the reduction in the heights of the inner rows relative to the outer rows. (>=0 and <=100) Type: integer. Default value: 0
LEN_STAT	Calculate the estimated wire length after each rearrangement of the cells in a region. Default: the wire length will only be calculated after each complete generation. Type: implicit boolean. Default value: FALSE
TRANS_STAT	Write statistics. Type: implicit boolean. Default value: FALSE
END	Last statement in controlfile Type: implicit boolean. Default value: FALSE

Table 1 illustrates an example of a control file for the calculation of a placement.

References

[1] Konrad Doll, Frank M. Johannes, and Kurt J. Antreich. Iterative placement improvement by network flow methods. *IEEE Transactions on Computer-Aided Design*, 13(10):1190–1200, October 1994.

[2] Konrad Doll, Frank M. Johannes, and Georg Sigl. Domino: Deterministic placement improvement with hill-climbing capabilities. *Proceedings International Conference on Very Large Scale Integration VLSI*, pages 3b.1.1–3b.1.10, 1991.

[3] Konrad Doll, Frank M. Johannes, and Georg Sigl. Probabilistisches Plazieren von Standardzellen: Eine effiziente Alternative zu Simulated Annealing. *Tagungsband des 5. E.I.S.-Workshops, GMD-Studie Nr.188*, pages 244–251, 1991.

[4] Georg Sigl and Konrad Doll. Gordian - users guide. Technical report, Institute of Electronic Design, Technical University of Munich, 1993.

Table 1: Example of a control file:

`DESIGN_FORMAT = SPACE;`	The design description is read from a SPACE database.
`DESIGN_FILE = SPACE;`	
`INIT_PLACEMENT = mc;`	The initial placement will be read from the file `daba/mckoo.asc`.
`ITSTEP_PLACEMENT = 13;`	The 13th step of the coordinates from the file `daba/mckoo.asc`.
`MAX_GENERATIONS = 6;`	The maximum number of generations is 6.
`REGIONS_X = 10;`	The number of regions in x direction is 10.
`REGIONS_Y = 8;`	The number of regions in y direction is 8.
`ROW_INDENT = 2;`	The inner cell rows should be 2 % shorter than the average row length.
`END;`	End of the control file.