# STANDARDIZED DATA EXCHANGE FORMATS FOR DEVICE MODELING

**Abstract:**
Saving and exchanging data between software tools for the characterization and modeling of electronic device usually is a major challenge, because all of these tools apply different data formats.
This makes it cumbersome to share data between these tools, or to flip software tools within a measurement and modeling chain.

Therefore, this paper covers the most common data format types and proposes a format which can cover all major device modeling measurement and simulation scenarios.
Much emphasis is taken to provide practical examples of the discussed data formats.

**SisConsult**
**Ingenieurbüro - Engineering Office**

**Consulting Services for Electronic Device Measurements, Data Verification and Modeling**

**Dr.-Ing. Franz Sischka**
**Birkenstr. 2**
**71106 Magstadt**
**Germany**

**+49 (0)7159 946 932**

**franz.sischka@SisConsult.de**
**www.SisConsult.de**

Sept. 2024

**C O N T E N T S**

# 1. General Requirements for Data Format Standards

When talking about a set of different CAD tools, it is mandatory to provide a seamless data exchange capability between them. However, most of these tools have proprietary data formats, which makes it difficult to swap tools or to import data into the CAD tools chain.

Many users may think of using a data format like the CSV format, what means Comma (or Character) Separated Values. Microsoft's Excel program or OpenOffice programs use this format a lot.

Since CSV is a plain text, i.e. an ASCII format, it can easily be read by a human, and its data structure is typically quite straight forward: mostly in tabular form, if required with comma-separated, blank fields to match the common number of columns and rows in that file. This makes it an attractive candidate. On the other hand, it belongs to the family of data formats with a very important drawback: it is not standardized ! /1/. Since the formatting of the data inside the CSV file is not defined, users who want to share data have to agree mutually on a formatting which is most appropriate to their actual requirements. And this means that reading data from a CSV file cannot be automated without knowing in advance this specific format agreement !
As a quite typical example of this kind of limitation, consider the import of a CSV file into Excel. Excel does not know about the format details inside the data file, and therefore, the user is typically asked about the
> kind of delimiter between the data (blank, tab, comma)
> identifier of text (e.g. double-quotes)
> the kind of decimal separator (comma or point) and -if applied- the kind of the thousands separator
> negative numbers with a leading minus or in brackets
> etc.

Therefore, a standard data format has to meet these requirements:
> includes a series of well-defined **<u>keywords</u>** which explain how to read the data file
  and where to find the data and its order
> features a defined **<u>decimal separator</u>** (comma or dot)
> includes the **<u>format revision</u>** to make its reading independent of future enhancements/changes
> applies a **<u>well-defined character</u>** set

As a side remark, when talking about saving data and data formats, it is strongly recommended to save exactly those kind of data which have been measured: for DC, it is recommended to save the currents and voltages (and not the calculated characteristics like gm for FET transistors or beta for bipolar transistors) and for AC, the recommendation is saving S-parameters, as measured by the network analyzer (and not converted data like Y- or H-parameters). These calculated numbers can always be recreated from the original meaurements easily, whereas it is often not possible to get calculated characteristics back from the original measurement or simulation results.
The general best-practice rule is: store exactly what had been measured.

# 2. Most Common Measurement Data Formats for Device Modeling

There are many software tools available for device characterization and modeling, and nearly every such tool applies its own data format. In many cases, the format is not even consistent within the same tool when it comes to storing data of different measurement or simulation scenarios. However, a few data formats can be considered as standards, and are used also by measurement instrument manufacurers for data storage. This chapter covers the most important among them.

## 2.1. MDF (Measured Data File) Format

The MDIF (Measurement Data Interchange Format) has already been introduced in the mid 1980ies by the EEsof company in Southern California, to allow the DC and S-parameter measurement data import from the Anacat measurement software into their device modeling tool Xtract /2/. The format uses US-ASCII and covers both, DC-only data (voltage and current measurements) and also two-port AC/DC data (S-parameters for a range of DC bias voltages and currents). Each data file refers to an individual measurement setup of a single device. The measurement node naming is related to the network analyzer measurement port1 and port2. In other words, for a FET transistor for example (in default common-Source mode), its Gate-Source voltage is referred to as v1, its Drain-Source voltage as v2, the Gate current as i1, and the Drain current as i2. If it was a bipolar transistor in default common-Emitter, the meaning would be: v1: Base-Emitter voltage, v2: Collector-Emitter voltage, i1: Base current and i2: Collector current.

### 2.1.1. Overview Examples

Two quick data format examples illustrate the MDF format.

#### Example 1: DC DATA

```
BEGIN DCDATA
# DC( V mA Y )
% v1           i1            v2            i2
-0.5000000     -0.000075     2.1000000     23.100000
END DCDATA
```

As documented by the first line (the line starting with the # sign), this DC data block refers to a measurement setup with voltages (in Volt) and currents (in mA), measured in the style of a twoport Y-parameter matrix measurement setup: applying voltages to both ports of the device, and measuring the corresponding currents. The next line (the line with the leading % sign), defines the sequence of the four DC measurement/stimulus data columns. In this example, assuming a FET transistor, v1 (the voltage at port1) represents the stimulating Gate-Source voltage Vgs = -0.5V, v2 (the voltage at port2) the stimulating Drain-Source voltage Vds = 2.1V, while the measured Gate current is i1 = Igs = -0.000075mA(!) and the measured Drain current is i2 = Ids = 23.1mA.

#### Example 2: AC DATA

```
BEGIN ACDATA
# DC( V A Y )   AC( MHz S MA R 50 )
% v1      i1       v2       i2
-2.0000   -0.0000  5.0000   0.0427

% F       n11x     n11y     n21x     n21y     n12x     n12y     n22x     n22y
2000.00   0.9270   -47.872  3.6336   142.25   0.0682   61.563   0.5481   -26.888
3000.00   0.8740   -68.581  3.3000   126.26   0.0916   50.598   0.5099   -37.908
4000.00   0.8145   -87.356  2.9526   112.06   0.1087   41.237   0.4653   -47.845
5000.00   0.7722   -103.42  2.6608   99.496   0.1197   33.808   0.4330   -56.951
END ACDATA
```

This sample AC data block documents a DC-biased S-parameter measurement.
Like in the example before, the first line (starting with the # sign) defines the measurement setup: a DC measurement, stimulated in Y-matrix style (applying voltages at both ports of the device, measuring the corresponding currents), and units in Volt and Ampere, while the AC part of the data file covers the frequency in MHz, S-parameters in Magnitude/Angle, and characteristic impedance R=50 Ohm. This # line is called 'Data Options' information line.

The next line has a leading % what means 'Data Format' information line. It defines the sequence of the DC bias voltage and current data. Assuming once again a FET transistor, v1 and v2 represent the stimulating Gate-Source voltage of Vgs = -2.0V and the stimulating Drain-Source voltage of Vds = 5.0V. The measured Gate current is i1 = Igs = 0A and the measured Drain current is i2 = Ids = 0.0427 A.
Then comes a second line starting with a % sign, i.e. once again a Data Format info line. It defines the columns for frequency and the S-parameters and their order

freq MAG(S11) ANGLE(S11) MAG(S21) ANGLE(S21) MAG(S12) ANGLE(S12) MAG(S22) ANGLE(S22)

The nomenclature of the voltages, currents and port parameters is as follows:

| MDF Variable | | FET Variable | Bipolar Variable |
|---|---|---|---|
| v1 | Portl DC Voltage | Vgs | Vbe |
| i1 | Portl DC Current | Igs | Ib |
| v2 | Port2 DC Voltage | Vds | Vce |
| i2 | Port2 DC Current | Ids | Ic |
| F | ACDATA Frequency | freq [units as defined in the % Data Format line] | |
| nijx nijy | | 2-Port Parameters [type and style of the complex numbers as defined in the % Data Format line] | |

## 2.1.2. The MDF format of EEsof Xtract in details

The following description refers to the classical EEsof Xtract definitions. Depending on the kind of measurement setup, DC or AC, the MDF file header is either

BEGIN DCDATA or BEGIN ACDATA

and the file footer is either

END DCDATA or END ACDATA.

These keywords enable the definition of several data blocks per MDF file. However, this special case is not discussed here further.

Going top-down the lines in the file, the line with the leading

#

denotes the 'Data Format' options.

| | |
|---|---|
| # DC (...) | DC only data |
| # AC (...) | AC data |
| # DC (...) AC (...) | AC data including DC bias information |

## 2.1.2.1. DC Data

DC data in an MDF file is represented like this:

```
BEGIN DCDATA
     # DC( [V|kV|mV|uV]    [A|mA|uA|nA|pA]    [Y|Z|H|G] )
     % v1                 i1                 v2                 i2
     port1_voltage#1      port1_current#1    port2_voltage#1    port2_current#1
     port1_voltage#2      port1_current#2    port2_voltage#2    port2_current#2
     .                    .                  .                  .
     .                    .                  .                  .
     .                    .                  .                  .
     port1_voltage#N      port1_current#N    port2_voltage#N    port2_current#N
END DCDATA
```

**# Identifies the Data Options line**

which defines:

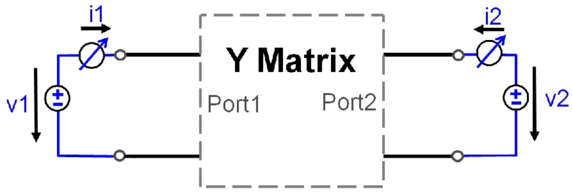
- **Voltage Data Units:**
  V Volts (default)
  kV Kilovolts
  mV Millivolts
  uV Microvolts

- **Current Data Units:**
  A Amperes
  mA Milliamperes (default)
  uA Microamperes
  nA Nanoamperes
  pA Picoamperes

- **Measurement Type:**
  Y → Independent control variables (stimuli) are v1 and v2,
  and measured are i1 and i2 (default)
  Z → Independent control variables (stimuli) are i1 and i2,
  and measured are v1 and v2
  H → Independent control variables(stimuli) are i1 and v2
  and measured are v1 and i2
  G → Independent control variables(stimuli) are v1 and i2
  and measured are i1 and v2

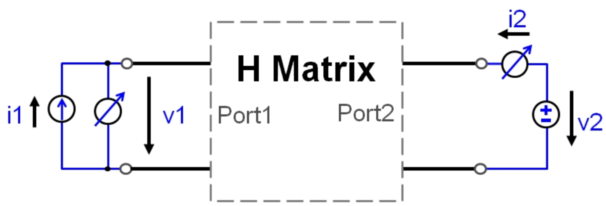
The syntax for the DC measurement setup type in the MDF format is related to the two-port theory. For the most common device modeling DC biasing, i.e. the stimulation with voltages at both ports of the device and the measurement of the currents, stands the Y-matrix, and hence, in the MDF format, the Measurement Type Y:

$$\begin{pmatrix} i1 \\ i2 \end{pmatrix} = \begin{pmatrix} y11 & y12 \\ y21 & y22 \end{pmatrix} * \begin{pmatrix} v1 \\ v2 \end{pmatrix}$$




For some bipolar transistor modeling measurements, the stimulation is with a current at port1 and a voltage at port2, measuring the port1 voltage and the port2 current, hence the H-matrix (in the MDF file the measurement type H):

$$\begin{pmatrix} v1 \\ i2 \end{pmatrix} = \begin{pmatrix} h11 & h12 \\ h21 & h22 \end{pmatrix} * \begin{pmatrix} i1 \\ v2 \end{pmatrix}$$




A characteristic of a good data format standard is the definition of defaults. For the MDF format, if no # line (data options) is present, the DC default options are:

```
DC( V mA Y )
```

The second definition line in the DC MDF file, beginning with

**% Identifies the Data Format Line**

which defines the column order for the data (v1, i1, v2, i2) in the data block. This line is mandatory.

After both lines, the Data Options and the Data Format, follow the data colums, as depicted below:

```
BEGIN DCDATA
# DC( V mA Y )
% v1  i1    v2    i2
-0.5000000  -0.000075   2.1000000   22.836000
-0.4500000  -0.000081   2.1000000   23.120000
   ...    ...    ...    ...
   ...    ...    ...    ...
   ...    ...    ...    ...
-0.1000000  -0.000310   5.0000000   83.120000
END DCDATA
```

It should be noted that for MDF, there is no preference in the data column sequence related to independent stimuli and dependent measurements or simulations. Such a preference in sequencing is quite often observed with CSV data files, where the first columns host the stimuli and the measurement/simulation result columns follow to the right.

### 2.1.2.2. Biased AC Data

While the DC data format also covers the different voltage and current biasing possibilities, the AC data format is quite straight-forward: frequency units, type of two-port parameters, format of two-port data pairs, and finally the characteristic impedance.

DC-biased two-port data in the MDF file must be represented in the following manner:

```
BEGIN ACDATA
# DC ([V|KV|mV|uV] [A|mA|uA|nA|pA] [Y|Z|H|G])   AC([GHz|MHz|KHz|Hz] [S|Y|Z|G|H] [RI|MA|DB] [R r])
% v1                    i1                      v2                      i2
port1_DC_volt_val       port1_DC_curr_val       port2_DC_volt_val       port2_DC_curr_val

% F       n11x     n11y      n21x     n21y      n12x     n12y      n22x     n22y
freq#1    n11_data_pair#1    n21_data_pair#1    n12_data_pair#1    n22_data_pair#1
freq#2    n11_data_pair#2    n21_data_pair#2    n12_data_pair#2    n22_data_pair#2
  ...          ...                ...                ...                ...
  ...          ...                ...                ...                ...
freq#N    n11_data_pair#N    n21_data_pair#N    n12_data_pair#N    n22_data_pair#N
END ACDATA
```

where

**# Identifies the Data Options line**

which defines:

**DC Bias Data Options**

DC ([ V | KV | mV | uV ] [ A | mA | uA | nA | pA ] [ Y | Z | G | H ])
for details, see the previous chapter.

**AC Data Options**

- Frequency Units
  GHz Gigahertz (default)
  Mhz Megahertz
  Khz Kilohertz
  Hz Hertz

- **2-Port Parameters used**
  **S** Scattering parameters (default)
  **Y** Admittance parameters
  **Z** Impedance parameters
  **G** Reverse-Hybrid parameters
  **H** Hybrid parameters

- **2-Port Data Pair Format used**
  RI Real-Imaginary
  MA Magnitude-Angle (default)
  DB Magnitude in dB, phase in degrees

- Characteristic Impedance
  R r where r = impedance value, typically 50 ohms (default)

If the 'Data Options' line is missing, the following default option line is assumed:

```
# DC( V mA Y )  AC( S GHz MA R 50 )
```

Right after the Data Options line comes comes the Format Line, first for the DC bias, then for the AC.

**% Indicates the Data Format Line**,

which assigns the column sequence for the DC bias data (v1, i1, v2, i2) in the DC data block. The format line is mandatory. Any DC data order is allowed.

Then follows a line with the DC bias data.
A second format line (starting with %) is used to define the order of the AC two-port data. The "F" stands for frequency and "n11x" is the label for the first part of the first two-port parameter pair, while "n11y" labels the second part. The contents of the pairs is defined in the # Data Options line.

In the example at the top of this chapter, repeated below once again for convenience, "S" and "MA" are specified in the AC Data Options line, right on the top of the data file (beginning with the # sign). Therefore, n11x is the magnitude of S11, n11y is the angle of S11, n21x is MAG(S21), n21y is ANGLE(S21) etc.

```
BEGIN ACDATA
# DC( V A Y )   AC( MHz S MA R 50 )
% v1      i1       v2       i2
-2.0000   -0.0000  5.0000   0.0427

% F       n11x     n11y     n21x     n21y     n12x     n12y     n22x     n22y
2000.00   0.9270   -47.872  3.6336   142.25   0.0682   61.563   0.5481   -26.888
3000.00   0.8740   -68.581  3.3000   126.26   0.0916   50.598   0.5099   -37.908
4000.00   0.8145   -87.356  2.9526   112.06   0.1087   41.237   0.4653   -47.845
5000.00   0.7722   -103.42  2.6608   99.496   0.1197   33.808   0.4330   -56.951
END ACDATA
```

## 2.2. Touchstone S2P (S-Parameter 2-Port) Format Definition

The high frequency circuit simulator "Touchstone", introduced in the mid 1980ies by the EEsof EDA company in California, was among the first simulators with a link to measurement data. It featured a real-world measured S-parameter data import from the then-popular Hewlett-Packard HP 8510 network analyzer to to ease the validation of simulation results with real measurements. The name of the file format was identical to the simulator name: Touchstone. Due to its compact and intuitive format, it is still one of the most often used S-parameter data formats even today.

Over the time, Touchstone files have been enhanced to also cover frequency-dependent small-signal Y-, Z-, G- or H-matrix two-port parameters. Today's Touchstone files can also host N-port parameters with 1 ≤ N ≤ 99. Therefore, a more global name for S2P files is SnP files, where n stands for the number of S-parameter ports. Additionally, the 2-port component files can now also contain high-frequency noise parameters /3,4/. The recently introduced Touchstone 2.0 format includes now also several new keyword lines like [version], [Number of Ports] etc. For details, see /5/.

The S2P description in this chapter, however, covers the classic, fundamental two-port S-parameter application.

As a small but important drawback for measurements of diodes and transistors, a Touchstone data file is –by its definition- hosting a single DC bias condition only. This information can be given by the file name itself, or inside the header section by a comment line. Therefore, there are typically a lot of per-DC-bias S2P files applied when it comes to saving transistor or diode S-parameter measurements. This important restriction is resolved by the MDM data format, described in the next chapter.

### 2.2.1. S2P Overview Example

For a better readability, the shown data samples are truncated in digits.

```
! File in Touchstone format containing S-parameter Data (MAG and Angle)
! Mon Dec 13 18:02:35 WEST 2010
# HZ S MA R 50
1.000E+08   9.096E-01   -1.710E+01   3.213E+00   1.555E+02   2.813E-02   6.625E+01   9.503E-01   -9.043E+00
1.500E+08   8.382E-01   -2.363E+01   2.940E+00   1.454E+02   3.861E-02   5.644E+01   9.089E-01   -1.219E+01
2.000E+08   7.639E-01   -2.854E+01   2.654E+00   1.369E+02   4.646E-02   4.830E+01   8.674E-01   -1.435E+01
.
.
.
1.010E+10   2.430E-01   -4.727E+00   7.818E-02   5.086E+00   4.656E-02   -3.257E+01   7.451E-01   -1.091E+02
```

In this example, there are first two (optional) comment lines. Then comes –like in the MDF case explained before- the specific data information with a Data Options line, identified by a # sign, defining the sequence and the format of the data columns. By definition, the first column represents the frequency (in the example above in [Hz]), the other columns represent the S-Parameters (in the example above in Magnitude and Angle [MA]).

In Touchstone files, the sequence of the S-parameter columns is always first the forward, then the reverse signal stimulation, i.e. S11-S21-S12-S22.

In the example above it is MAG(S11), ANGLE(S11), MAG(S21), ANGLE(S21), MAG(S12), ANGLE(S12) and MAG(S22), ANGLE(S22). Finally, the last entry to the Data Options line is related to the characteristic impedance, here R=50 Ohm.

### 2.2.2. The S2P format in details

1. Touchstone file are not case sensitive.
2. Comment lines begin with an exclamation mark [!]. They should be placed at the very beginning of the file, before the Data Options line (the line starting with the # sign ). They should not appear within the data columns.
3. The file contents is US-ASCII text, one line per frequency point, with increasing frequency. The data values are pairs of REAL numbers, and the sequence of the S-parameter data columns is always S11-S21-S12-S22. So-called engineering numbers like milli, kilo, micro etc. are not allowed.
4. The decimal point is a dot.

There may be several comment lines at the beginning of a Touchstone file.
After that, a single line is sufficient to denote the data options:

**# Identifies the Data Options line**

which defines:

- Frequency Units
  Hz Hertz
  GHz Gigahertz (default)
  Mhz Megahertz
  Khz Kilohertz

- **2-Port Parameters used**
  **S** Scattering parameters (default)
  optional,but not used very much in practice:
  **Y** Admittance parameters
  **Z** Impedance parameters
  **G** Reverse-Hybrid parameters
  **H** Hybrid parameters

- **2-Port Data Pair Format used**
  MA Magnitude-Angle in degrees (default)
  MP Magnitude-Phase in radians
  dB Magnitude in dB, phase in degrees
  RI Real-Imaginary

- Characteristic Impedance
  R r where r = impedance value, typically 50 Ohms (default)

If omitted, the default Data Options line is
# GHZ S MA R 50

Right after the Data Options line come the data columns.
This is depicted by the following data file structure definition:

```
# [Hz|kHz|MHz|GHz]   S   [MA|MP|dB|RI]   R   r]
freq#1   n11_data_pair#1   n21_data_pair#1   n121_data_pair#1   n22_data_pair#1
freq#2   n11_data_pair#2   n21_data_pair#2   n12_data_pair#2    n22_data_pair#2
                .                 .                 .                  .
                .                 .                 .                  .
freq#N   n11_data_pair#N   n21_data_pair#N   n12_data_pair#N    n22_data_pair#N
```

## 2.3. MDM (Measured Data Management) Format

To enable the separation from extraction and measurement, and to enable a seamless data exchange, the MDM data format has been introduced in the mid 1990ies to the IC-CAP device modeling software from Hewlett-Packard, now Keysight Technologies. It is again US-ASCII based, and besides the measurement or simulation data, it additionally includes information describing the measurement/simulation setups. It is also not limited to specific types of measurements like the other data formats described so far. Covering many measurement scenarios of device modeling, like DC, CV, S-parameter, nonlinear network analysis, noise, time domain etc., it is nevertheless compact and easily readable /6/.

Different to the previous data formats, data inside MDM files are all REAL numbers, no so-called engineering formats like p (pico), m (milli), k (kilo) etc. are supported. And it does not use units like Volt, Ampere, Farad etc. because this information is automatically given by the measurement/simulation setup information.

Several topics from before can be re-recognized in the MDM format:
- Comment lines start with a ! sign each.
  the BEGIN and the END statement (like in MDF)
  the VAR statement to define variables (like in the today's MDF format example of the nonlinear network analyzer before)
  the Data Format line starting with the % sign, indicating the sequence of the data columns (like in MDF)

Furthermore, the MDM file format features
- an MDM Format Version line to enable easy automatic reading also in the future
- a standardized header on top of the file describing the kind of measurement/simulation setup, how the data are stimulation and what kind of data to expect further down in the file.

The file extension for the data files is .MDM (Measured Data Management).

A specific enhancement and important feature of the MDM format over the classical MDF and Touchstone formats is the standardized header information. It covers in a well-defined format all information related to the kind of measurement/simulation, and especially the complete number of data points and the sweep types. When reading such a file by a program, all the data fields can be auto-created immediately without first going through the whole file to identify the number of data samples included. The data fields of the stimuli sources can immediately be filled with data from the header. The fields of the dependent data can then immediately be read from the MDM file's Data Block.

Different to the other so far discussed format standards, the MDM format can cover *both*, measurement specific and simulation specific details: Besides the conventional information about the measurement/simulation stimuli like voltages, stimulus currents of frequency span, it also covers measurement details like
  > current/voltage compliance
  > applied measurement instruments
and simulation details like
  > the corresponding simulation nodes.

Additionally, MDM files can hold an infinite number of devices. This is done by adding sweeps like device geometry, die location etc. around the so-called physical sweeps (voltage, current, frequency, power etc.). This means that at the end, a single MDM file can host for example id-vd measurements of an unlimited number of devices with different geometries, what makes the MDM file usable like a data base:
> individual sub-information of a single device can be read out of it.
> sweep orders can be inverted (get id-vg from a id-vd MDM file)
> spot measurements vs. geometry can be obtained easily (e.g. idsat vs. geometry from all id-vd curves of a wafer etc.)

### 2.3.1. The MDM format in details

An MDM data file hosts two main sections:
- the HEADER, and
- the DATA section

#### 2.3.1.1. The MDM HEADER Section

The header information begins with the BEGIN_HEADER keyword and ends with the END_HEADER keyword. The file header contains all the relevant information about the inputs sweeps as well as a listing of all the outputs.

The header information is comprised of two, optional three sections: ICCAP_INPUTS, ICCAP_OUTPUTS and (optional) ICCAP_VALUES:
- The mandatory ICCAP_INPUTS and ICCAP_OUTPUTS sections contain information that fully describe the measurement/simulation setup, but without the resulting measured/simulated data values (which are located in the DATA section). Optionally this section may also contain additional (usually numeric) information like the MOS transistor geometry information (length L and width W etc., capacitance area and perimeter information like AD, PD etc), temperature (TNOM, TEMP), etc.
- The optional ICCAP_VALUES section contains parameter and variable data, which usually represents additional non-numeric information like operator name, die and wafer name etc.

**The header structure is as follows:**

```
BEGIN_HEADER
 ICCAP_INPUTS
  <my_Input_Name_1>  <Input_Mode>  [<Input_Mode_Options_List>]  <Sweep_Type>
[<Sweep_Type_Options_List>]
        ...                 ...                  ...                      ...                 ...
        ...                 ...                  ...                      ...                 ...
  <my_Input_Name_X>  <Input_Mode>  [<Input_Mode_Options_List>]  <Sweep_Type>
[<Sweep_Type_Options_List>]

 ICCAP_OUTPUTS
  <my_Output_Name_1>  <Output_Mode>  [<Output_Mode_Options_List>]  <Data_Type>
        ...                     ...                  ...                      ...
        ...                     ...                  ...                      ...
  <my_Output_Name_Y>  <Output_Mode>  [<Output_Mode_Options_List>]  <Data_Type>

 ICCAP_VALUES
  <myValue_Name_1>  <Value_1>
        ...               ...
        ...               ...
  <myValue_Name_Z>  <Value_Z>
END_HEADER
```

where,

**<my_Input_Name>** is a unique but arbitrary name for an independent stimulus, e.g. for MOS the voltage names vg, vd, vs, vb, but also additional numeric information like L, W etc.

**<Input_Mode>** defines the kind of (independent) stimulus modes: V, I etc. with **[<Input_Mode_Options_List>]** a list of fields that depend on the selected <Input_Mode>.

**<Sweep_Type>** defines the kind of sweep like LIN, LOG, LIST, CON etc. with **[<Sweep_Type_Options_List>]** a list of fields that depend on the selected <Sweep_Type>.

---------------------------------

**<my_Output_Name>** is a unique but arbitrary name for a dependent measurement/simulation result like for MOS the current names id, ig

**<Output_Mode>** defines the mode of the dependent data: V, I, S etc. with **[<Output_Mode_Options_List>]** a list of fields that depend on the selected <Output_Mode>.

**<Data_Type>** M = measured data, S = simulated data, B = either

## 2.3.1.1.1. ICCAP_INPUTS

### Definitions for the fields <Input_Mode> and [<Input_Mode_Options_List>]

| **<Input_Mode>** | **[<Input_Mode_Options_List>]** |
|---|---|
| V (voltage) | <+ Node> <- Node> <Instrument Name> <Compliance> |
| I (current) | <To Node> <From Node> <Instrument Name> <Compliance> |
| W (AC power) | <Node> <- Node> <d (power in dBm)> <CharResistance> <1> <Instrument Name> <Compliance><br>or <W (power in Watt) └> '1' means the fundamental's AC power |
| F (frequency) | (no options) |
| T (temperature) | (no options) |
| P (parameter) | <Param Name> <Instrument Name> |
| U (user-defined) | <+ Node> <- Node> <Instrument Name> <Compliance> * |

Note: if entries to the Input_Mode_Options_List are unknown, enter DEFAULT.

* the user-defined Input_Mode U enables user-specific specials.

### Definitions for the fields <Sweep_Type> and [<Sweep_Type_Options_List>]

| **<Sweep_Type>** | **[<Sweep_Type_Options_List>]** |
|---|---|
| LIN | <Sweep Order> <Start> <Sstop> <Nr. Points> <Step Size> |
| LOG | <Sweep Order> <Start> <Stop> <Nr. Points per> <D (decimal Sweep Scale)> <Total Nr. Points><br>or <O (octave Sweep Scale)> |
| LIST | <Sweep Order> <Nr. Points N> <Value_1> <Value_2> … … <Value_N> |
| CON | <Value> |
| HB<br>(harmonic balance) | <Sweep Order> <Fundamental Freq.> <Nr.Harmonics> <Nr. Used Harmonics> * |
| SEG<br>(segment) | <Sweep Order> <Nr. Segments> <Nr. Per-Segment Points N> <Value_1> <Value_2> … <Value_N> |
| AC | <Magnitude> <Phase> |
| PULSE | <Initial Value> <Pulsed Value> <Delay Time> <Rise Time> <Fall Time> <Pulse Width> <Period> |
| PWL<br>(piece-wise linear) | <Nr. of Pairs N> <Time_1> <Value_1> … <time_N> <Value_N> <Start Time> <Nr.Repetitions> |
| TDR<br>(time domain reflectometer) | <Init.Value> <Pulsed Value> <Delay Time> <Rise Time> <Fall Time> <Pulse Width> <Period> <Resistance> |

* for harmonic balance simulations, 'Nr.Harmonics' represents the number of harmonics applied during simulation,
while 'Nr. Used Harmonics' represents the number of harmonics present in the MDM data file. This emulates the simulation of an unlimited number of harmonics in real-world physics, but a measurement with a limited number of harmonics applying a nonlinear network analyzer (NVNA).
Note: the entered 'Nr.Harmonics' must be equal or larger than 'Nr. Used Harmonics'.

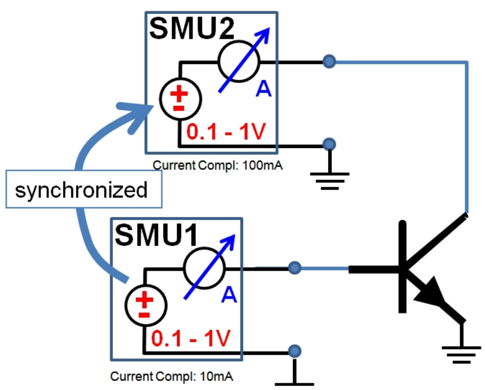
**Examples of:**
```
 ICCAP_INPUTS !============================================================
```
**full definitions:**
```
  my_Input_V     V  D    GROUND  SMU1  0.1  CON  1
  my_Input_I     I  D    GROUND  SMU1  5    CON  0.1
  my_Input_W     W  P1   GROUND  d  50  1  NVNA  10  CON  -30
  my_Input_F     F  CON  1E+09
  my_Input_F_HB  F  HB   1  1E+09  7  5
  my_Input_T     T  LIN  1  1E-06  1E-05  11  9E-07
  my_Input_P     P  L    DEFAULT  CON  1E-06
  my_Input_U  U  D   GROUND  SMU1  0.1  CON  1
```

**The same but with minimalistic definitions:**
```
  my_Input_V  V  DEFAULT  DEFAULT  DEFAULT  DEFAULT  CON  1
  my_Input_I  I  DEFAULT  DEFAULT  DEFAULT  DEFAULT  CON  0.1
  my_Input_W  W  DEFAULT  DEFAULT  d  50  1 DEFAULT  DEFAULT  CON  -30
  my_Input_F  F  CON  1E+09
  my_Input_T  T  LIN  1  1E-06  1E-05  11  9E-07
  my_Input_P  P  L  DEFAULT CON  1E-6
  my_Input_U  U  DEFAULT  DEFAULT  DEFAULT  DEFAULT  CON  1
```

A special case are **Synchronized Sweeps** in ICCAP_INPUTS**.** When a stimulus source is swept, other sources can follow, either dependingly (Sweep_Type  SYNC) or independently but with the same number of points (Sweep Type  LSYNC what stands for  List SYNChronized).

## Dependent Synchronized Sweeps

A classical example of a ***dependent*** synchronized sweep SYNC is the Gummel plot sweep of a bipolar transistor: the independent source (Master Sweep Name) is in 'Input_Mode' V (voltage sweep), while the dependent source is also in 'Input_Mode' V, and follows the master sweep with a ***fixed Offset*** of 0 and a ***Ratio*** of 1.

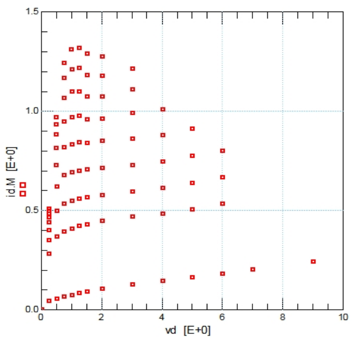



A quick example MDM file of this Gummel plot scenario is as follows:
```
BEGIN_HEADER
 ICCAP_INPUTS
! Name    | Mode | ModeOptions            | SweepType | SweepTypeOptionsList
  my_vb     V       B  GROUND  SMU1  0.1    LIN          1  0  1  11  0.1
  my_vc     V       C  GROUND  SMU2  0.1    SYNC         1  0  my_vb          ! see the table below
 ICCAP_OUTPUTS
  my_ic      I       C  GROUND  SMU2  M
END_HEADER
```

| <Mode> | **<Sweep_Type>** | **[<Sweep_Type_Options_List>]** |
|---|---|---|
| V<br>(referring e.g. to a voltage Master_Sweep) | SYNC | <Ratio>  <Offset>  <Master Sweep Name> |

## Independent Synchronized Sweeps

If the independent source (Master Sweep Name) is in ‘Input_Mode’ P (parameter sweep), an unlimited number of other parameter sweeps can follow with ***independent*** values but ***synchronized*** to each other. This is an important MDM feature when saving data with non-rectangular stimuli, what means (for example for a FET transistor) irregular, non-rectangular vg and vd biases (to cover e.g. the device power compliance), as depicted below:

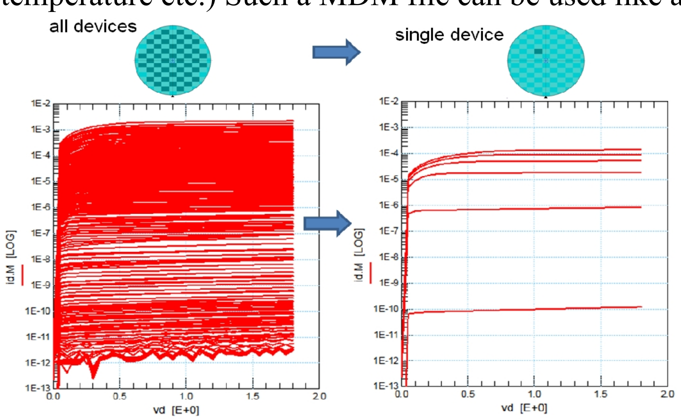



A simple, short example of a FET transistor MDM file with five syncronized, independent bias voltage triplets is:

```
BEGIN_HEADER
 ICCAP_INPUTS
! Name    Mode ModeOptions     SweepType   Sweep Options
!                                          a table of synchronized independent stimuli
  my_vg   P    vg_val  SMU1    LIST        1  5    0.3   0.3   0.5   0.5   1
  my_vd   P    vd_val  SMU2    LSYNC       my_vg   0.5   1     0.5   0.7   0.5
  my_vb   P    vb_val  SMU3    LSYNC       my_vg   0     -1    -0.5  0     -1
 ICCAP_OUTPUTS
  my_id   I   D GROUND SMU2 M
END_HEADER
```

| <Mode> | **<Sweep_Type>** | **[<Sweep_Type_Options_List>]** |
|---|---|---|
| P (referring to a Master_Sweep in mode P) | LSYNC | <Master Sweep Name> <Value_1> <Value_2> … … <Value_N> |

Another application of handling setups with independent but synchronized stimuli is saving data of a whole wafer measurement in a single MDM file. For a file with id-vd curves of differently sized MOS transistors, an extra sweep is placed on top of the voltage sweeps. This extra sweep consists of a MASTER (covering e.g. device length L) and an infinite number of LSYNC sweeps (covering e.g. device width W, wafer position, temperature etc.) Such a MDM file can be used like a data base.



## 2.3.1.1.2. ICCAP_OUTPUTS

**Definitions for the fields <Output_Mode> and [<Output_Mode_Options_List>]**

| **<Output_Mode>** | **[<Output_Mode_Options_List>]** |
|---|---|
| V | <+ Node> <- Node> <Instrument Name> |
| I | <To Node> <From Node> <Instrument Name> |
| C | <High Node> <Low Node> <Instrument Name> |
| G | <High Node> <Low Node> <Instrument Name> |
| R | <High Node> <Low Node> <Instrument Name> |
| X | <High Node> <Low Node> <Z (Impedance)> <Instrument Name><br>or <Y (Admittance)> |
| S, Y, H, Z, K, A<br>(TwoPort Parameter) | <Port 1> <Port 2> <AC Ground> <Instrument Name> |
| M<br>(MultiPort S Parameter) | <S> <Nr. of Ports N> <Port 1> <Port 2> ... <Port 2> <AC Ground> <Instrument Name> |
| N<br>(1/f Noise) | <+ Node> <- Node> <Instrument Name> |
| F<br>(RF Noise Parameters) | <NF (noise figure)> <Port> <Instrument Name><br>or <GAMMAOPT><br>or <RN (equiv. R noise)><br>or <NFMIN (min. noise figure)><br>or <TE (equiv. noise temp.)> |
| U | <High Node> <Low Node> <Instrument Name> |

**Definitions for the field <Data_Type>**

This is a single character field:

- M indicates that the output, is measured data.
- S indicates simulated data
- B indicates either.

**Examples of:**

**ICCAP_OUTPUTS !=====================================================**

**full definitions**

the example is for a FET: D=Drain, S=Source, P1=Port1, P2=Port2, P3=Port3

the used instruments are SMU2(DC), CM(cap.meter), NWA(network analyzer), DSA(dynamic signal analyzer)

```
my_Output_I        I   D   GROUND     SMU2   M
my_Output_V        V   D   GROUND     SMU2   M
my_Output_C        C   D   S          CM     M  ! capacitance measured by a cap.meter
my_Output_G        G   D   S          CM     M  ! conductance    “      “ “ “     “
my_Output_R        R   D   S          CM     M  ! resistance     “      “ “ “     “
my_Output_Z        X   D   S   Z      CM     M  ! impedance      “      “ “ “     “
my_Output_Y        X   D   S   Y      CM     M  ! admittance     “      “ “ “     “
my_Output_Spar     S   P1  P2  GROUND NWA  M
my_Output_MultiPortSpar  M   S   3 P1  P2  P3  GROUND  NWA  M
```

```
  my_Output_N        N  D  GROUND    DSA   M
  my_Output_HFNoise F  NF P2        NWA   M
  my_Output_NVNA_I  I  P2 GROUND    NVNA  M
  my_Output_NVNA_V  V  P2 GROUND    NVNA  M
```

**The same but with minimalistic definitions:**

```
  my_Output_I        I  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_V        V  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_C        C  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_G        G  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_R        R  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_Z        X  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_Y        X  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_Spar     S  DEFAULT  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_MultiPortSpar  M  S  3  DEFAULT DEFAULT DEFAULT DEFAULT
DEFAULT  M
  my_Output_N        N  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_HFNoise F  NF       DEFAULT  DEFAULT  M
  my_Output_NVNA_I  I  DEFAULT  DEFAULT  DEFAULT  M
  my_Output_NVNA_V  V  DEFAULT  DEFAULT  DEFAULT  M
```

## 2.3.1.2. The MDM Data Block Section

The data within a file is organized into multiple groups (Data Blocks) of tabular data. Each Data Block is arranged in columns representing the innermost sweep data and its associated dependent data.

The Data Block section begins with ICCAP_VAR lines, documenting the CONstant and the higher order stimulus variable names and their value for the actual Data Block.

The follows the Data Format line, starting with a % sign, and defining the sequence of the data columns.

The innermost sweep of the actual Data Block is always the first column. This is mandatory.
As a special case, any inputs with SYNC or LSYNC sweep type and with the innermost sweep as its master are listed next, to the right of the master sweep.
Note:
This is for the ease of data reading by the user, and usually not used during data reading - the information is already defined in the header.

All the remaining columns in the Data Block are columns of measured or simulated data.

The data may be either real or complex, depending on the following rules:

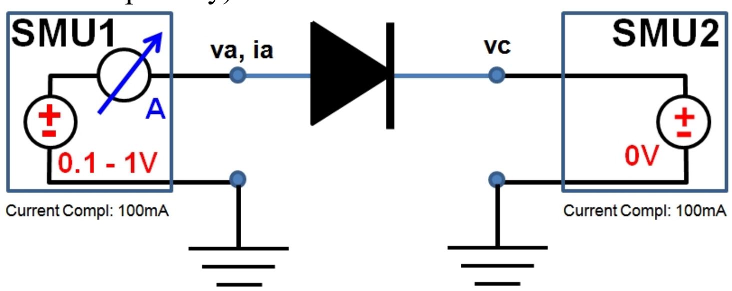
- Output_Modes V and I: depend on the sweep types of the inputs specified in the setup:
  - If the ICCAP_INPUTS don't refer to Sweep Type AC or HB, V and I output are REAL data: one column.
    Example: DC, CV or S-Parameter setups.
  - If any of the inputs in a setup have sweep type AC or HB, then V and I output modes require 2 columns,
    one for the REAL and one for IMAG data.
    Note: MDM always hosts REAL and IMAG, never MAG or PHASE !
- C, G, R and N(1/f noise) output modes are always REAL: only 1 column.
- Z, Y and RF noise output modes are always complex: 2 columns (REAL, IMAG).
- TwoPort (S, H, Z, Y, K or A) output modes are always 2-port REAL and IMAG: 8 columns.

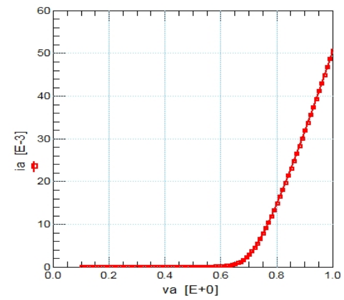
```
BEGIN_DB

 <CON_or_Higher_Order_Sweep_Name_1>  <Value_1>
        ....                            ...
        ....                            ...
 <CON_or_Higher_Order_Sweep_Name_1>  <Value_Z>

 #<sweep_1_name>            <my_Output_name_1>         ...  ...  <my_Output_name_Y>
  <sweep1_column, point_1>  <data column_1, point_1>   ...  ...  <data column_Y, point_1>
       ...                         ...                 ...  ...       ...
       ...                         ...                 ...  ...       ...
       ...                         ...                 ...  ...       ...
  <sweep1_column, point_N>  <data column_1, point_N>   ...  ...  <data column_Y, point_N>
END_DB
```

## 2.3.2. Overview Examples

### 2.3.2.1. Example 1: DC DATA

Diode DC forward measurement, applying two SMUs (Source Monitor Units, i.e. voltage supplies with current measurement capability)




```
! VERSION = 6.00
BEGIN_HEADER
 ICCAP_INPUTS
  va   V   A   GROUND   SMU1   0.1   LIN   1   0.1   1   91   0.01
  vc   V   C   GROUND   SMU2   0.1   CON   0

 ICCAP_OUTPUTS
  ia   I   A   GROUND   SMU1   M
END_HEADER


BEGIN_DB
 ICCAP_VAR vc  0
 #va      ia
  0.1     -3.2536E-11
  0.11    2.55219E-11
  0.12    -3.9527E-11
     .
     .
     .
  0.99    0.0486397
  1       0.0505508
END_DB
```

This sample MDM printout begins with the important VERSION line. As mentioned, a format which is considered to be a standard has to include this notice to enable an automatic file reading also in the future when the format may have been enhanced or changed.

The next information block, called HEADER, the most important one in a MDM file, covers all details about the stimuli (inputs to the device: ICCAP_INPUTS) as well as a listing of all the measurement/simulation results (outputs from the device: ICCAP_OUTPUTS). This block begins with the BEGIN_HEADER keyword and ends with the END_HEADER keyword. The HEADER block completely describes both, the independent stimuli conditions, and the type and kind of dependent measurement or simulation data.

This header block is a mandatory part of every MDM file.

The file header information of the above example in details:

```
BEGIN_HEADER
 ICCAP_INPUTS      ! (independent sources)
 ! Stimulus Name
 !     Stimulus Type (here: voltage)
 !        SMU +Node Name     (for simulation)  [enter DEFAULT if not defined]
 !           SMU -Node Name  (for simulation)  [enter DEFAULT if not defined]
 !                 Applied Instrument          [enter DEFAULT if not defined]
 !                       Current Compliance    [enter DEFAULT if not defined]
 !                             Sweep Type
 !                                  Sweep Order
 !                                      From
 !                                           To
 !                                              Nr. Data Points
 !                                                   Step Size
 !
  va  V  A  GROUND  SMU1  0.1  LIN  1  0.1  1  91  0.01
  vc  V  C  GROUND  SMU2  0.1  CON  0
 !
 !                                  Value (of CONstant source)
 ICCAP_OUTPUTS      ! (dependent results)
 ! Measurement/Simulation Name
 !     Measurement Type (here: Current)
 !        + Node Name      (for simulation)  [enter DEFAULT if not defined]
 !            - Node Name  (for simulation)  [enter DEFAULT if not defined]
 !                 Used Instrument           [enter DEFAULT if not defined]
 !                       Data Type           [M = measured, S = simulated]
 !
  ia  I  A  GROUND  SMU1  M
END_HEADER
```

Note: in the ICCAP_INPUTs and ICCAP_OUTPUTs, the number of column entries per Stimulus or Measurement/Simulation Name depends in the selected stimulus sweep type and measurement/simulation type. If some information is not available, a character string has to be defined nevertheless, to maintain the number of columns for the specific line. It is recommended to enter DEFAULT in this case.

As can be seen above, the MDM data format covers both, the measurement and also the simulation setup scenario, but it holds only data from either a measurement or a simulation (defined by the Data Type entry in the ICCAP_OUTPUTs).

After the HEADER block follow the data columns, starting with the keyword BEGIN_DB (Data Block) and ending with the keyword END_DB. In the simple diode example, there is only a single voltage sweep, hence a single data block.

The data block always starts with an ICCAP_VAR entry (this corresponds to the VAR statements in today's MDF format, see there) and this documents the stimulus information which is not represented by the first sweep order: here the constant cathode voltage vc = 0V.

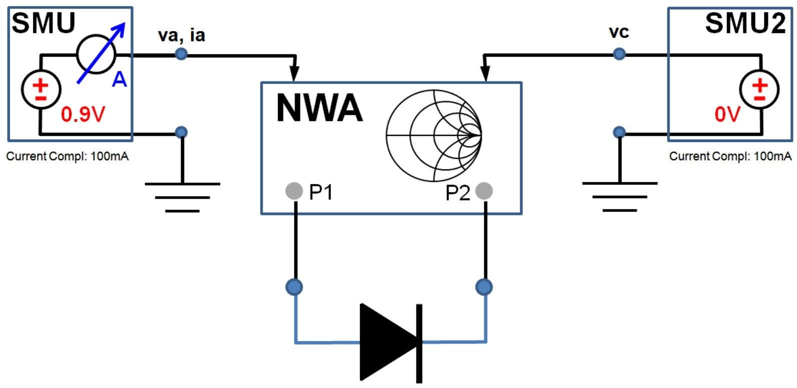
Then comes (beginning with the # sign, like in the MDF file format) the Data Options line which defines the order of the data columns. For the reader's convenience, the first order sweep is ***always*** given as the first column in the block, although this information is redundant since it is defined in the HEADER section. It's just to ease the reading of the file by a human ! The presence of this first line is mandatory.

All data are given as REAL numbers, in exponential format, without annotation of units.
The so-called engineering numbering format using p (pico), m (milli), k (kilo) etc. is ***not*** supported.

Staying with the simple diode example, but considering now an S-parameter measurement with a fixed single DC bias, is shown in the

#### 2.3.2.2. Example 2: S-PARAMETER DATA

Diode S-parameter at a fixed single DC forward bias (for simplicity)




Note: For a better readability, extra spaces have been added in the file header section and the data samples in the DATA section are truncated in digits.

```
! VERSION = 6.00
BEGIN_HEADER
 ICCAP_INPUTS
  freq F                          LIN  1   1E+09  1E+10  10  1E+09
  va   V  P1  GROUND  SMU1  0.1  CON  0.9
  vc   V  P2  GROUND  SMU2  0.1  CON  0

 ICCAP_OUTPUTS
  S   S  P1  P2  GROUND  NWA   M
  ia  I  P1      GROUND  SMU1  M
END_HEADER

BEGIN_DB
 ICCAP_VAR  va  0.9
 ICCAP_VAR  vc  0

 #freq  R:S(1,1) I:S(1,1) R:S(1,2) I:S(1,2) R:S(2,1) I:S(2,1) R:S(2,2) I:S(2,2) ia
 1E+09  0.0398   -0.0001  0.96019  -0.0008  0.9601   -0.0008  0.0398   -0.0002   1.85E-02
 2E+09  0.0398   -0.0013  0.96008  -0.0054  0.9600   -0.0054  0.0398   -0.0014   1.85E-02
 3E+09  0.0398   -0.0025  0.96003  -0.0100  0.9600   -0.0100  0.0398   -0.0026   1.85E-02
     .
     .
 9E+09  0.0157   -0.0498  0.9250  -0.2207  0.9250   -0.2207  0.0194  -0.0517   1.85E-02
 1E+10  0.0147   -0.0505  0.9236  -0.2249  0.9236   -0.2251  0.0187  -0.0524   1.85E-02
END_DB
```

Again, the HEADER covers all details about the stimuli (inputs to the device: ICCAP_INPUTS) as well as a listing of all the measurement/simulation results (outputs from the device: ICCAP_OUTPUTS). In addition to the previous DC example, the network analyzer (NWA) setup information is present too.

The detailed file header information of the example above:

```
BEGIN_HEADER
ICCAP_INPUTS      ! (independent sources)
! Stimulus Name
!        Stimulus Type (here: frequency [F] and voltage [V])
!            SMU +Node Name     (for simulation)
!                SMU -Node Name (for simulation)
!                         Used Instrument          [enter DEFAULT if not defined]
!                                Current Compliance [enter DEFAULT if not defined]
!                                     Sweep Type
!                                           Sweep Order
!                                              From
!                                                     To
!                                                            Nr. Data Points
!                                                                 Step Size
!
  freq  F                                 LIN  1  3E+08  1E+10  50  2E+08
  va    V  P1  GROUND  SMU1  0.1  CON  0.9
  vc    V  P2  GROUND  SMU2  0.1  CON  0
!
!                                          Value (of CONstant source)

ICCAP_OUTPUTS      ! (dependent results)
! Measurement/Simulation Name
!    Measurement Type (here: S-Param [S] and Current [I]
!       NWA Port1 Node Name           (for simulation)  [enter DEFAULT if not defined]
!          NWA Port1 Node Name        (for simulation)  [enter DEFAULT if not defined]
!              NWA Ground Node Name   (for simulation)  [enter DEFAULT if not defined]
!                     Used Instrument                   [enter DEFAULT if not defined]
!                            Data Type  [M = measured, S = simulated]
!
  S   S  P1  P2  GROUND  NWA   M
  ia  I  P1      GROUND  SMU1  M
!
!                SMU -Node Name      (for simulation)  [enter DEFAULT if not defined]
!        SMU +Node Name              (for simulation)  [enter DEFAULT if not defined]
END_HEADER
```

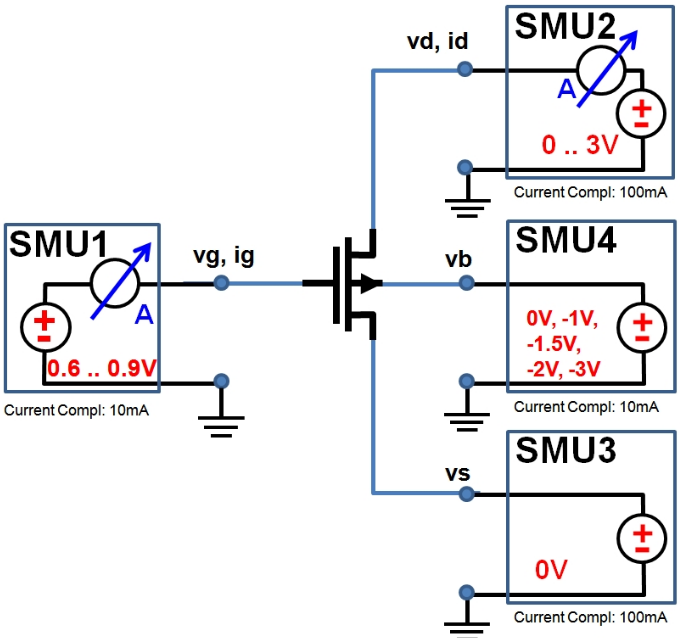
After this HEADER Block follow the data columns, starting with the keyword BEGIN_DB (Data Block) and ending with the keyword END_DB. In the simplified diode example case above, there is only a single frequency sweep, hence a single data block.

The data block starts with ICCAP_VAR entries to document the stimulus information which is not represented by the first sweep order: here both, the anode voltage va=0.9V and the cathode voltage vc = 0V.

The Data Options line (starting with an #) defines the order of the data columns, here both, the S-parameter and DC anode bias current ia.
As a recapitulation: the first order sweep for this actual data base block is ***always*** present as the first column, although this information is redundant since it is defined in the HEADER section. This is for the reader's convenience. The presence of this first line is mandatory.

## 2.3.2.3. Example 3: MOSFET id-vd DC DATA

It is a MOS Transistor id-vd DC measurement, applying four SMUs (Source Monitor Units, i.e. voltage supplies with current measurement capability). The drain voltage vd is the first sweep (this is the innermost sweep), in LINear mode, then comes the gate voltage vg as a second sweep –again in LINear mode–, and as a third sweep follows the bulk voltage vb in LIST mode (5 steps: 0V, -1V, -1.5V, -2V, -3V).




```
! VERSION = 6.00
BEGIN_HEADER
 ICCAP_INPUTS
  vd   V   D   GROUND   SMU2   0.1    LIN   1   0     3     61    0.05
  vg   V   G   GROUND   SMU1   0.01   LIN   2   0.6   0.9   5     0.075
  vb   V   B   GROUND   SMU4   0.01   LIST  3   5     0     -1    -1.5    -2   -3
  vs   V   S   GROUND   SMU3   0.1    CON   0

 ICCAP_OUTPUTS
  id   I   D   GROUND   SMU2   M
  ig   I   G   GROUND   SMU1   M
END_HEADER


BEGIN_DB
 ICCAP_VAR vg   0.6
 ICCAP_VAR vb   0
 ICCAP_VAR vs   0
 #vd     id          ig
  0      0           0
  0.05   3.778E-06   1.341E-12
  0.1    4.258E-06   -2.30E-13
  ...    ...         ...
  ...    ...         ...
  ...    ...         ...
  2.95   1.018E-05   3.452E-12
  3      1.029E-05   5.471E-12
END_DB
```

```
BEGIN_DB
 ICCAP_VAR vg   0.675
 ICCAP_VAR vb   0
 ICCAP_VAR vs   0
 #vd    id         ig
  0     1.462E-13
  0.05  2.724E-05  2.512E-12
  0.1   3.111E-05  4.112E-12
  ...   ...        ...
  ...   ...        ...
  2.95  7.317E-05  -2.122E-12
  3     7.389E-05  0
END_DB

   ...  ...   etc. incrementing vg up to 0.9, keeping vb=0
   ...  ...
   ...  ...   etc. decrementing vb following the LIST in the header section, down to -3,
   ...  ...   and for each decremented vb sweeping vg again in 5 LIN steps from 0.6 to 0.9
   ...  ...

BEGIN_DB
 ICCAP_VAR vg   0.9
 ICCAP_VAR vb   -3
 ICCAP_VAR vs   0
 #vd    id         ig
  0     3.250E-12  1.101E-12
  0.05  2.761E-08  -2.13E-12
  0.1   3.213E-08  0.121E-13
  ...   ...        ...
  ...   ...        ...
  2.95  7.617E-08  0
  3     7.692E-08  -1.234E-13
END_DB
```

**The bias stimuli (ICCAP_INPUTS) conditions in details:**

| | | | | | | |
|---|---|---|---|---|---|---|
| vd | Voltage | Drain-Ground | SMU2 | Compl.=100mA | LIN sweep 1st order | from 0V to 3V with 61 Pts, Step = 50mV |
| vg | Voltage | Gate-Ground | SMU1 | Compl.=10mA | LIN sweep 2nd order | from 0.6V to 0.9V with 5 Pts, Step = 75mV |
| vb | Voltage | Bulk-Ground | SMU4 | Compl.=10mA | LIST sweep 3rd order | with 3 Points: 0V, -1V, -3V |
| vs | Voltage | Source-Ground | SMU3 | Compl.=100mA | CONstant sweep 0V | |

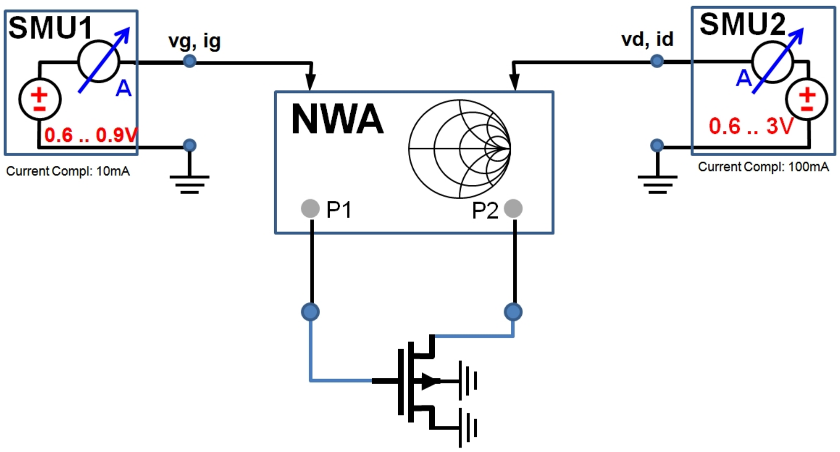
**and also the measurements (ICCAP_OUTPUTS):**

| | | | |
|---|---|---|---|
| id | current | Drain-Ground | meas. by SMU2 |
| ig | current | Gate-Ground | meas. by SMU1 |

Note:
The LIST sweep mode of the bulk voltage in the example above does not affect the Data Block structure, compared to the case if it was in LIN sweep mode. Between the individual BEGIN_DB and END_DB lines of a Data Block are always mentioned as ICCAP_VAR lines the fixed bias points of the higher order stimulus sweeps or of the constant stimuli sources, and as the first column in the Data Block the values of the current 1st order stimulus sweep. As always just given as numbers. Not mentioned at all in the Data Block is the sweep type LIN, LOG, CON, LIST etc. All this LIN, LOG etc. sweep style information is only present in the compact MDM header section !
This underlines once again the clear and clean structure of the MDM data format.

### 2.3.2.4. Example 4: MOSFET S-PARAMETER DATA with swept vd and vg

MOS transistor S-parameter measurements with first order (innermost) sweep frequency, followed by a second order vd and third order vg bias sweep, all in LINear mode. The combination of SMU DC bias and AC signal is done by the internal bias TEEs of the network analyzer (NWA). The transistor gate is connected to NWA port1 (P1), the drain to port2 (P2).




Note: For a better readability, extra spaces have been added in the file header section and the data samples in the DATA section are truncated in digits.

```
! VERSION = 6.00
BEGIN_HEADER

 ICCAP_INPUTS
  freq  F                         LIN  1  1E+08  1E+09  10  1E+08
  vd    V  P2  GROUND  SMU2  0.1  LIN  2  0.6    3      5   0.6
  vg    V  P1  GROUND  SMU1  0.01 LIN  3  0.6    0.9    5   0.075

 ICCAP_OUTPUTS
  S     S  P1  P2  GROUND  NWA   M
  id    I  P2      GROUND  SMU2  M
  ig    I  P1      GROUND  SMU1  M
END_HEADER


BEGIN_DB
 ICCAP_VAR  vd  0.6
 ICCAP_VAR  vg  0.6
 #freq  R:S(1,1) I:S(1,1) R:S(1,2) I:S(1,2) R:S(2,1) I:S(2,1) R:S(2,2) I:S(2,2) id     ig
  1E+08 0.997    -0.0577  0.00085  0.01096  -0.0140  0.01230  0.99594  -0.0867  5.4E-6 0
  2E+08 0.991    -0.1151  0.00341  0.02167  -0.0113  0.02435  0.98444  -0.1727  5.4E-6 0
  3E+08 0.981    -0.1718  0.00760  0.03191  -0.0067  0.03588  0.96540  -0.2573  5.4E-6 0
    .
    .
  1E+09 0.805    -0.5258  0.07193  0.07362  0.06305  0.08463  0.64786  -0.7526  5.4E-6 0
END_DB

BEGIN_DB
 ICCAP_VAR  vd  1.2
 ICCAP_VAR  vg  0.6
 #freq  R:S(1,1) I:S(1,1) R:S(1,2) I:S(1,2) R:S(2,1) I:S(2,1) R:S(2,2) I:S(2,2) id     ig
```

```
  1E+08 0.99793 -0.05773  0.00079  0.01096  -0.0174  0.01247  0.99680  -0.0763  6.6E-6 0
  2E+08 0.99175 -0.11513  0.00318  0.02171  -0.0148  0.02472  0.98784  -0.1521  6.6E-6 0
  3E+08 0.98149 -0.17189  0.00711  0.03205  -0.0106  0.03653  0.97298  -0.2270  6.6E-6 0
    .
    .
  1E+09 0.8002  -0.58810  0.07182  0.07111  0.06105  0.08363  0.65786  -0.7526  6.6E-6 0
END_DB
```

... ... etc. incrementing vd up to 3, in steps of 0.6, and keeping vg=0.6
... ...
... ...
... ... etc. incrementing vg 5 times up in steps of 0.075 until to 0.9,
... ... and for each incremented vg value sweeping vd again in 5 LIN steps from 0.6 to 3
... ...

```
BEGIN_DB
 ICCAP_VAR  vd  3
 ICCAP_VAR  vg  0.9
 #freq  R:S(1,1) I:S(1,1) R:S(1,2) I:S(1,2) R:S(2,1) I:S(2,1) R:S(2,2) I:S(2,2) id     ig
  1E+08 0.99252  -0.1074  0.00100  0.01065  -3.7183  0.38009  0.94538  -0.0812  3.9E-3 0
  2E+08 0.97042  -0.2123  0.0039  0.020915  -3.6414  0.75089  0.93164  -0.1610  3.9E-3 0
  3E+08 0.93459  -0.3119  0.0088  0.030409  -3.5168  1.10368  0.90928  -0.2379  3.9E-3 0
    .
    .
  1E+09 0.44749  -0.7432  0.0713  0.057585  -1.8264  2.6396   0.59101  -0.6258  3.9E-3 0
END_DB
```

**The bias stimuli (ICCAP_INPUTS) conditions in details**

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| freq | Frequency | | | | LIN | sweep 1st order | from 100MHz | to 1GHz | with 10 Pts, | Step = 100MHz |
| vd | Voltage | P2(Drain)-Ground | SMU2 | Compl.=100mA | LIN | sweep 2nd order | from 0.6V | to 3V | with 5 Pts, | Step = 0.6V |
| vg | Voltage | P1(Gate)-Ground | SMU1 | Compl.=10mA | LIN | sweep 3rd order | from 0.6V | to 0.9V | with 5 Pts, | Step = 75mV |

**and also the measurements (ICCAP_OUTPUTS):**

| | | | | |
|---|---|---|---|---|
| S | S-Parameter | between Port1 and Port2 and common Ground | meas. by a NWA | Measurements |
| id | Current | P2(Drain)-Ground | meas. by SMU2 | |
| ig | Current | P1(Gate)-Ground | meas. by SMU1 | |

Hint:
you can add both, Spar and de-embedded Spar to a single MDM file for easy data maintenance.

### 2.3.3. Summary: The MDM Format at a Glance

- the ASCII based MDM format can be applied to measurement or simulation data:
  - a header at the top of the file provides an outline of all the data in the file.
  - from the header information, the location of any data group can be computed quickly,
  - permitting rapid location of data groups scattered throughout the file.
  - all numbers are REAL, no support of so-called engineering format (p=pico, m=milli etc.)

**Header Section**

- the header describes all information related to the kind of data (DC, S-parameter, time domain, noise etc.), the complete number of data points and the sweep types.
  Also the instrument connections and the simulation nodes.
- by adding additional sweeps around a measurement/simulation,
  a MDM file can hold a quasi-infinite number of measurement of the same kind, e.g a wafer scan.
- additional variables can be stored to document auxiliary setup specifics,
  but also text for documentation.

**Data Block**

- contains all dependent data fields in blocks grouped by the $1^{st}$ order-swept stimulus.
- the first order sweep as well as the current stimulus values of the other independent sources
  are repeated in the Data Block to ease the reading of the file for the user.

# 3. References


/1/ The Internet Engineering Task Force (IETF)
http://www.ietf.org/rfc/rfc4180.txt (as of May 2013)
/2/ EEsof Xtract Manual 1993
/3/ Agilent ADS 2012 Manual, Chapter 'Touchstone SnP Format'
/4/ Touchstone File Format Specification, Rev 1.1, The EIA/IBIS Open Forum, 2002
http://www.vhdl.org/ibis (as of May 2013)
/5/ Touchstone File Format Specification Version 2.0, IBIS Open Forum, April 24, 2009
/6/ Agilent IC-CAP 2013 Manual, Chapter 'MDM File Structure'
/7/ Agilent IC-CAP 2013 mdm format generation for different setup examples , see
$ICCAP_ROOT\examples\demo_features\6_SIMULATORS\2_ADS\01_ADS_SIMULATION_TYPE_TEMPLATES.mdl