TECC PROGRAM APPLICATION NOTE

S. de Graaf

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

Report EWI-ENS 04-13
December 20, 2004

Last revision: March 1, 2013.

1. INTRODUCTION

This application note is intended to give more background information about the *tecc* program (the technology compiler of the *space* system). It tries also to give a more detailed grammar description of the technology file format in BNF (Backus-Naur Form).

I know that this application note is not yet finished. For other technology file information consult the *tecc* manual page and the following *space* documents:

- Space Tutorial
- Space User's Manual
- Xspace User's Manual
- Space 3D Capacitance Extraction User's Manual
- Space Substrate Resistance Extraction User's Manual

2. TECHNOLOGY FILE BNF DESCRIPTIONS

The following BNF conventions are used for the technology file grammar:

```
< >        non terminal symbol
''         one character (terminal)
|          or (one of the items)
#          start of comment
[items]    items 0 or 1 time (optional)
[items]*   items 0 or more times
(items)    items 1 time (grouping)
(items)*   items 1 or more times
```

The technology file parser divides the technology file description in 16 parts.

```
<tech_descr> ::= <header_descr>  # 1
                 <conduc_def>    # 2
                 <tor_def>       # 3
                 <bjt_def>       # 4
                 <junction_def>  # 5
                 <connect_def>   # 6
                 <contact_def>   # 7
                 <cap_def>       # 8
                 <vdim_def>      # 9
                 <eshape_def>    # 10
                 <cshape_def>    # 11
                 <dielec_def>    # 12
                 <sublay_def>    # 13
                 <subcaplay_def> # 14
                 <selfsub_def>   # 15
                 <mutsub_def>    # 16
```

Each of these parts don't need to be present. Thus, an empty technology file is also a valid description. However, when present, each description must be in the above order.

2.1 TECHNOLOGY FILE: header_descr

The items in the technology file header description may be in any order, and each item can appear several times. However, this is not true for items "key_info" and "settings", these items may only be once specified.

```
<header_descr> ::= [<header_item>]*
<header_item>  ::= <key_info>  | <settings> | <unit_def> | <newmask_def>
                   | <color_def> | <resizemask_def> | <wafer_def>
```

2.1.1 Technology file header_descr: key_info

```
<key_info> ::= keys ':' (<layer>)*
             | maxkeys <integer> [<integer> [<integer>]]
```

With the above statement the size and order of the key-list can be specified. Only one of the two specifications is allowed. Note that a "layer" can not be a new mask. This statement is not more so important, because *space* is also using a key hashing method.

2.1.2 Technology file header_descr: settings

```
<settings> ::= set specific_resistance
             | set bem_depth  <value>
```

Note that the bem_depth must be specified, when wafer definitions are specified.

2.1.3 Technology file header_descr: unit_def

```
<unit_def> ::= unit resistance    <value>
             | unit c_resistance  <value>
             | unit s_resistance  <value>
             | unit capacitance   <value>
             | unit a_capacitance <value>
             | unit e_capacitance <value>
             | unit distance      <value>
             | unit vdimension    <value>
             | unit shape         <value>
             | unit layer_depth   <value>
             | unit resize        <value>
```

Note that "c_resistance" is used for contact resistance values and that "s_resistance" is used for specific resistance values. Note that "a_capacitance" is used for area (surface) capacitance values and that "e_capacitance" is used for edge capacitance values.
The "distance" unit is only used for edge capacitance value pairs. The "layer_depth" unit is only used in ‘‘junction_def’’.

2.1.4 Technology file header_descr: newmask_def

```
<newmask_def> ::= new ':' <mask_cond> ':' <mask>
```

The newmask_def defines a new mask name (a name not already in use for a mask).
Note that the ‘‘mask_cond’’ list may only contain standard masks or previous defined new masks. Edge and other-edge masks can not be used in the mask_cond list. Note that standard mask names are specified in the technology "maskdata" file.

2.1.5 Technology file header_descr: color_def

```
<color_def>      ::= colors ':' [<mask_color_def>]*
<mask_color_def> ::= <mask_sub> <color_name>
<color_name>     ::= <name> | <hex_name>
```

The ‘‘name’’ must be found in the X11 "rgb" color database. The "hex_name" specifies directly the pixel values used for (R,G,B) and must be a multiple of 3 hexa-decimal digits. The colors are only used by the *Xspace* and *view3d* programs. Note that the color for mask "@sub" is used for the substrate mesh.

2.1.6 Technology file header_descr: resizemask_def

```
<resizemask_def> ::= resize ':' <mask_cond> ':' <mask> ':' [<value>]
```

An empty ‘‘mask_cond’’ is not a possible condition. A positive "value" specifies a grow and a negative "value" specifies a shrink. When the "value" is left out, a zero value is

taken. The unit of the value is default in meters, use the "resize" unit statement to specify another unit. Note that the ‘‘mask_cond’’ must contain the "mask" specified or else the "mask" must be the name of a new defined mask (see newmask_def). Note that, when a new mask is resized, the new mask becomes a physical mask. It is also possible to create negative masks.

2.1.7 Technology file header_descr: wafer_def

The ‘‘wafer’’ definitions can be used by 3D BEM / FEM substrate resistance extraction.

```
<wafer_def> ::= wafer ':' <mask_cond> ':' <w_values> [':' [<w_options>]*]
<w_values>  ::= <value> <value> <value>
<w_options> ::= viamask '=' <mask>
              | subconn '=' ( on | off )
              | restype '=' ( 'm' | 'p' | 'n' )
```

The first "value" specifies the conductivity in S/m, the second "value" the thickness in micron, and the third "value" the number of layers (≥ 1). Default, no "viamask" is used. But, when specified, only on the place of the via-mask there is the layer stack. The "subconn" is default "on". A substrate connection can only be made, when the wafer reaches the bem_depth. Thus, the bem_depth must be set. The default "restype" is ’p’. Note that the ‘‘mask_cond’’ list may only contain standard masks or previous defined new masks. Edge and other-edge masks can not be used in the mask_cond list.

2.2 TECHNOLOGY FILE: conduc_def

A conductor definition is needed for each element pin or element node. The conductor ‘‘sort’’ is seldom used (the default is "res"). Each conductor must have an unique ‘‘name’’ for identification. The "value" specifies the sheet-resistivity in ohms. An optional carrier-type can be specified (the default type is ’m’).

```
<conduc_def> ::= [ conductors [<sort>] ':' [<conductor>]* ]*
<conductor> ::= <name> ':' <mask_cond> ':' <mask> ':' <value> [':' <type>]
<sort> ::= <name>
<type> ::= 'm' | 'p' | 'n'
```

2.3 TECHNOLOGY FILE: tor_def

A field-effect transistor (e.g. MOS transistor) can be specified with the "tor_def" statement. The first "mask" specifies the gate conductor and the second "mask" specifies the drain/source conductor. Note that the optional ‘‘bulk’’ may not be connected to "@gnd".

```
<tor_def> ::= [ (transistors | fets) ':' [<tor>]* ]
<tor> ::= <name> ':' <mask_cond> ':' <mask> <mask> [<dscap>] [':' <bmask>]
<dscap> ::= '(' <mask_cond> ')'            # drain/source capacitance
<bmask> ::= <mask_sub> | <perc_subcont>    # bulk mask
```

2.4 TECHNOLOGY FILE: bjt_def

A bipolar junction transistor can be specified with the "bjt_def" statement. The first "layer" specifies the emitter conductor, the second "layer" the base conductor and the

third "layer" the collector conductor. The bjt type can be vertical (npn) or lateral (pnp). Note that the optional ‘‘bulk’’ may not be connected to "@gnd".

```
<bjt_def> ::= [ (bjtors | bjts) ':' [<bjt>]* ]
<bjt> ::= <name> ':' <mask_cond> ':' <b_type> ':' <b_pins> [':' <lay_sub>]
<b_pins> ::= <layer> <layer> <layer>
<b_type> ::= ver | npn | lat | pnp
```

2.5 TECHNOLOGY FILE: junction_def

This bipolar junction element is currently not supported by *space*.

```
<junction_def> ::= [ junctions ':' [<junction>]* ]
<junction> ::= <name> ':' <mask_cond> ':' <layer> [<layer>] ':' <j_values>
<j_values> ::= [<j_depth>] ':' [<j_pars>]
<j_depth>  ::= <value>
<j_pars>   ::= <value> [<value>]
```

2.6 TECHNOLOGY FILE: connect_def

A connect element connects different conductor regions of the same carrier-type with each other. A connect can be a surface or edge element. The connect can have optional a resistivity value. This value can only be used with certain *space* parameter settings.

```
<connect_def> ::= [ connects ':' [<connect>]* ]
<connect> ::= <name> ':' <mask_cond> ':' <layer> <layer> [':' <value>]
```

2.7 TECHNOLOGY FILE: contact_def

A contact element connects different conductors on top of each other. Thus, a contact can only be a surface element. The contact ‘‘sort’’ is seldom used (the default is "res"). Note that normally the "mask_cond" contains the name of a special via-mask. The contact resistivity value is default in ohms-square meters. Use the "c_resistance" unit statement to specify another unit.

```
<contact_def> ::= [ contacts [<cont_sort>] ':' [<contact>]* ]*
<contact> ::= <name> ':' <mask_cond> ':' <lay_sub> <lay_sub> ':' <value>
<cont_sort> ::= <name>
```

2.8 TECHNOLOGY FILE: cap_def

The capacitance elements are defined with the following statement. See the "Space User’s Manual" for a complete description.

```
<cap_def> ::= [ [junction] capacitances [<cap_sort>] ':' [<cap>]* ]*
<cap> ::= <name> ':' <mask_cond> ':' <lay_gsp> [<lay_gsp>] ':' <cap_val>
<cap_val> ::= <value> | (<value> <value>)*
```

For edge capacitances the cap_val can be specified as a function of the distance. In that case value pairs are used. The first value of the pair is the distance value.

2.9 TECHNOLOGY FILE: vdim_def

The vdimension definitions are used by 3D capacitance extraction.

```
<vdim_def> ::= [ vdimensions ':' [<vdim>]* ]
<vdim> ::= <name> ':' <mask_cond> ':' <mask> ':' <value> <value>
```

The "mask" must be specified as a conductor. The first "value" (> 0) specifies the distance between the substrate and the bottom of the conductor. The second "value" (≥ 0) specifies the thickness of the conductor. The unit of the values is default in meters. Use the "vdimension" unit statement to specify another unit.

2.10 TECHNOLOGY FILE: eshape_def

The following conductor shape definition is used by 3D capacitance extraction.

```
<eshape_def> ::= [ eshapes ':' [<eshape>]* ]
<eshape> ::= <name> ':' <mask_cond> ':' <layer> ':' <value> [<value>]
```

The first "value" specifies the extension of the bottom of the conductor. The second "value" specifies the extension of the top of the conductor. When not specified, the value is equal to the first value. The unit of the values is default in meters. Use the "shape" unit statement to specify another unit.

2.11 TECHNOLOGY FILE: cshape_def

The following conductor shape definition is used by 3D capacitance extraction.

```
<cshape_def> ::= [ cshapes ':' [<cshape>]* ]
<cshape> ::= <name> ':' <mask_cond> ':' <layer> ':' <cshape_values>
<cshape_values> ::= <value> <value> [<value> <value>]
```

The cross-over shape list specifies for different conductors the extensions for bottom and top in both directions. The first value pair is used for the extension to the left, and the second pair for the extension to the right. When only two values are specified (the bottom values), the top values are equal to the bottom values. The unit of the values is default in meters. Use the "shape" unit statement to specify another unit.

2.12 TECHNOLOGY FILE: dielec_def

The dielectric structure of the chip is specified with the following statement. It is used by 3D capacitance extraction. The "name" is the name of the dielectricum. The first "value" specifies the permittivity (relative epsilon) and the second "value" specifies the distance (≥ 0) between substrate and the bottom of the dielectricum. The first dielectricum starts always directly above the substrate and has a zero distance value. The distance values must always be specified in microns.

```
<dielec_def> ::= [ dielectrics ':' [<name> <value> <value>]* [<d_opts>]* ]
<d_opts> ::= grid_count ':' <value>
           | max_adjoint_binning_error     ':' <value>
```

```
           | max_determinant_binning_error ':' <value>
           | max_annealed_inverse_matrix_binning_error ':' <value>
           | max_preprocessed_annealing_matrices_binning_error ':' <value>
           | max_reduce_error    ':' <value>
           | max_annealing_error ':' <value>
           | num_annealing_iterations ':' <value>
           | r_switch  ':' <value>    | r_values  ':' [<value>]*
           | zp_values ':' [<value>]* | zq_values ':' [<value>]*
```

Note that the dielectric options are used by the ‘‘unigreen’’ method.
See for more details the "Space 3D Capacitance Extraction User's Manual".

2.13 TECHNOLOGY FILE: sublay_def

The substrate structure of the chip is specified with the following statement. It is used by 3D substrate resistance extraction. The "name" is the name of the substrate layer. The first "value" specifies the conductivity (in S/m) and the second "value" specifies the distance (≤ 0) between the top of the substrate and the top of the layer. The first layer starts always on the top of the substrate and has a zero distance value. The distance values must always be specified in microns.

```
<sublay_def> ::= [ sublayers ':' [<name> <value> <value>]* [<s_opts>]* ]
<s_opts> ::= grid_count ':' <value>
           | max_adjoint_binning_error     ':' <value>
           | max_determinant_binning_error ':' <value>
           | max_annealed_inverse_matrix_binning_error ':' <value>
           | max_preprocessed_annealing_matrices_binning_error ':' <value>
           | max_reduce_error    ':' <value>
           | max_annealing_error ':' <value>
           | neumann_simulation_ratio ':' <value>
           | num_annealing_iterations ':' <value>
           | r_switch  ':' <value>    | r_values  ':' [<value>]*
           | zp_values ':' [<value>]* | zq_values ':' [<value>]*
```

Note that the sublayer options are used by the ‘‘unigreen’’ method.
See for more details the "Space Substrate Resistance Extraction User's Manual".

2.14 TECHNOLOGY FILE: subcaplay_def

A second substrate structure of the chip may be specified with the following statement. It is used by 3D substrate capacitance extraction. The "name" is the name of the substrate capacitance layer. The first "value" specifies the permittivity (relative epsilon) and the second "value" specifies the distance (≤ 0) between the top of the substrate and the top of the layer. The first layer starts always on the top of the substrate and has a zero distance value. The distance values must always be specified in microns.

```
<subcaplay_def> ::= [ subcaplayers ':' [<name> <value> <value>]* ]
```

Note that the ‘‘unigreen’’ method is not implemented. Use parameter "add_sub_caps=2" with the *space3d* program to enable 3D substrate capacitance extraction.

2.15 TECHNOLOGY FILE: selfsub_def

The following list of definitions is used by simple substrate resistance extraction. The list specifies typical interpolation values for different substrate contact dimensions. The first "value" specifies the area (in square microns) and the second "value" specifies the perimeter (in microns). The third "value" specifies the resistance (in ohms) to the substrate node and the fourth "value" specifies a ratio factor (for which the conductance must be decreased because of direct coupling).

```
<selfsub_def> ::= [ selfsubres ':' [<selfsub_values>]* ]
<selfsub_values> ::= <value> <value> <value> <value>
```

2.16 TECHNOLOGY FILE: mutsub_def

The following list of definitions is used by simple substrate resistance extraction. The list specifies typical interpolation values for different substrate contact coupling situations. The first and second "value" specifies two substrate contact area's (in square microns). The third "value" specifies the minimum distance (in microns) between the two contacts, and the fourth "value" specifies the direct coupling resistance (in ohms), and the fifth "value" specifies a ratio factor (for which the conductance must be decreased).

```
<mutsub_def> ::= [ coupsubres ':' [<mutsub_values>]* ]
<mutsub_values> ::= <value> <value> <value> <value> <value>
```

2.17 TECHNOLOGY FILE: mask conditions

The ‘‘mask_cond’’ is a list of masks, which specifies the condition (also called minterms). An empty mask_cond is equal to 1 and means that the condition is always true. Note that a zero ('0') condition is no good condition, because it is always false.
A ‘‘mask_cond’’ contains masks, which is given by ‘‘layer’’ in the primary definition. The layer specifies more than just a standard or new mask name, it can also be an edge or other-edge mask. However, this is not always allowed in the above technology file definitions.

```
<mask_cond>       ::= [ <and_expression> ['|' <and_expression>]* ]
<and_expression> ::= ( <primary> | '!'<primary> )*
<primary>         ::= '(' <mask_cond> ')' | <layer> | '1' | '0'
```

2.18 TECHNOLOGY FILE: layers and masks

The standard mask names are specified in the technology "maskdata" file, but the user can add new mask names with ‘‘newmask_def’’. As can be seen below, a ‘‘layer’’ is more than just a standard mask (or new mask). It can also be an edge mask (specified with a leading '-' character) or an other-edge mask (specified with a leading '=' character). But these layers can only be used in the definitions of edge elements. Note that for the pins (or nodes) of a number of elements also "@gnd" or "@sub" may be used. The node "@gnd" denotes the ground plane and node "@sub" denotes the substrate node. Both nodes are maybe connected to each-other or possible connected to one of the power lines.

```
<mask>     ::= <name>
<mask_sub> ::= <mask> | @sub
<layer>    ::= ['-'|'='] <mask>
<lay_gsp>  ::= <layer> | @gnd | @sub | <perc_subcont>
<lay_sub>  ::= <layer> | @sub | <perc_subcont>
```

The ‘‘perc_subcont’’ can be used to specify the "mask_cond" for the substrate node. The "mask_cond" must be in agreement with the "mask_cond" of the element definition.

```
<perc_subcont> ::= '%' '(' <mask_cond> ')'
```

2.19 TECHNOLOGY FILE: names and values

An identifier "name" must start with a letter or underscore and furthermore only letters and digits and underscores may be used in names. An "name" cannot be a reserved word.

```
<name>     ::= <let>[<let>|<dig>]*
<hex_name> ::= '@'(<hex><hex><hex>)*
<dig> ::= '0'|'1'|...|'9'
<let> ::= 'a'|'b'|...|'z' | 'A'|'B'|...|'Z' | '_'
<hex> ::= 'a'|'b'|...|'f' | 'A'|'B'|...|'F' | <dig>
```

A "value" is a floating point number, which can also be negative. Thus, the floating point value is an integer string followed with (optional): (a) a decimal point and a number of digits, and/or (b) an E-notation, and/or (c) a modifier (unit letter).

```
<value>   ::= <integer>['.'(<dig>)*][<exp>][<mod>]
<integer> ::= ['-'](<dig>)*
<exp> ::= ('D'|'E'|'d'|'e')['-'|'+'](<dig>)*        # exponent
<mod> ::= 'f'|'p'|'n'|'u'|'m'|'k'|'M'|'G'           # modifier
```

The value modifiers have the following meaning:

```
'G'  giga  (1e+9)    'm'  milli (1e-3)
'M'  mega  (1e+6)    'u'  micro (1e-6)
'k'  kilo  (1e+3)    'n'  nano  (1e-9)
                     'p'  pico  (1e-12)
                     'f'  femto (1e-15)
```

2.20 TECHNOLOGY FILE: reserved words

The following lexical keywords can not be used for other names:

keywords	comment
bem_depth	set substrate bem depth
bjtors, bjts	begin of bjt definition
capacitance, a_capacitance, e_capacitance	used by unit definition
capacitances, colors	begin of ... definition
conductors, contacts, connects	begin of ... definition
coupsubres, selfsubres	begin of ... substrate definition
cshapes, eshapes	begin of ... definition
dielectrics	begin of dielectric definition
distance, layer_depth	used by unit definition
fets, transistors	begin of fet definition
grid_count	dielec_option / sublay_option
junction	capacitance type
junctions	begin of ... definition
keys, maxkeys	mask keylist definition
new	begin of new mask statement

neumann_simulation_ratio	sublay_option
num_annealing_iterations	dielec_option / sublay_option
max_determinant_binning_error	dielec_option / sublay_option
max_adjoint_binning_error	dielec_option / sublay_option
max_annealed_inverse_matrix_binning_error	dielec_option / sublay_option
max_preprocessed_*_matrices_binning_error	dielec_option / sublay_option
max_reduce_error	dielec_option / sublay_option
max_annealing_error	dielec_option / sublay_option
r_switch, r_values	dielec_option / sublay_option
resistance, c_resistance, s_resistance	used by unit definition
resize	used by unit definition and resize stat.
restype, subconn, viamask	wafer option
set, specific_resistance	used to set specific_resistance
shape, vdimension	used by unit definition
subcaplayers	begin of substrate cap definition
sublayers	begin of substrate res definition
unit, vdimensions, wafer	begin of ... definition
zp_values, zq_values	dielec_option / sublay_option
@sub	specifies the substrate mask
@gnd	specifies the ground mask

Note that "max_preprocessed_*_matrices_binning_error"
must be: "max_preprocessed_annealing_matrices_binning_error".

2.21 TECHNOLOGY FILE: characters with a special meaning

The "space", "tab" and "newline" characters are white space characters, which separate names and special characters, while scanning input.
The following lexical characters have a special meaning:

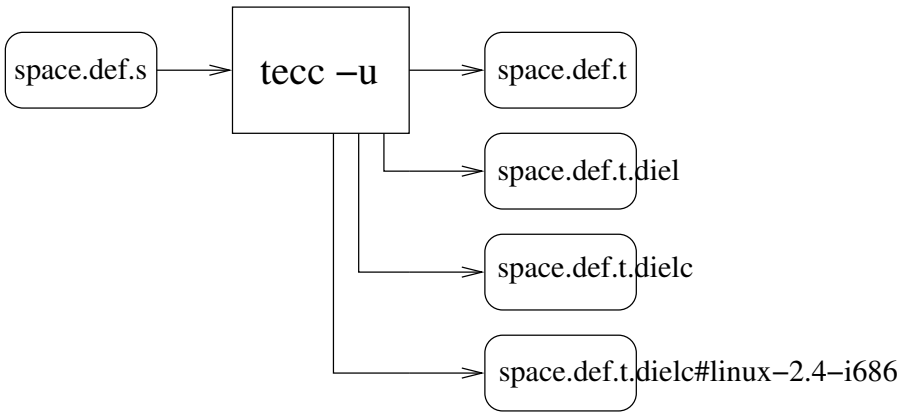
character	comment
#	start of comment
-	minus sign, edge mask indicator
=	equal sign, other-edge mask indicator
(	can be used around a mask condition
)	can be used around a mask condition
:	used as statement separator
!	NOT symbol, can be used in a mask condition
|	OR symbol, can be used in a mask condition
%	used before substrate mask condition
@	used before hexa-decimal colorname

All other special characters, not mentioned above, are treated to be illegal.

3. UNIGREEN TECHNOLOGY FILES

The technology compiler generated normally only a ".t" file out of a ".s" file (the element definition source file). Note that for the ‘‘unigreen’’ method other ".t" files can be generated (for more than 2 dielectrics or sublayers). These files can be used by the *space3d* program, when 3D capacitances or 3D substrate resistances are extracted.

The following picture gives the *tecc* flow for these files. See also the *tecc* manual for more details.



NOTE:

The new version of *tecc* now uses a process cache for these special unigreen files. First, *tecc* tries to write these files to the CACD system cache, directory:

```
$ICDPATH/share/lib/processcache
```

If this directory is not writable, then the user cache is used, directory:

```
$HOME/.cacd/cache/process
```

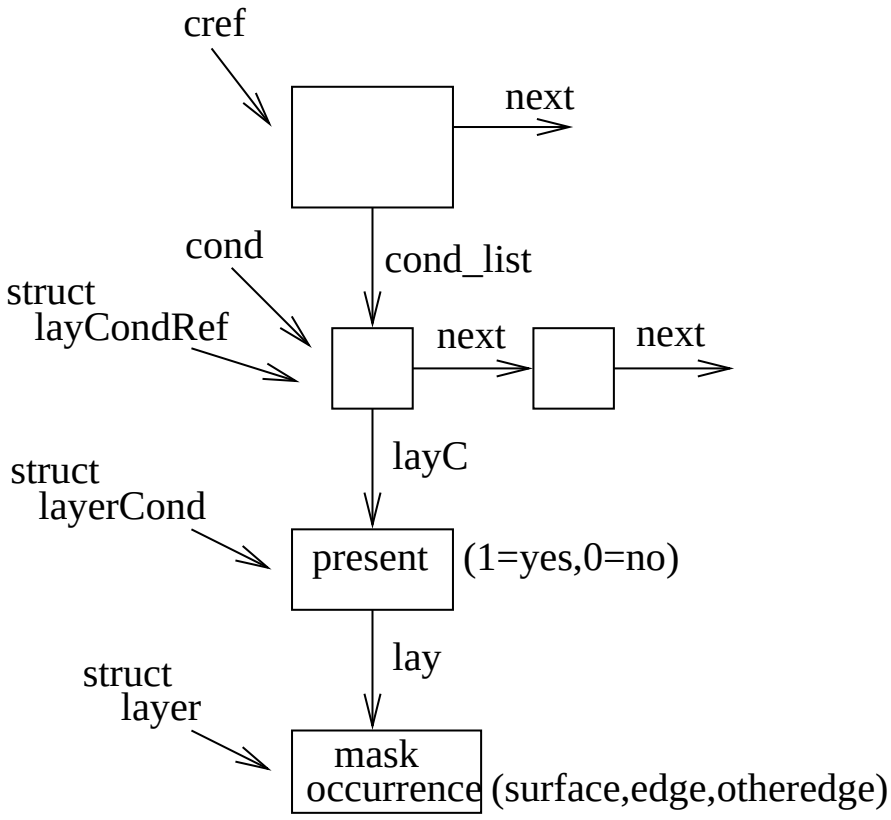
The file names of the ASCII versions are 33 characters long. The name is formed by a 32 byte checksum followed by a sequence number (1 - 9). The binary files have an extention (#0 or #1). The ’#1’ is used for little-endian architectures.

Note that the new *space3d* program tries first to read these unigreen files from the cache. If that is not successful, it tries to read the old names.

Note that *tecc* possible shall try to move the old unigreen files to a cache directory.

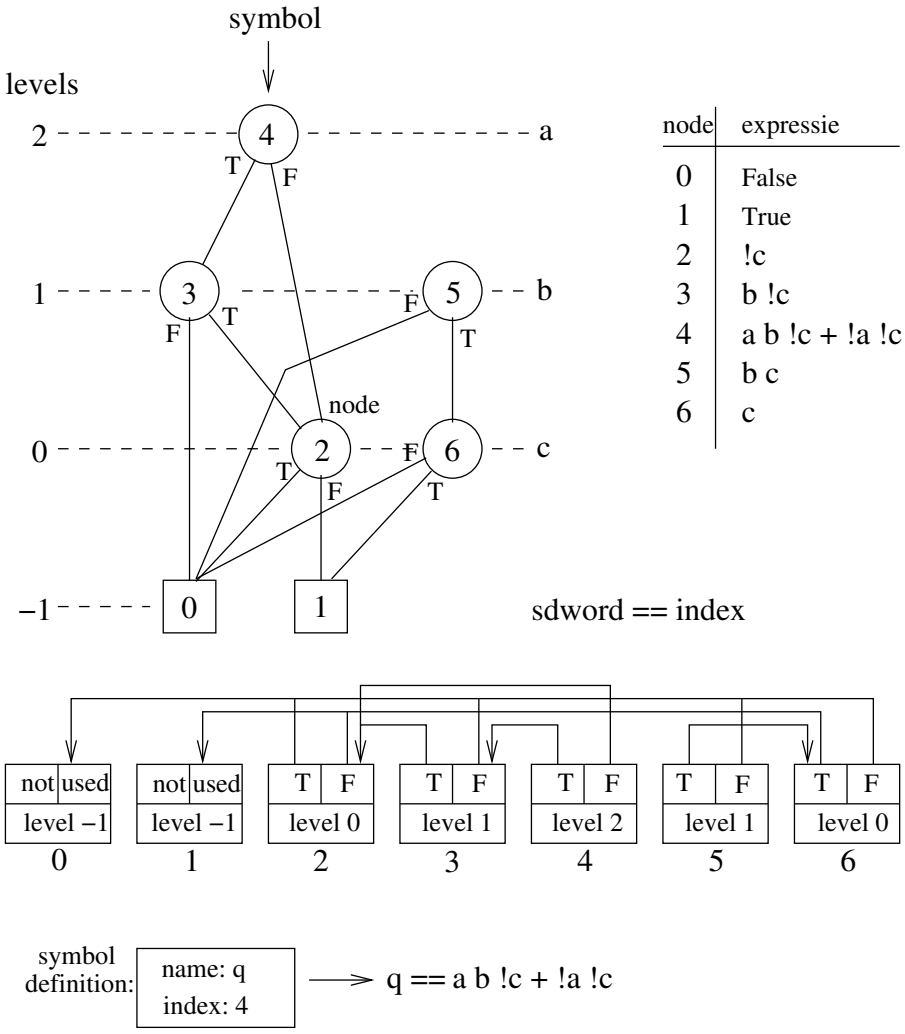
4. TECC DATA STRUCTURES

The following picture gives a overview of some of the data structures, which are used in the *tecc* program. The parser is building these data structures for different elements.
A list of these data structures describes the element ‘‘mask_cond’’ list.



5. ROBDD OF MINTERMS

The robdd code is implemented in *tecc* by Kees-Jan van der Kolk. The name robdd means "reduced ordered binary decision diagram". The C++ code can be found in directory "src/generic/libs/libmin". The following figure shows how it works.



node	expressie
0	False
1	True
2	!c
3	b !c
4	a b !c + !a !c
5	b c
6	c