SPACE 3D SUBSTRATE EXTRACTION BY USING MAKESUBRES

S. de Graaf

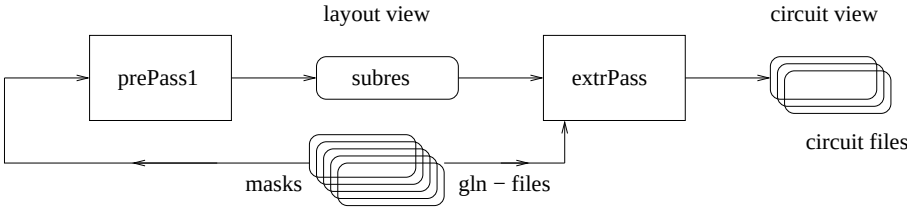
Circuits and Systems Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
Delft University of Technology
The Netherlands

Report EWI-ENS 04-06
August 3, 2004

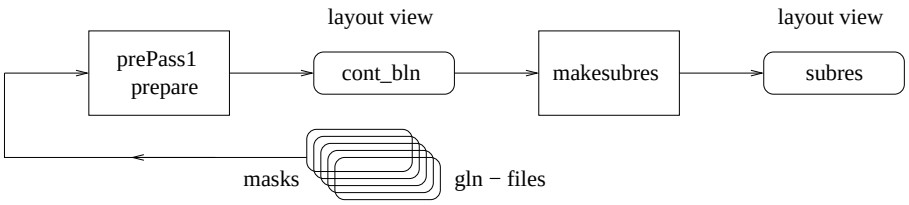
Last revision: August 31, 2005.

1. INTRODUCTION

Normally, the *space3d* layout to circuit extractor uses for accurate (3D) substrate resistance extraction (option **-B**) the *makesubres* program to calculate the substrate resistances. See also the "Space 3D Substrate Extraction Application Note" (report EWI-ENS 03-04). For the calculation, the *makesubres* program uses a substrate spider mesh, Green's functions and Schur matrix inversion. The calculated results are written in the cell layout view "subres" file. In the *space3d* extract pass, the results are read again and the substrate resistances are added to the extracted circuit network. See the figure below.



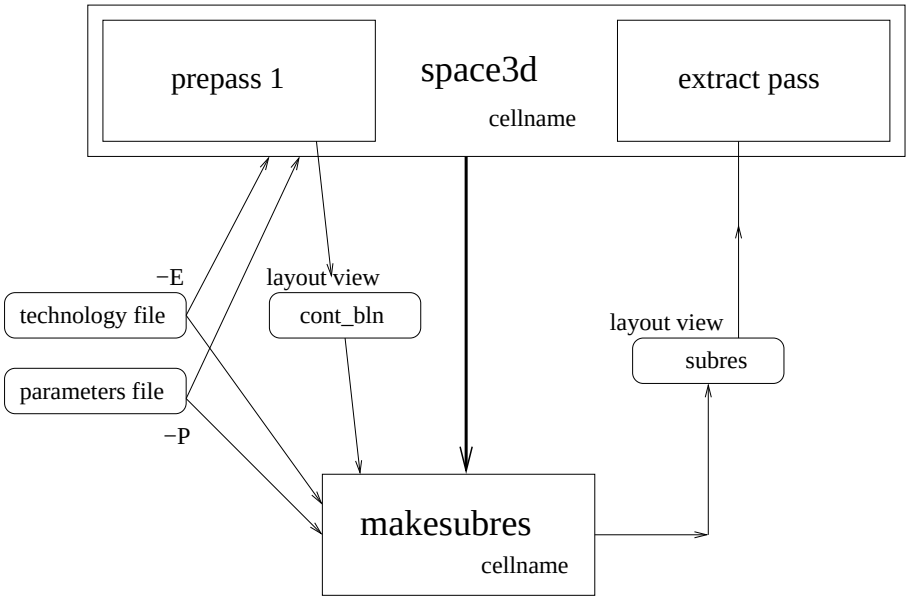
Currently, *makesubres* is a symbolic link to the *space3d* program. In the future, this part may be replaced by an user program. This program must use the same extraction environment, *space* parameter file and technology file and must know the extracted cell name and project path.



Now, as example, an initial *makesubres* user program is created for the calculation of the substrate resistances. The code is taken from *space3d* and a little stripped and changed to make it more easy to understand. For an explanation of the *makesubres* user program, see the following section.

2. THE MAKESUBRES PROGRAM

The *makesubres* program reads the substrate contacts and calculates the substrate conductance values. The calculated substrate conductance values are written to stream "subres". For an overview see the following flow chart:



Note that by the mesh refinement option **-z** also a prepass 0 is used.

In appendix A, a listing is given of a number of files. As example, the cell "sub3term" is used (see the "Space Substrate Res. Extraction User's Manual").

Appendix B gives a flow chart about how the *makesubres* program can be tested.

3. MAKESUBRES OPTIONS

Command line:

```
makesubres [-e def | -E file] [-p def | -P file] [-S param=value] [-Uhv] cell
```

Options explanation:

-e,-E	for specifying a technology file (see *space*)
-p,-P	for specifying a parameter file (see *space*)
-S	for specifying additional parameters (see *space*)
-U	for estimation, no green / schur calculation (see *space*)
-h	for getting help info
-v	for setting verbose mode (equal to **-i**)

4. MAKESUBRES PARAMETERS

parameter	default	comment
debug.print_cap3d_init	off	prints debug information
debug.server_matrix	off	prints debug information
debug.print_matrix	off	prints debug information
debug.print_green	off	prints debug information
debug.print_spider	off	prints debug information
no_green	off	see option **-U**
no_schur	off	see option **-U**
param_verbose	off	prints read parameters
print_time	off	prints used time of program modules
sub3d.be_shape	4	only for pwc setable to 3
sub3d.be_window	--	must be specified
sub3d.edge_be_ratio	1	see documentation
sub3d.edge_be_split	0.5	see documentation
sub3d.edge_be_split_lw	4	see documentation
sub3d.max_be_area	-1	value > 0 needed
elim_sub_node	off	eliminate the substrate node "SUBSTR"
verbose	off	verbose mode, equal to **-v**

Note that "sub3d.be_shape" is always 3 for pwl.
The program does not support parameter "extract_window".

5. GREEN / SCHUR MODULE PARAMETERS

parameter	default	comment
collocation_mode	0	
check_collocation	0	
an_turnover	100	
accelerate_levin	off	
debug.print_green_init	off	
debug.print_green_terms	off	
debug.print_green_gterms	off	
use_adptv_intgrtn	off	
force_adptv_intgrtn	off	
force_mp_anaint	on	
force_mp_exint	off	
use_multipoles	on	
test_multipoles	off	
multipoles_oMaxMpOrder	oMaxMpOrder	
multipoles_cMaxMpOrder	cMaxMpOrder	
sub3d.use_mean_green_values	off	
sub3d.merge_images	on	
sub3d.mp_min_dist	2.0	
sub3d.mp_max_order	2	
sub3d.saw_dist	infinity	in microns
sub3d.edge_dist	0.0	in microns
sub3d.be_mode	0c	
sub3d.use_lowest_medium	on	
sub3d.use_old_images	off	
sub3d.neumann_simulation_ratio	100	
sub3d.point_green	infinity	
sub3d.collocation_green	0	
sub3d.asym_collocation_green	infinity	
sub3d.max_green_terms	500	
sub3d.y_stretch	1.0	
sub3d.green_eps	0.001	
sub3d.col_rel_eps	0.2	
lu	off	schur module
coefficient	off	schur module
print_coef	off	schur module, for debugging

Note that "sub3d.collocation_green" is "infinity" for FeModeGalerkin.

6. PREPASS PARAMETERS

parameter	default	comment
verbose	off	equal to **-v**
no_parasitics	off	also no **-B** (don't specify)
progress_timer	0	gives progress time info
print_time	off	print module usage time
param_verbose	off	print specified parameters
flat_extraction	off	**-B** uses default **-F**
allow_hierarchical_subres	off	when "on" no flat extraction
expand_connectivity	off	when "on" pseudo hier. extraction
no_labels	off	read the "annotations" stream for labels
hier_labels	off	when "on", read "anno_exp" stream
hier_terminals	off	when "on", read "anno_exp" stream
prep_bin	--	when specified, use this program path
simple_sub_extraction	off	equal to **-b** (don't specify)
sub3d_extraction	off	equal to **-B** (specify **-B**)
sub3d.be_window	--	must be specified, used as bandwidth
sub3d.internal	off	don't specify, else no *makesubres*
sub3d.old_mesh	off	don't specify, else no *makesubres*
substrate_extension	10	when cell bbox must be used
max_tan_slice_y	-1	when specified, used by **-z**
max_obtuse	-1	when specified, used by **-z**
low_sheet_res	1	used for distributed sub_terms
omit_unused_sub_term	off	for sub_terms of capacitors
sub_term_distr_xx	off	to specify distributed sub_terms
sep_sub_term	off	to omit joining of touching sub_terms

Note that "@sub" must be specified as last mask in the technology file. When no substrate terminals are specified, the mask is omitted by *tecc*, and no substrate resistances can be extracted.

Note that option **-X** and *Xspace* default use the internal method (no exec of *makesubres*). Also parameter "sub3d.old_mesh" is using the internal method (see the space3d substrate extraction application note 03-04).

Note that distributed sub_terms are only possible, when also interconnect resistances are extracted (see note 03-04).

Note that the used labels and terminals can give additional tile splits and can give more distributed sub_terms. The number of strips used (parameter "sub3d.be_window"), is also responsible for more or less distributed sub_terms.

7. THE COILGEN EXAMPLE

Depending on substrate parameters "sep_sub_term" and "sub_term_distr_*mask*" the files "cont_bln" and "subres" are different. These parameters are only interpreted in the *space3d* prepass. The *makesubres* program must change the number of tiles found in the "cont_bln" file into the correct number of substrate contacts for the "subres" file.

The following table gives some statistics:

mode	number of tiles	number of contacts
sep_sub_term=off, not distr.	34	1
sep_sub_term=on, not distr.	36	4
sep_sub_term=on, distr_m5/m6	126	126

The above given number of tiles does not include the two extra tiles used by the scan function. Note that the extract pass of *space3d* can not use a "subres" file with an incorrect number of substrate contacts.
See for example the following execution sequence:

```
% space3d -%ZB -P params coilgen
% space3d -%2Br -P params coilgen

internal: *** Assertion ‘sc->nr != 0’ failed at src/space/substr/subcont.cc
Abort (core dumped)
%
```

This assertion failure happens, because there are substrate contacts who have not get a substrate contact number. Because these contacts are not matched with "subres" file records. Maybe, the "subres" file is too short.

Note that special option **-%Z** runs only the prepass and runs also the *makesubres* program. Special option **-%2** runs only the extract pass. However, the extract pass is also run with option **-r** and therefor the substrate contacts are differently joined together (because parameter distr. is specified).

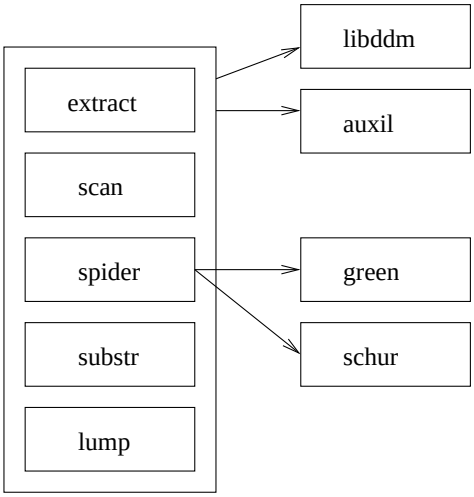
Note that in some cases the end of the "subres" file is not reached. Maybe, because the "subres" file contains more substrate contacts than expected. In that case the following message occurs:

```
internal: *** Assertion ‘next_xl == Inf’ failed at src/space/substr/subcont.cc
Abort (core dumped)
```

8. MAKESUBRES SOURCES

The *makesubres* program contains only the parts of the *space3d* program, which are needed for 3d substrate conductance calculation. Basically, the program consists of the following modulles:

makesubres module overview



The program must be loaded with four external libraries. Library "libddm" for the layout database functions. Library "auxil" for some general used functions. Library "green" for the Green's value calculation. Library "schur" for the Schur matrix inversion. Note that the spider data structs must be used unchanged, because the "green" library is using these data structs.

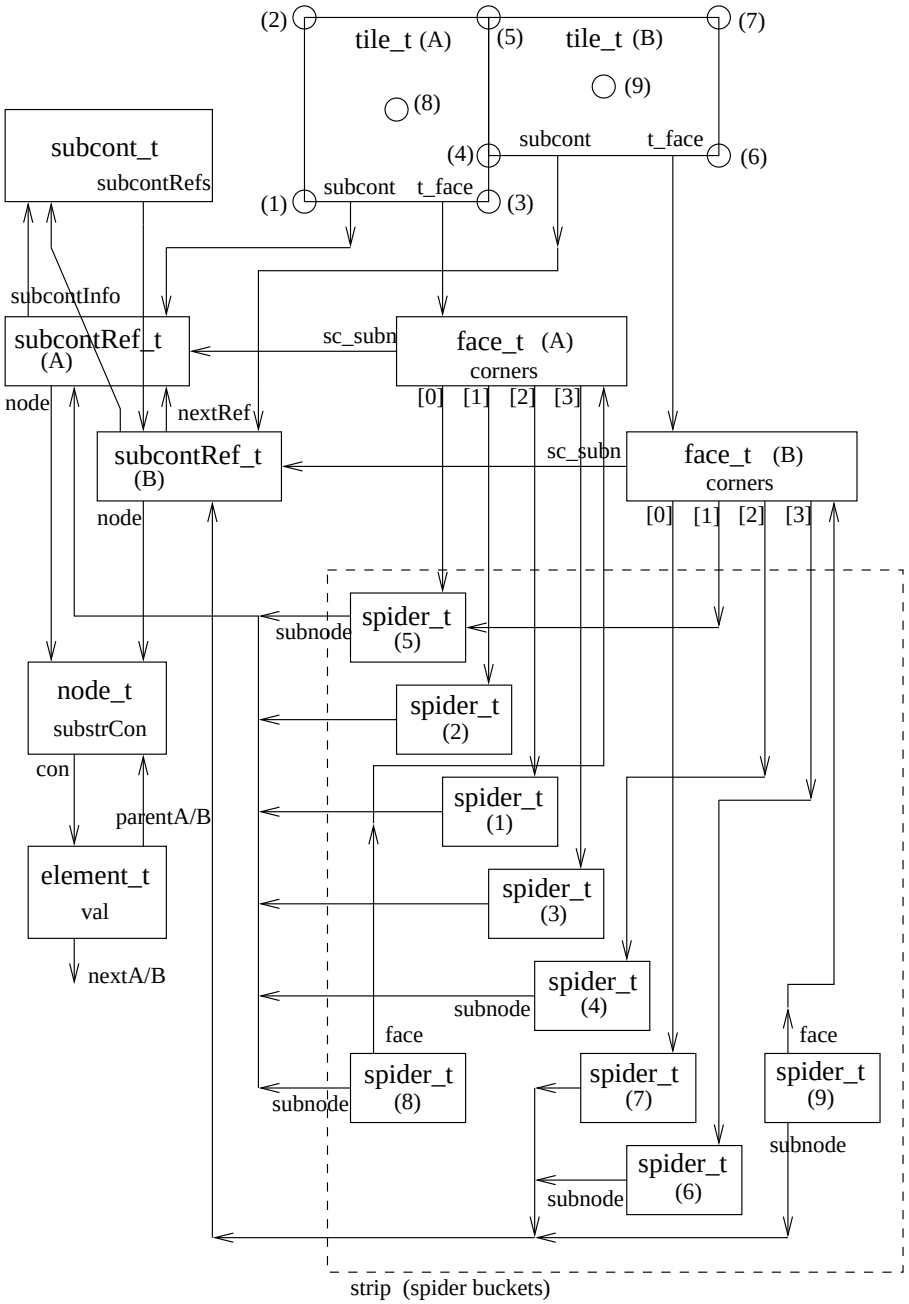
In general, the structure of the *space3d* sources is left unchanged. But superfluous code is left out. And because code is left out, some other modifications are also made. For example, the "spider" module can only handle the 2d mesh used for the substrate, it contains not the 3d code for cap3d extraction. There is not calculated with the act_z values in the program, because the act_z of all spiders is the same (for example spiderDist). The program does not contain code for network generation or reduction.

The program does not use or read the extraction elements from the technology file and does not use the maskdata. It only reads the substrate material data from the tech_file.

The data structs of tile_t and node_t are simplyfied. For data struct subnode_t is used data struct subcontRef_t.

New spiders are directly placed in strip buckets. No spider hash-table is used, but function stripFindSpider uses the strip buckets. I added function stripFreeSpider, to remove superfluous spiders from the strip. Note that no separate LFT_TOP and RGT_BOT spider entries are used. The unused face corner entries are used for this purpose. And each tile points directly to its mesh face.

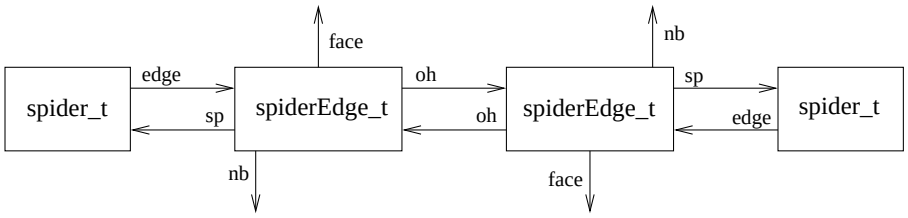
The following figure shows the data struct connections for two tiles.



The corners of the face are set by function meshSetCorners, in function spiderTile, when the tile is ready. The center spiders (8) and (9) are added in function meshRefine, by function meshAddExtraSpider. These spiders must point to the correct subnode. This subnode is taken from face→sc_subn. Note that a conductor element is always placed between two nodes. Note that the use of the node_t data struct can be taken over by the subcont_t data struct.

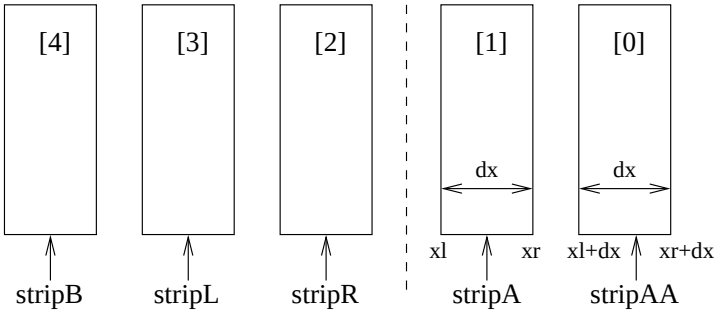
There are two different types of spiders in a strip. The first type are the edge spiders, and the second type the center spiders. Function stripTraverse shall use one of both types, depending on Green mode piecewise linear (pwl) or piecewise constant (variable FeModePwl). Note that only the center spiders point to a face. The center spiders are also used for disposal of the faces, when the strip is freed.

Note that each strip contains also a list of spiderEdges. This data struct spiderEdge_t is not shown in the previous figure. But between two edge spiders are two spiderEdge_t data structs situated. See the following figure.



These spiderEdges are used to walk around the face by piecewise linear mode. Member "nb" is used to go to the next spiderEdge of a spider. Going counter clockwise around a face, the spiderEdge member "face" points to the face. See also the "Space Application Note About Implementing a Green API", report ET-CAS 02-02. This program uses only two types of spiderEdges (CONDUCTOR and INTERNAL).

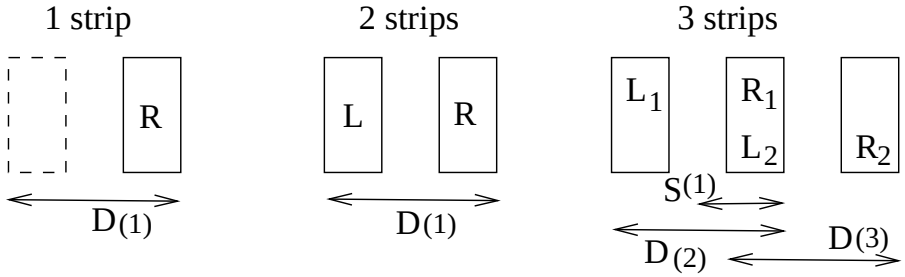
There are in total five strips in the program, see function stripInit. They are used in a circular fashion, see function stripMove. The program starts with filling "stripA".



A spider, spiderEdge and face can only be in one strip. The contents of "stripB" is freed by a call to stripMove and "stripB" becomes "stripL". Function stripMove is called by function sub3dStrip. Function sub3dStrip shall first call meshRefine to refine the faces in "stripR". For piecewise linear all faces must have a triangular shape and no face may have more than 3 edge spiders. Note that different faces can point to the same subContRef (by "sc_subn"). For mesh refinement, see also the "sub3term" example in section 4.8 of the "Space Substrate Resistance Extraction User's Manual".

By the third (or more) call to sub3dStrip shall after refine also a call to doStrip(SINGLE) and doStrip(DOUBLE) be done. Function doStrip shall call stripTraverse for "stripR" or

both "stripR" and "stripL". The stripTraverse calculates the green values for the spiders. The green values calculated for a SINGLE strip are saved and can be reused in DOUBLE mode. Function sub3dStop does for the last strip a meshRefine and a call to doStrip(DOUBLE) and function stripStop shall free all strips. Thus, only by three or more strips doStrip(SINGLE) is done. See the following figure. Note, before meshRefine needs always a stripMove (sub3dStrip call) to be done. This takes place, because each nextStrip x-position is also state ruler scan x-position.



See also "N.P. van der Meijs, Accurate and Efficient Layout Extraction, promotion report 1992", in special section 4.7 "Approximate Inversion of Multiple Band Matrices" and figure 4.11.

Note that the substrate contacts are flushed after clearTile, which calls subContDel. However, clearTile is only called when the tiles are old enough (after bandWidth). Note that this bandWidth is 4 times the strip width.

The substrate contacts are placed in a sorted list (the order is the lower left tile coordinate). They are also outputted in that order to the "subres" file, when they are ready (see subContDel). Each time a new substrate contact is found, function subContNew shall place the contact with add2subcont_list to the list. There is also a subcontRef for the tile to the substrate contact. When two tiles are the same substrate contact, then there are two subcontRefs pointing to the same substrate contact. Function subContJoin is used to merge two substrate contacts. One contact is removed from the list. When all subcontRefs to a substrate contact are removed, the contact becomes ready. Only, when the contact is in the begin of the subcont_list, its data is outputted. By output, each contact gets a sequential number (≥ 1). The conductances to all ready neighbor contacts are also outputted. At last, a "nr_neigh" count is outputted, which is the total number of neighbor connections.

Substrate contacts, which touch each other and are not joined, are placed in the same substrate contact group. A group number (≥ 1) is also outputted for each substrate contact. However, when two substrate contact groups are joined together at a very late stage, it is possible that the wrong group number is already outputted.

For an example of a "subres" file, see appendix A.

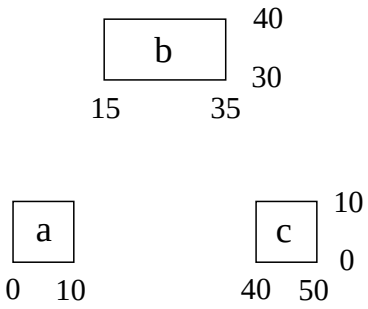
The following table shows the source files of the *makesubres* program. It also shows from which module of *space3d* the file is taken and what the main functions are.

module	source	functions
auxil	auxil.h	macro's: Abs, Max, Min, Null, Round, Swap, strsame
--	define.h	macro's: Y, Slope, compareSlope
--	extern.h	external functions & variables
extract	enumtile.c	enumPair, enumTile, clearTile
extract	gettech.c	getnum, readTechFile, giveICD, getTechnology
lump	lump.c	initLump, endLump, createNode, disposeNode, elemAdd, elemDel
scan	main.c	main, getParams, doCell, die
scan	input.c	openInput, closeInput, fetchEdge
scan	scan.c	scan, scanPrintInfo, bundle, unbundle
scan	slant.c	testIntersection, split
scan	update.c	TL, BR, TR, tileInsertEdge, tileDeleteEdge, tileCrossEdge, tileAdvanceScan, tileStopScan
scan	edge.c	createEdge, disposeEdge, edgeStatistics
scan	tile.c	createTile, disposeTile, tileStatistics
spider	face.c	newFace, disposeFace, faceEnumerate, freeFaces
spider	mesh.c	meshCcwAdjacent, meshCwAdjacent, meshFindEdge, meshMakeEdge, meshSplitEdge, meshSetFace, meshSetFaces, meshSetCorners, isEdgeFace, meshPrintFace
spider	pqueue.c	pqInsert, pqChange, pqEmpty, pqDeleteHead, pqCompare
spider	refine.c	meshRefine, meshRenameFace, splitTriangle, refineTriangle, meshAddExtraSpider, faceRecur, feSize, meshSplitPath, refineTrapezoid, mkTriangle, triangulate
spider	spedge.c	newSpiderEdgePair, disposeSpiderEdge
spider	spider.c	newSpider, disposeSpider, spiderDist
spider	sppair.c	spiderFindNew, spiderPair
spider	sptile.c	spiderTile
spider	strip.c	stripInit, stripMove, stripStop, stripVerbose, freeSpiders, freeEdges, setGreenBuf, stripAddSpider, stripFreeSpider, stripFindSpider, stripAddEdge, inStripA, stripTraverse, ...
spider	sub3d.c (cap3d.c)	mathInit, mathStop, sub3dInitParam, sub3dInit, sub3dStart, sub3dStop, sub3dEnd, sub3dStrip, doStrip, computeMatrixSize, schurStartMatrix, schurStopMatrix, doGreen, computeConductance, schurInput, addQueue, delQueue, schurRowOut, addCon, sub3dStatistics
substr	subcont.c	initSubstr, endSubstr, subContNew, subContJoin, subContDel, subContGroupNew, subContGroupJoin, ...

9. APPENDICES

APPENDIX A -- List of files

layout cell "sub3term"
==========



cont_bln
==========

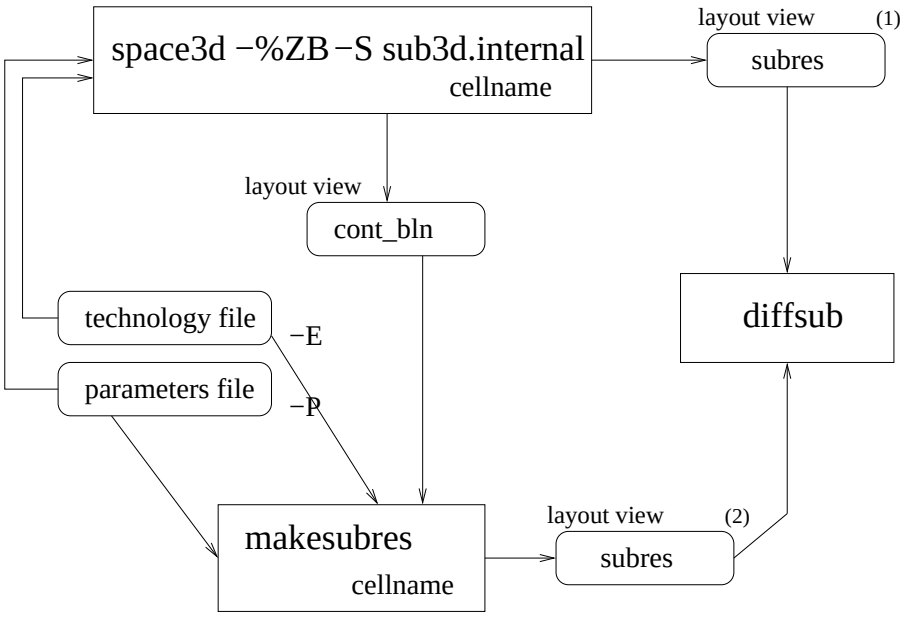
```
% dbcat -vs cont_bln sub3term
=> layout/sub3term/cont_bln
  xl:   xr:   yb:   yt:   ct:
    0    40     0     0     1 (BN 1)
    0    40    40    40  1025 (EN 1)
   60   140   120   120     1 (BN 1)
   60   140   160   160  1025 (EN 1)
  160   200     0     0     1 (BN 1)
  160   200    40    40  1025 (EN 1)
```

subres
==========

```
c 1 1 xl 0 yb 0 g 1.318092e-05
nr_neigh 2
c 2 2 xl 60 yb 120 g 2.022499e-05
nc 1 g 1.472344e-06
nr_neigh 2
c 3 3 xl 160 yb 0 g 1.318092e-05
nc 2 g 1.472344e-06
nc 1 g 7.899695e-07
nr_neigh 2
```

NOTE: coordinates are in internal units (4 * database unit).

APPENDIX B -- Testing makesubres



The result of subres (1) must be equal to subres (2).