Documentation of the Matlab functions for circuit design

February 26, 2004

Piet Wambacq, IMEC
February 26, 2004

Contents

1 Matlab files related to the circuit level

1.1 cirAllowedPrintFields

```
returns a list names of the fields of a circuit
    element of type ELTYPE, that can be dumped into/modified in a circuit
    simulation file.

    See also cirInit

    EXAMPLE:

    cirAllowedPrintFields(circuit, 'nmos')

    This returns the list {'w', 'lg', 'ad', 'as', 'pd', 'ps', ...
    'nFingers', 'lsos', 'lsod', 'lsogs', 'lsogd', 'mult'}
```

1.2 cirChangePrintNames

```
CIRCHANGEPRINTNAMES adapts the list of fields that will be written into a
    circuit simulation file.

    cirChangePrintNames(CIRCUIT, ELTYPE, NAMELIST, PRINTNAMELIST) modifies two
    cell arrays:
    1. the one that contains the parameters of a circuit element of a given
    type ELTYPE (must be a string) is set equal to the argument NAMELIST;
    2. the one that contains for each of these parameters the string that is
    written into the simulation file, is set equal to the argument
    PRINTNAMELIST. paramFields and the paramPrintNames, which
    are strings that are the fieldnames of a circuit element of type ELTYPE
    when this circuit element is defined with cirElementDef.

    This function changes the fields circuit.defaults.(eltype).fieldValues{5}
    and circuit.defaults.(eltype).fieldValues{6}. If this function is called,
    it must be done before any circuit element is defined.

    See also cirInit, cirElementDef, cirDefaultValues.

    EXAMPLE:

   cir = cirChangePrintNames(cir, 'nmos', {'w', 'lg', 'nFingers'}, ...
     {'Wnom', 'Lnom', 'FOLD'})

   With this example, we determine for circuit "cir" that in the circuit simulation
   will be written or modified, the width (w), the length (lg) and the number
   of fingers of an n-MOS will be written/modified. For example, in a circuit
   netlist (Spectre format) that contains n-MOS transistor Mn1, for which we
   have computed in Matlab w=10e-6, lg=0.2e-6 and nFingers = 5, the above
   example would yield a line in the circuit simulation file of the form

   Mn1 (drain gate source bulk) modelname Wnom=10e-6 Lnom=0.2e-6 FOLD=5
```

1.3 cirCheckEltype

```
CIRCHECKELTYPE check whether the specified string is a valid element type.

  cirCheckEltype(STRING) returns nothing if the string corresponds to a valid
  element type. Otherwise, an error is signaled. Valid element types are
  'nmos', 'pmos', 'res' (for resistor), 'cap' (for capacitor), 'v' (for
  voltage source), 'i' (for current source) and subcircuit ('subckt').
  These names are case sensitive.

  EXAMPLE :

      checkEltype('nmos')
         returns nothing, since the element type exists.
      checkEltype('NMOS')
          returns an error.

 See also cirInit
```

1.4 cirDefaultNames

```
CIRDEFAULTNAMES lists the fields names that are initialized when a circuit
element is defined.

    LIST = cirDefaultNames(CIRCUIT, ELTYPE) returns a cell array of
    strings that are the fieldnames of a circuit element of type ELTYPE
    when this circuit element is defined with cirElementDef.

    See also cirInit, cirElementDef, cirDefaultValues.
```

1.5 cirDefaultValues

```
CIRDEFAULTVALUES lists the default values of the fields that are initialized
when a circuit element is defined.

    LIST = cirDefaultValues(CIRCUIT, ELTYPE) returns a cell array with the
    default values to which the fields of a circuit element of type ELTYPE
    are initialized when this circuit element is defined with cirElementDef.

    See also cirInit, cirElementDef, cirDefaultNames.
```

1.6 cirElementCopy

```
CIRELEMENTCOPY copies a circuit element to another one.

   DESTINATION = cirElementCopy(SOURCE, DESTINATION) copies a circuit
   element SOURCE to another circuit element DESTINATION. By this copy
   operation, all fields except for the name field are copied from the
   SOURCE element to the DESTINATION element. It is required that the
   element types of SOURCE and DESTINATION correspond.


   EXAMPLES:

   m1b = cirElementCopy(m1a, m1b);
```

1.7 cirElementDef

```
CIRELEMENTDEF definition of a circuit element.

    CIRCUIT = cirElementDef(CIRCUIT, ELTYPE, ELEMENTNAME) adds a circuit
    element to the datastructure of the given CIRCUIT. This function changes
    the datastructure of the given circuit, which is returned. The circuit
    element needs to be specified by a name ELEMENTNAME, which should be a
    string and by its element type ELTYPE, which should also be a string.
    The circuit element is added as a field to the circuit, such as
    CIRCUIT.ELEMENTNAME. This is in turn a structure, which will have
    different fields depending on the element type. Two fields are already
    defined in this function, namely CIRCUIT.ELEMENTNAME.eltype and
    CIRCUIT.ELEMENTNAME.name, which are filled with the strings ELTYPE and
    ELEMENTNAME, respectively.
    Further, the field nElements of the given CIRCUIT is increased with one,
    the field CIRCUIT.(ELTYPELIST) is augmented with the ELEMENTNAME, and the
    field CIRCUIT.n(ELTYPE) is augmented with one. For example, for an n-MOS
    transistor Mn1, eltype is ’nmos’, and the field CIRCUIT.nmosList is
    extended with the name ’Mn1’, and the field CIRCUIT.nnmos is augmented
    with one.

    For the circuit element itself, the field "parent" is defined and it is
    equal to the name of the circuit.

    In addition to the fields eltype and name, some other fields, that are
    specific for each element type (see cirDefaultNames(CIRCUIT, ELTYPE)),
    are already defined and initialized with default values.
    These default values can be overridden during initialization
    by specifying extra arguments to cirElementDef in addition to the three
    arguments CIRCUIT, ELTYPE and ELEMENTNAME. If no extra arguments are
    specified, then default values are used to initialize the fields of
    cirDefaultNames(CIRCUIT, ELTYPE). Else, when default values need to be
    overridden, the syntax is as follows:
    CIRCUIT = cirElementDef(CIRCUIT, ELTYPE, ELEMENTNAME, ’fieldname’, value,
    ’fieldname’, value, ...)
    This means that the fieldname (string) has to be specified as an argument,
    immediately followed by its value as the next argument.

    See also cirInit, cirDefaultNames, cirDefaultValues.

    EXAMPLES:

    cir = cirElementDef(cir, ’nmos’, ’m1’)

         This example defines n-MOS transistor m1 and takes default values for
         the fields lg, w, table, paramFields, paramPrintNames, lsos,
         lsod, mult, lsogs, lsogd, mult.

    cir = cirElementDef(cir, ’nmos’, ’m1’, ’lg’, 2e-6)

        This example defines n-MOS transistor m2. The channel length lg is
        initialized to 2 micrometer, while the other fields w, table,
        paramFields, paramPrintNames, lsos, lsod, mult, lsogs, lsogd, mult
```

are initialized to their default value.

```
cir = cirElementDef(cir, ’nmos’, ’m3’, ’paramFields’, {nFingers})
```

This example defines n-MOS transistor m3. The field "paramField" is initialized with {nFingers}. This means that only the value field "nFingers" will be printed to the circuit simulation file. The other fields are initialized with their default value.

```
cir = cirElementDef(cir, ’nmos’, m4, ’paramFields’, {gm, lg, w})
```

This will give an error, since "gm" is not a parameter that can be specified as one that will be printed to the circuit simulation file.

1.8 cirElementOfType

```
CIRELEMENTOFTYPE returns a list of elements of a given type.

    ELEMENTLIST = cirElementOfType(CIRCUIT, ELTYPE) returns a cell array of
    circuit elements of a given type ELTYPE (specified as a string), that
    have been defined in the given CIRCUIT.

    See also cirInit, cirDefaultNames, cirDefaultValues.

    EXAMPLES:

    nmosList = cirElementOfType(cir, 'nmos')
```

1.9 cirInit

```
CIRINIT initialization of a circuit

 circuit=cirInit(NAME, TITLE, PRINTPARAMTUPLE (optional), SIMULFILE
 (optional), NMOSTABLE (optional), PMOSTABLE (optional), PARAMSIMULSKELFILE
 (optional))
   initializes a datastructure for a circuit that is returned. This
   datastructure is a structure with the following fields:

     1. name: a string, which is specified as the argument NAME of this
        initialization function.

     2. title: a title string, which is specified as the argument TITLE
        of this initialization function.

     3. simulFile: a string that represents the name of the file to which the
        values will be written for the different circuit parameters that are
        determined with Matlab. This string is specified as the argument
        SIMULFILE to this initialization function.

     4. simulSkeletonFile: a string representing the name of the file that
        contains the part of a circuit simulation file that does not contain
        any parameters that are determined in Matlab. This field is only used
        for circuit simulations in SmartSpice. Therefore, it is an optional
        argument SIMULSKELFILE, that does not need to be specified when other
        simulators are used.

     5. defaults: a structure that contains default values for the different
     circuit elements that are supported. These default values are used when a
     circuit element is initialized. For a nMOS and pMOS transistor,
     default values are used for
       - channel length (field "lg")
       - channel width (field "w")
       - table of intrinsic operating point parameters (field "table")
       - length of drain/source diffusion areas not between two
         poly stripes (fields "lsod" and "lsos", respectively)
       - length of drain/source diffusion areas between two poly stripes
         (fields "lsogd" and "lsogs", respectively)
       - the amount of fingers (field "nFingers")
       - the multiplicity (field "mult")
       - the parameters that will be written
         to a circuit simulation file (field "paramFields"). This is a cell
         array of strings, each of them being a field of a MOS transistor
         structure (e.g. "lg"). The value of these fields will be determined
         in the Matlab design plan.
       - the printnames that will be used for the parameters in the field
         paramFields of the transistor (field "paramPrintNames"). This is also
         a cell array of strings. This cell array has the same length as the
         cell array "paramFields". The printnames are used for the names of
         the parameters that are written into the circuit simulation
         file. E.g. the printname of the length of a MOS transistor could be
         ’l’. In that case, a parameter would be made of the form
         ’<mos><glue string>l’, e.g. m1_l
```

For a voltage source, current source, capacitor, inductor and resistor, default values are used for
  - value (field "val")
  - field "paramFields" (similar meaning as for a MOS transistor)
  - field "paramPrintNames" (similar meaning as for a MOS transistor)

6. allowedPrintFields: an array of strings (cell array) that contains the names of the fields that can be passed to/changed in a circuit simulation file. For example, for a MOS transistor this list is
```
{'w', 'lg', 'ad', 'as', 'pd', 'ps', 'nFingers', 'lsos', 'lsod', ...
 'lsogs', 'lsogd', 'mult'}.
```
For a resistor, capacitor, inductor, voltage source and current source, this list contains one element, namely 'val'.

7. nElements: number of circuit elements that already have been defined.

8. fields that each represent a circuit element that belongs to the circuit. At initialization there is no such field. During the Matlab program these fields are filled. E.g., when an n-MOS transistor m1 is defined (see function cirElementDef) then there will be a field "m1" in the circuit.
   The following types of elements are supported: nmos, pmos, resistor, capacitor, voltage source, current source (see function cirCheckEltype). Note that not every circuit element that appears in the netlist of the circuit, needs to be defined in Matlab. Only those elements need to be defined for which some parameters are computed in Matlab. For example, if the power supply voltage is fixed, then the corresponding voltage source does not need to be added to the datastructure.

9. fields that represent for each circuit element type a list of the elements of that type and the length of that list:
   - nmosList and nnmos
   - pmosList and npmos
   - resList and nres, capList and ncap, indList and nind, vList and nv, iList and ni, subcktList and nsubckt

The argument PRINTPARAMTUPLE to cirInit is a cell array that is constructed as follows. It consists of groups of three elements. The first element of each group is a string that corresponds to a circuit element type. The second element contains a list of strings that correspond to the parameters that can be printed to or modified in a circuit simulation file. The third element is a list of the same length as the second element. Every element in this list is a string that is used in the simulation file for the value of the corresponding parameter of the first element. For example, for a resistor, the first element is 'res'. If now the second element is {'val'} and the third element is {'r'}, then the field "val" of a resistor is represented by "r" (this corresponds to the Spectre syntax). The possible strings in the second elements of each trio of this list should belong to the fields circuit.allowedPrintFields.res, circuit.allowedPrintFields.cap, ...

1.10 cirSimulFile

```
CIRSIMULFILE retrieves the circuit simulation file

   FILENAME = cirSimulFile(CIRCUIT) returns a string that contains for the
   given CIRCUIT the name of the file that will be used to verify the design
   with Matlab of this CIRCUIT.
```

1.11 cirSimulSkelFile

```
CIRSIMULSKELFILE retrieves the circuit simulation file

   FILENAME = cirSimulSkelFile(CIRCUIT) returns a string that contains for the
   given CIRCUIT the name of the file that contains the part of the circuit
   simulation input file that is fixed (i.e. contains no elements that are
   determined with Matlab).
```

1.12 cirSubcktInit

```
CIRSUBCKTINIT initialization of a subcircuit of a given parent circuit.
   This parent circuit is returned.

 PARENT = cirSubcktInit(CIRCUITNAME, TITLE, PARENT)
 initializes a datastructure for a circuit that is returned. This
   datastructure is a structure with the following fields:

     1. name: a string (argument CIRCUITNAME)

     2. title: a title string (argument TITLE), which can be a short
     description of the circuit

     3. simulFile: a string that represents the name of the file to which the
        values will be written for the different circuit parameters that are
        determined with Matlab. This is inherited from the parent circuit.

     4. simulSkeletonFile: also inherited from the parent circuit.
```

1.13 cirSubcktReadin

```
CIRSUBCKTREADIN construct a subcircuit from  an existing circuit
 datastructure and attach it to the datastructure of a given parent circuit.
```

1.14 cirWriteParam

```
CIRWRITEPARAM write a parameter statement for the given simulator

    cirWriteParam(FID, SIMULATOR, PARAM, VALUE) writes a parameter
    statement with the appropriate syntax for the given SIMULATOR for the
    given PARAMETER (should be a string). The VALUE of that parameter
    should also be given as an argument. The parameter statement is
    written to the given file identifier FID

    See also fopen, cirWriteParams
```

1.15 cirWriteParams

```
CIRWRITEPARAMS writes parameters that have been determined in Matlab to a
circuit simulation file for a given SIMULATOR.

   Supported simulators are smartspice, hspice, spectre


   For smartspice we put everything into one file in order to avoid .include
   statements, which seem to be buggy in smartspice. Therefore, we dump the
   parameters in a temporary file, and in the end we combine this file and
   the skeleton file into the simulation file.

   For HSPICE, the simulation file is just a set of lines with .param
   statements. The skeleton file is not used here.

   For spectre, we replace in a circuit netlist the values of the circuit
   elements in the top circuit and in the subcircuits. The skeleton file is
   not used neither here.

  See also cirWriteParamsSpice and cirWriteParamsSpectre
```

1.16 cirWriteParamsSpectre

```
CIRWRITEPARAMS writes parameters that have been determined in Matlab to a
circuit simulation file for spectre. We replace
   in a circuit netlist the values of the circuit elements in the top circuit
   and in the subcircuits.
```

1.17 cirWriteParamsSpice

```
CIRWRITEPARAMSSPICE writes parameters that have been determined in Matlab to
 a circuit simulation file for Spice. Supported Spice versions are
  smartspice and hspice.


   For smartspice we put everything into one file in order to avoid .include
   statements, which seem to be buggy in smartspice. Therefore, we dump the
   parameters in a temporary file, and in the end we combine this file and
   the skeleton file into the simulation file.
   For HSPICE, the simulation file is just a set of lines with .param
   statements. The skeleton file is not used here.
```

2 Matlab files related to the MOS transistor

2.1 mosAcmPerimeter

```
MOSACMPERIMETER computation of the perimeter of a source or drain region

   mosAcmPerimeter(ACM, LSWS, LSWG) returns the perimeter of a diffusion region
   (= source or drain region). This perimeter is computed according to the area
   calculation method ACM with the given perimeter of the diffsuion region under
   the gate (argument LSWG) and not under the gate (argument LSWS).
```

2.2 mosAreaPerimeter

```
MOSAREAPERIMETER computation of areas and perimeters of source and drain
 regions of a MOS transistor.

   [AD, LSD, LGD, AS, LSS, LGS] = mosAreaPerimeter(Wnom, LSOs, LSOd, ...
    LSOgs, LSOgd, FOLD, MULT) returns the area of source and drain regions
    (AD and AS), the length of the sidewall of the source or drain area which
    is not under the gate (LSD and LSS), length of the sidewall of the source
    or drain area which is under the gate (LGD and LGS).
    These quantities are computed as a function of the nominal transistor
    width WNOM, the length of drain/source diffusion areas not between two
    poly stripes (LSOd and LSOs, respectively), the length of drain/source
    diffusion areas between two poly stripes (LSOgd and LSOgs, respectively),
    the amount of fingers FOLD and the multiplicity MULT.

    EXAMPLES :

     [AD, LSD, LGD, AS, LSS, LGS] = mosAreaPerimeter(Wnom, LSOs, LSOd, ...
    LSOgs, LSOgd, 1, 1)

    [AD, LSD, LGD, AS, LSS, LGS] = mosAreaPerimeter(20, 1, 1, 1, 1, 1, 1)
    returns
    AD = 20, LSD = 22, LGD = 20, AS = 20, LSS = 22, LGS = 20

    [AD, LSD, LGD, AS, LSS, LGS] = mosAreaPerimeter(20, 1, 1, 1, 1, 4, 1)
    returns
    AD = 10, LSD = 4, LGD = 20, AS = 15, LSS = 16, LGS = 20
```

2.3 mosCheckSaturation

```
CHECKMOSSATURATION check whether a transistor operates in the saturation
    region.

    checkMosSaturation(MOS) gives a warning when transistor MOS is not
    saturated. This function compares MOS.vds to MOS.vdsat if they exist.
    When one of these fields does not exist, an error is signaled.
```

2.4 mosCheckWidth

```
MOSCHECKWIDTH comparison of a transistor width to the critical width

   result = mosCheckWidth(WIDTH, TABLE) returns 0 and gives a warning when
   the specified WIDTH is below the critical width of the given TABLE.
   This critical width can be obtained with tableWcrit(TABLE)

   See also tableWcrit
```

2.5 mosExtValue

```
MOSEXTVALUE computation of parameters related to the extrinsic
part of a MOS transistor.

    value = mosExtValue(NAME, MOS) returns the value of a parameter related
    to the extrinsic part of a MOS transistor. The NAME of the parameter
    should be specified as a string. Possible names are
    cdbE, csbE, as, ad, ps, pd, lsd, lgd, lss, lgs

    EXAMPLE :

       m1.cdbE = mosExtValue('cdbE', m1);
```

2.6 mosIntValue

```
MOSINTVALUE gets the value of a intrinsic MOS parameter

    PARAMVALUE = mosIntValue(ELECPARAM, MOS) returns the
    value of a MOS operating point parameter ELECPARAM (specified as a string) for
    transistor MOS. The value is computed
    in S.I. units.
    ELECPARAM must be included in the table of operating point parameter values
    of a transistor. Possible names for ELECPARAM can be found by running
    tableDisplay(mosTable(mos)).

   See also tableDisplay, tableValueWref, mosWidth, tableWref, tableWcrit


    EXAMPLE :

      ids = mosIntValue('ids', m1)
```

2.7 mosIntValueWref

```
MOSINTVALUEWREF retrieving value of an intrinsic MOS parameter for
a given MOS transistor, assuming its width equals the reference width.

   RESULT = mosIntValueWref(PARAM, MOS) returns the value of the
   MOS operating point parameter PARAM (specified as a string) for a given MOS
   transistor.
   PARAM must be included in the TABLE of intrinisc values of a transistor
   of the same type of the given transistor. Possible names for PARAM can be
   found by running tableDisplay(TABLE).

   EXAMPLE :

      id = mosIntValueWref('ids', m1)

   See also tableDisplay, mosIntValue, mosWidth, tableWref, tableValueWref
```

2.8 mosJuncap

```
MOSJUNCAP computation of junction capacitance of a source or drain region
of a MOS transistor.

    cJunction = mosJuncap(TABLE, VDIFFBULK, AREA, LSWS, LSWG) returns
    the value of the junction capacitance of a source or drain region for a
    given type of a MOS transistor, specified by the given TABLE. For the
    junction one must specify the AREA, the length of the sidewall under the
    gate (LSWG) and not under the gate (LSWS) and the voltage over the
    junction VDIFFBULK. The latter is the voltage difference between the
    source or drain region and the bulk (should be positive for an n-MOS and
    negative for a p-MOS)

    EXAMPLE :

       for a MOS transistor with one finger, a width of 2 micrometer, a
       length of 1 micrometer for the diffusion areas, and 1 Volt over the
       junction between the diffusion area of the bulk and a table name N, the
       function call is
       cdb = mosJuncap(N, 1, 2e-12, 4e-6, 2e-6);
```

2.9 mosName

```
MOSNAME returns the name of a given MOS transistor as a string


    NAME = mosName(MOS) returns a string that contains the name of the
    given MOS transistor.

    EXAMPLE :

      nameM1 = mosName(m1);
```

2.10 mosOpValues

```
MOSOPVALUES computation of all operating parameters of a MOS transistor

   MOS = mosOpValues(MOS) computes the value of intrinsic and extrinsic
   operating point parameters (in S.I. units) of a given transistor MOS. The
   datastructure of the transistor which is specified as an argument is also
   returned. However, after this function has returned, all fields of the
   transistor related to intrinsic and extrinsic operating point parameters
   have been added to the datastructure of the transistor. Also, the
   values of vgb, vgd and vdb are computed (or updated) from vgs, vds and vsb.

   EXAMPLE :

       m1 = mosOpValues(m1)
```

2.11 mosPrintSizesAndOpInfo

```
MOSPRINTSIZESANDOPINFO print sizes and operating point information of MOS
transistors in a circuit and its subcircuits.

  mosPrintSizesAndOpInfo(FID, CIRCUIT) prints the sizes and operating point
  information of the transistors in the given CIRCUIT and its subcircuits to
  the given file identifier FID. Two file identifiers are automatically
  available and need not be opened.  They are FID=1 (standard output) and
  FID=2 (standard error).

 See also fopen.
```

2.12 mosTable

```
MOSTABLE returns the table of operating point parameters from which the
intrinsic operating point of a MOS transistor are derived.

    TABLE = mosTable(MOS) returns the table (this is a structure) from
    which the intrinsic operating point parameters of the given MOS
    transistor are derived.
```

2.13 mosVsbBody

```
MOSVSBBODY computation of the source voltage of a MOS transistor with body
effect

    VSB = mosVsbBody(MOS, VGB, VDB, VOV, ESTIMATE) determines the
    source-bulk voltage VSB of a MOS transistor by using the given TABLE.
    A specified value for the overdrive voltage VOV is given as an argument,
    as well as a value for the gate-bulk voltage VGB and drain-bulk voltage VDB.

    The voltage VSB is determined by fixed-point iteration. A start value
    ESTIMATE needs to be given for this iteration.

   EXAMPLE:

      m1.vsb = mosVsbBody(m1, vdd/2, vdd, 0.2, vdd/2 - 0.2);
```

2.14 mosWidth

```
MOSWIDTH returns MOS transistor width for a given operating point parameter

   W  = mosWidth(ELECPARAM, VALUE, MOS) returns the width W of a MOS
   transistor. This returned width W is such that a wanted value VALUE for
   the MOS operating point parameter ELECPARAM (specified as a string) is
   realized. ELECPARAM must be included in the TABLE of a transistor of the same
   type. Possible names for PARAM can be found by running tableDisplay on
   mosTable(mos). If the returned width is below the critical width a warning
   is given. If the computed width is below the minimum width for the given
   technology, then an error is given.

   See also mosTable, tableDisplay, tableWcrit, tableWmin, tableWref

   EXAMPLE :

     M1.w = mosWidth('ids', 0.001, M1);

     In this example, the width of M1 is computed as M1.w, based on a
     specification of 1mA for the drain current of M1. It is assumed that the
     terminal voltages and the length of M1 are already determined.
```

2.15 mosWriteParams

```
MOSWRITEPARAMS writes parameters that have been determined in Matlab to a
the fields of a structure parStruct
```

3 Matlab files related to technology characterization

3.1 techCapDigital

```
TECHCAPS plots intrinsic parasitic (trans)capacitance for |vgs| = VDD.

    techCapDigital(TABLE, CAPPARAM, LG, VSB, FIGNUMBER) plots the intrinsic
    parasitic capacitance CAPPARAM as a function of VDS.
    The plot is generated for a MOS transistor characterized by the
    given TABLE of intrinsic operating point parameters.
    The length of the channel and vsb need to be specified as an argument
    (LG and VSB, respectively). These values should belong to the arrays
    tableInArray('lg', TABLE) and tableInArray('vsb', TABLE), respectively.

    See also tableInArray.
```

3.2 techCaps

```
TECHCAPS plots intrinsic parasitic (trans)capacitances.

    techCaps(TABLE, CAPPARAM, LG, VSB, NVDSVALUES, FIGNUMBER) plots the intrinsic
    parasitic capacitance CAPPARAM as a function of |VGS| with |VDS| as a
    parameter. The plot is generated for a MOS transistor characterized by the
    given TABLE of intrinsic operating point parameters. To limit the number
    of parameters, the number of vds values can be specified (argument
    NVDSVALUES).
    The length of the channel and vsb need to be specified as an argument
    (LG and VSB, respectively). These values should belong to the arrays
    tableInArray('lg', TABLE) and tableInArray('vsb', TABLE), respectively.

    See also tableInArray.
```

3.3 techGmId

```
TECHGMID plots gm/Ids and gm/Ids*Vov as a function of Vov.

    techGmId(TABLE, VSBARRAY, FIGNUMBER) plots gm/Ids and gm/Ids*Vov for
    different values of vsb, which are specified in the array VSBARRAY.
    The number of plots that are generated is equal to two times the length of
    the array VSBARRAY. The first plot is generated in the figure with number
    FIGNUMBER.
```

3.4 techIds

```
TECHIDS plots IDS and its derivatives gm, gmbs, gds, and the ratios gm/gds
and gm/gmbs as a function of VDS for different values of VGS.

   techIds(TABLE, LG, VSBINDEX, NVGSVALUES, FIGNUMBER) will generate five
   plots in the figures with number FIGNUMBER, ..., FIGNUMBER + 4. The
   following parameters are plotted: |ids|, gm, gds, gmbs, gm/gds and gm/gmbs.
   The x-axis of these plots is |VDS|. The above parameters are generated with
   VGS as a parameter. To limit the number of parameters, the number of vgs
   values can be specified (argument NVGSVALUES).
   The length of the channel and vsb also need to be specified as an argument
   (LG and VSB, respectively). These values should
   belong to the arrays tableInArray('lg', TABLE) and tableInArray('vsb',
   TABLE), respectively.

   See also tableInArray
```

3.5 techPlots

```
TECHPLOTS generates many plots to characterize an n-MOS or p-MOS in a CMOS
    technology, specified by a given TABLE.

    EXAMPLE:

    techplots(P)
```

3.6 techVdsat

```
TECHVDSAT plots |VDSAT| as a function of |VGS| - |VT| for different
channel lengths

    techVdsat(TABLE, VSBINDEX, FIGNUMBER) generates a plot of |VDSAT| in the
    figure with number FIGNUMBER. |VDSAT| is plotted as a function of
    |VGS| - |VT| with the channel length as a parameter for the given
    table TABLE of intrinsic operating point parameters of a MOS transistor.
    Further, vsb needs to be specified. The value of this should occur in
    tableInArray('vsb', TABLE).

    See also tableInArray

    EXAMPLE :

    techVdsat(N, 0, 1)

    This will generate a plot of VDSAT for an n-MOS transistor characterized
    by the table with the name N and with vsb = 0V. The plot is generated in
    figure number 1.
```

3.7 techVth

```
TECHVTH plots |VT| as a function of |VDS| for different
channel lengths

    techVth(TABLE, VSB, FIGNUMBER) generates a plot of |VT| in the
    figure with number FIGNUMBER. |VT| is plotted as a function of |VDS|
    with the channel length as a parameter for the given table TABLE of
    intrinsic operating point parameters of a MOS transistor. Further, vsb
    should be specified with a value that occurs in tableInArray('vsb', TABLE).

    EXAMPLE :

    techVth(N, 0, 1)

    This will generate a plot of VT for an n-MOS transistor characterized
    by the table with the name N and with vsb = 0V. The plot is generated in
    figure number 1.
```

4 Matlab files related to tables of a MOS transistor

4.1 tableDisplay

```
 TABLEDISPLAY view the parameters that are present in a table for a MOS transistor

    tableDisplay(TABLE) displays information contained in the table that has
    been constructed for a given transistor type.
    The following information is displayed:
      1) DIMENSIONS: minimum gate length and width, width for which the table
         has been constructed (so-called reference width), width below which
         narrow-channel effects become visible (so-called critical width).
      2) A list of the input variables and their ranges
      3) A list of operating point parameters that are stored in the table
          (gm, ids, cgs, vdsat), .... Some parameters scale proportionally to
          the transistor width (at least above the critical width), other ones
          (e.g. vdsat, vth) do not.

    EXAMPLE :

      tableDisplay(N)
```

4.2 tableInArray

```
TABLEINARRAY getting the array of values of an input variable of a MOS table.

    VALUE = tableInArray(INPUTVAR, TABLE) returns the array
    of values of the input variable INPUTVAR for a given TABLE of a MOS
    transistor.

    EXAMPLE :

      vgsArray = tableInArray('vgs', N);

    See also tableInFinal, tableInInit, tableInValue, tableInStep,
    tableInLength.
```

4.3 tableInCheckRange

```
TABLEINCHECKRANGE check whether an input parameter is inside the validity range

   tableInCheckRange(INPUTVAR, VALUE, TABLE) tests whether the VALUE of a given
   input parameter INPUTVAR (specified as a string) is inside the range of
   validity, i.e. the range that is used to construct the given TABLE for a
   MOS transistor.
   The function signals an error if the VALUE is not inside the range of
   validity.
```

4.4 tableInFinal

```
TABLEINFINAL getting the final value of an input variable of a MOS table.

    VALUE = tableInFinal(INPUTVAR, TABLE) returns the last element of the array
    of values of the input variable INPUTVAR for a given TABLE of a MOS
    transistor.

    EXAMPLE :

      vgsFinal = tableInFinal('vgs', N);

    See also tableInInit, tableInArray, tableInValue, tableInStep,
    tableInLength.
```

4.5 tableInInit

```
TABLEININIT getting the initial value of an input variable of a MOS table.

    VALUE = tableInInit(INPUTVAR, TABLE) returns the first element of the array
    of values of the input variable INPUTVAR for a given TABLE of a MOS
    transistor.

    EXAMPLE :

      vgsInit = tableInInit('vgs', N);

    See also tableInFinal, tableInArray, tableInValue, tableInStep,
    tableInLength.
```

4.6 tableInLength

```
TABLEINLENGTH getting the number of values of an input variable of a MOS table.

    N = tableInLength(INPUTVAR, TABLE) returns the number
    of values of the input variable INPUTVAR for a given TABLE of a MOS
    transistor.

    EXAMPLE :

      nVgs = tableInLength('vgs', N);

    See also tableInInit, tableInFinal, tableInArray, tableInValue, tableInStep,
    tableInLength.
```

4.7 tableInNames

```
TABLEINNAMES input variables for the table of a MOS transistor

   CELLARRAY = tableInNames(TABLE) returns an array of strings that
   are the names of the input variables of a given TABLE of intrinsic
   operating point parameters of a MOS
   transistor. The order of these strings also corresponds to the order of the
   indices in the table values. E.g. when ’vgs’ is the second element in the
   resulting CELLARRAY, then this also means that the second index of the
   table corresponds to vgs.

   EXAMPLE:

     stringArray = tableInNames(N);
```

4.8 tableInStep

```
TABLEINSTEP step between two adjacent values of an input variable of a MOS table.

    VALUE = tableInStep(INPUTVAR, TABLE) returns the value of the step
    of the input variable INPUTVAR for a given TABLE of a MOS
    transistor. This step can be positive or negative. When the step is not
    constant, then the function returns NaN.

    EXAMPLES :

      vgsStep = tableInStep('vgs', N);
      lgStep = tableInStep('lg', N);

       This second example will return NaN.

    See also tableInInit, tableInFinal, tableInArray, tableInValue, tableInStep,
    tableInLength.
```

4.9 tableInterpol

```
TABLEVALUEWREF retrieving the value of an intrinsic MOS parameter for
the reference width, directly from a given table.

   RESULT = mosIntValueWref(PARAM, TABLE, LENGTH, VGS, VDS, VSB) returns
   the value of the MOS operating point parameter PARAM
   (specified as a string) for a given TABLE.
   PARAM must be exist as a field of the given TABLE.
   Possible names for PARAM can be found by running tableDisplay(TABLE).

   EXAMPLE :

      id = tableValueWref('ids', N, 0.18e-6, 0.7, vdd/2, 0)

   See also tableDisplay, mosIntValue, mosWidth, tableWref
```

4.10 tableInValue

```
TABLEINVALUE gets a particular value from the array of values
of an input variable of a MOS table of intrinsic operating point parameters.

    VALUE = tableInValue(INPUTVAR, INDEX, TABLE) returns the element with
    index INDEX in the array of values of the input variable INPUTVAR in a
    given TABLE of a MOS transistor.

    EXAMPLE :

      vgsInter = tableInValue('vgs', 5, N);

    See also tableInFinal, tableInInit, tableInArray, tableInStep,
    tableInLength.
```

4.11 tableLmin

```
 TABLELMIN minimum channel length of a MOS transistor

    LMIN = tableLmin(TABLE) returns the value of the minimum
    channel length that is allowed for the transistor type that is specified by
    the given TABLE of intrinsic operating point parameters of a MOS transistor.

    EXAMPLE :

      lmin = tableLmin(N);

    See also tableWmin, tableWcrit
```

4.12 tableMaxVdd

```
TABLEMAXVDD retrieving the maximally allowed VDD for a MOS transistor

   VALUE = tableMaxVdd(TABLE) returns the value of the maximally allowed VDD for
   a MOS transistor that is specified by a given TABLE of intrinsic
   operating point parameters of a MOS transistor.

   EXAMPLE :

     vdd = tableMaxVdd(N);
```

4.13 tableModelName

```
TABLEMODELNAME model name of MOS transistor with which the table values
 are computed.

   STRING = tableModelName(TABLE) returns the name of the MOS model with which
   the values in the given TABLE have been computed.

   EXAMPLE :

     modelname = tableModelName(N);
```

4.14 tableModelParam

```
TABLEMODELPARAM getting the value of a model parameter from a MOS table

   VALUE = tableModelParam(PARAMETER, TABLE) retrieves from the data of a given
   MOS table TABLE the numerical value of a model parameter that is specified
   with the string PARAMETER. No unit conversion is performed, which means
   that modelparameters that are not specified in S.I. units are returned in
   non-S.I. units.

   EXAMPLE :

      M1.cjsw = tableModelParam('cjsw', N);

    See also tableModelParamNames, tableModelName
```

4.15 tableModelParamNames

```
TABLEMODELPARAMNAMES list of all names of model parameters

   RESULT = tableModelParamNames(TABLE) returns a cell array of strings. This cell
   array contains the names of all model parameters that have been defined in
   the given TABLE of a MOS transistor.

   EXAMPLE :

    modelParamNames = tableModelParamNames(N);

   See also tableModelParam, tableModelName
```

4.16 tablePre

```
preprocessing of the tables REG018N and REG018P of UMC 0.18um CMOS
```

4.17 tablePreprocess

```
modification of MosTable.Table:
```

4.18 tableType

```
TABLETYPE returns the type of a transistor (n or p)

   string = tableType(TABLE) returns a string 'n' or 'p' depending on
   whether the TABLE under consideration refers to an n-MOS or p-MOS
   transistor.
```

4.19 tableValueWref

```
TABLEVALUEWREF retrieving the value of an intrinsic MOS parameter for
the reference width, directly from a given table.

   RESULT = tableValueWref(PARAM, TABLE, LENGTH, VGS, VDS, VSB) returns
   the value of the MOS operating point parameter PARAM
   (specified as a string) for a given TABLE.
   PARAM must be exist as a field of the given TABLE.
   Possible names for PARAM can be found by running tableDisplay(TABLE).

   EXAMPLE :

      id = tableValueWref('ids', N, 0.18e-6, 0.7, vdd/2, 0)

   See also tableDisplay, mosIntValue, mosWidth, tableWref
```

4.20 tableValueWrefNew

```
TABLEVALUEWREF retrieving the value of an intrinsic MOS parameter for
the reference width, directly from a given table.

   RESULT = mosIntValueWref(PARAM, TABLE, LENGTH, VGS, VDS, VSB) returns
   the value of the MOS operating point parameter PARAM
   (specified as a string) for a given TABLE.
   PARAM must be exist as a field of the given TABLE.
   Possible names for PARAM can be found by running tableDisplay(TABLE).

   EXAMPLE :

      id = tableValueWref('ids', N, 0.18e-6, 0.7, vdd/2, 0)

   See also tableDisplay, mosIntValue, mosWidth, tableWref
```

4.21 tableWcrit

```
GETWCRIT width of a MOS transistor below which narrow-channel effects play a role

   WCRIT = tableWcrit(TABLE) returns the value of the minimum
   channel width that is allowed for the transistor type that is specified by
   the given TABLE. This value is usually smaller than the reference width.

   EXAMPLE :

     wCrit = tableWcrit(N);

   See also tableLmin, tableWref
```

4.22 tableWIndepParamNames

```
TABLEWINDEPPARAMNAMES returns operating point parameters of a MOS that are
assumed to be width independent.

   stringArray = tableWIndepParamNames(TABLE) returns a cell array of
   strings with the names of operating point parameters stored in the given
   TABLE, and that are independent of transistor width above a critical width.

   EXAMPLE :

    stringArray = tableWIndepParamNames(N);

   See also tableWcrit
```

4.23 tableWmin

```
TABLEWMIN minimum channel width of a MOS transistor

   WMIN = tableWmin(TABLE) returns the value of the minimum
   channel width that is allowed for the transistor type that is specified by
   the given TABLE. This value is usually smaller than the reference width.

   EXAMPLE :

     Wmin = tableWmin(N);

   See also tableLmin, tableWcrit
```

4.24 tableWref

```
TABLEWREF gets the MOS transistor width with which the table of that MOS
is computed.

   W = tableWref(TABLE) returns the width in micrometer of the transistors with
   which the values in the given TABLE have been computed.

   EXAMPLE :

      width = tableWref(N);
```