Holger Vogt

19.02.2025

**Parsing of device instances in ngspice (example: resistor)**

A major activity during netlist parsing is filling a table with device, model and node info. This table tab is locally defined in function inp\_dodeck() and is finally saved in ft\_curckt→ci\_symtab. For later setting up the matrix, an instance of each device is defined, including its nodes, and saved in a hash table for faster access.

If a netlist instance entry starting with R, function INP2R in inp2r.c is called with 3 parameters: the circuit, the line to be parsed and the table:

ngspice.exe!INP2R(CKTcircuit \* ckt, INPtables \* tab, card \* current) Zeile 220

 unter D:\Spice\_general\ngspice-master\src\spicelib\parser\inp2r.c (220)

ngspice.exe!INPpas2(CKTcircuit \* ckt, card \* data, INPtables \* tab, TSKtask \* task) Zeile 117

 unter D:\Spice\_general\ngspice-master\src\spicelib\parser\inppas2.c (117)

ngspice.exe!if\_inpdeck(card \* deck, INPtables \* \* tab) Zeile 175

 unter D:\Spice\_general\ngspice-master\src\frontend\spiceif.c (175)

ngspice.exe!inp\_dodeck(card \* deck, char \* tt, wordlist \* end, bool reuse, card \* options, char \* filename) Zeile 1414

 unter D:\Spice\_general\ngspice-master\src\frontend\inp.c (1414)

ngspice.exe!inp\_spsource(\_iobuf \* fp, bool comfile, char \* filename, bool intfile) Zeile 1088

 unter D:\Spice\_general\ngspice-master\src\frontend\inp.c (1088)

In INP2R the following code sequence is processed:

If an R model is given, add the model to tab by INPinsert(). Then check by INPgetMod() if the model is in the listed hash table modtabhash. If yes, create a new model structure by create\_model() , which by CKTmodCrt() then adds it to ckt→MODnameHash and the linked list of models ckt→CKThead[type], type being the device type (42 for resistor).

Add nodes to tab as a hash table and to ckt→CKTnodes by INPtermInsert(). Set and return unique node numbers, here stored in node1 and node2. Uniqueness is checked effectively by the hash function.

Create a new resistor (R) model if necessary, by IFC(newModel...) and CKTmodCrt(). Add it to linked list of models ckt→CKThead[type] , with type denoting the device type. Add it to the hash table ckt->MODnameHash .

Create a new instance, add it to the instance hash table ckt→DEVnameHash, and update the linked list of instances referenced by the pointer to its model by IFC(newInstance…) calling CKTcrtElt().

If a resistance has been given by for example R=1.5k, set this as instance parameter by INPpName() via GCA().

IFC(bindNode, (ckt, fast, 1, node1)) then binds a node to its place in the specified circuit (node numbers given in node1 and node2), using CKTbindNode(). In the respective instance of struct sRESinstance the storage area for the node numbers is right after the general instance struct GENinstance gen.

PARSECALL() parses the instance parameters (e.g. TC) via INPdevParse(): check if parameters are valid, obtain their value and set the parameters via CKTparam() and RESparam().

The matrix will be stamped in by scanning through the list of models ckt->CKThead[i], and for each model available scanning through the linked list of device instances using this model. The device type i will lead to the device specific matrix setup code.