How Space does Extract Bipolar Junction Transistors

S. de Graaf

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

Report EWI-ENS 09-02
September 11, 2009

Last revision: September 14, 2009.

1. INTRODUCTION

In the year 1994 Frederik Beeftink extended the space extractor for the extraction of bipolar junction transistors (BJTs). See also his technical report nr. 93-127 (December 1993) with the title "Bipolar Layout-to-Circuit Extraction, Concepts and Implementation".

There are two types of BJTs, vertical BJTs (VBJTs) and lateral BJTs (LBJTs). The implementation is used for the extraction of bipolar elements for the dimes01 and dimes03 process.

This report evaluates the current implementation and describes how it works. This report describes also new improvements and changes to the source code.

2. CURRENT IMPLEMENTATION

In the space configuration file is added a definition to make the compilation of the bipolar source code possible:

```
% cat src/space/include/config.h

/* Feature: bipolar extraction.
*/
#define BIPOLAR              /* Enable bipolar extraction. */

#ifdef PUBLIC /* definition of the public version, see ../INFO */
#undef BIPOLAR
#endif /* PUBLIC */
```

The following include files are added for bipolar extraction:

```
% cat src/space/include/bipolar.h
#define MAX_BIPOTORNAME  15
typedef struct junction  { ... } junction_t;
typedef struct vBJT      { ... } vBJT_t;
typedef struct lBJT      { ... } lBJT_t;
```

```
% cat src/space/include/polnode.h
typedef struct pnTorLink { ... } pnTorLink_t;
typedef struct nodeLink  { ... } nodeLink_t;
typedef struct pnEdge    { ... } pnEdge_t;
typedef struct polnode   { ... } polnode_t;
```

The following include files are also changed for bipolar extraction:

```
% cat src/space/include/type.h
#ifdef BIPOLAR
#include "src/space/include/bipolar.h"
#include "src/space/include/polnode.h"
#endif /* BIPOLAR */

% cat src/space/include/node.h
typedef struct Node {
    ...
    struct nodeLink * pols; /* BIPOLAR */
} node_t;

% cat src/space/include/subnode.h
typedef struct subnode {
    ...
#ifdef BIPOLAR
    struct polnode * pn;
    struct subnode * next_pn;
#endif /* BIPOLAR */
} subnode_t;
```

```
% cat src/space/include/extract.h
typedef struct resElemDef {
    ...
    int type;   /* BIPOLAR: 'a', 'n' or 'p' */
} resElemDef_t;

#ifdef BIPOLAR
typedef struct pnconElemDef { ... } pnconElemDef_t;
typedef struct junElemDef   { ... } junElemDef_t;
typedef struct bjtorElemDef { ... } bjtorElemDef_t;
#endif /* BIPOLAR */

typedef struct elemDef {
    ...
    union {
        ...
#ifdef BIPOLAR
        pnconElemDef_t pnc;
        junElemDef_t   jun;
        bjtorElemDef_t bjt;
#endif
    } s;
} elemDef_t;
```

The "bipolar.h" file limits a bjt name to 15 characters. Three data structures are defined, one for a junction and two for the BJTs (one for vertical an one for lateral). The lateral structure contains one more field (coor_t basew) for the lateral basewidth (the width between emitter/collector). Thus, it is not really useful to define two different structures. The junction structure is only used for lateral BJT recognition. A junction is only made for the right and top side of an emitter or collector polnode and the base polnode. Between these two polnodes must always be a different polarity.

The "polnode.h" file defines four other data structures. The pnTorLink for a link between a polnode and BJT structure. The nodeLink for a link between a polnode and a node structure. The pnEdge structure is used for the linked list of polnode edges, which is only used for lateral BJT recognition. Only the edges of emitter and collector polnodes are stored. The last one is the data structure of the polnode. A polnode is a node with a specific polarity type ('p' or 'n'). The polnode polarity is inherited from the conductor polarity. This polarity is specified in the technology file by the conductors. Polnodes of interconnect conductors get a non-polarity type 'a'. The polnodes define the emitter, collector, base and optional the substrate terminal areas of BJTs. A polnode has a "nodes" pointer, which points to its linked list of nodeLink structures (links to different nodes). A polnode can only have more than one node in case of resistance extraction, because the area can then be split in separate parts by resistors.

In "node.h" gets the Node structure a "pols" pointer to its linked list of nodeLink structures (links to different polnodes). Thus, one node can have different polnodes, even with different polarity. This happens, when there are contacts or connects between different conductors. Note that polnodes are only joined together when they have the

same polarity and the same conductor number.

In "subnode.h" the subnode structure has get a "pn" pointer, which points to the polnode. Also a next subnode pointer "next_pn" is added, which connects the subnodes of the polnode. If the last subnode of a polnode is deleted, then the polnode can be deleted.

In "extract.h" three new element structures are defined. One for the connect, one for the junction and one for the BJT element. However, the junction element is not used by space. And the conductor element has get a "type" field.

3. CODE IMPROVEMENTS

In the "bipolar.h" file the two BJT structures are merged. The typedef's are changed from vBJT_t and lBJT_t into BJT_t. I have deleted the sidewall field in the junction structure, because the junction is always of the sidewall type.

In the "polnode.h" file the union of the BJT structures is removed. In the pnTorLink structure the BJT structure is now addressed with "tor" in place of "t.vT" or "t.lT". In the polnode structure the pointer "j" to the junction list is renamed into "juncs" and the pointer "t" to the pnTorLink list is renamed into "tors".