

Problem 8: Subcircuit Identification

Source: Cadence Taiwan

February 15, 2000

1 Introduction

The problem of finding sub-circuits in a larger circuit arises in many contexts in computer-aided design. See Figure 1(a) for an instance of the subcircuit (a NAND gate) in the main circuit shown in Figure 1(b). This is a problem often solved using ad hoc techniques that rely on the circuit technology and implementation details. In fact, the problem of finding a subcircuit in a circuit can be transformed into that of finding a subgraph in a graph. A solution to the subgraph isomorphism problem as a general technique that is applicable across technologies and circuit domains is proposed in [1].

2 Preliminaries

Before specifying the problem, we need some background information presented in the following.

2.1 CDL and Circuit Graph

CDL is a common text format in the industry to describe a circuit design. A circuit design can be expressed in different styles. Text format such as CDL is preferable to computers while graph representation is more readable to human brains. Figure 2(a) gives a CDL description of a NAND gate whose circuit graph is shown in Figure 2(b). A circuit graph is an undirected bipartite graph with **devices**, represented by squares, forming one set of vertices, and **nets (wires)**, represented by circles, forming another set. A connection between two devices is made by connecting each device vertex to a net vertex. Each device and net (wire) have their own names. Devices have **types** which distinguish the devices according to their functions. In Figure 2(b), the MOS device has three terminals, namely drain, gate, and source; among them two of the terminals belong to class “s” (drain and source) while the other one is assigned class “g” (gate).

2.2 CDL Syntax

The syntax of CDL related to this problem is described as follows:

- 1 **.subckt** *xxx terminal List*
Specify a circuit definition.

<i>xxx</i>	The name of circuit.
<i>terminal List</i>	Nodes that can be connected from outside.

Example:

```
.subckt nand2 Z A B
```

A circuit named nand2 with three terminals Z, A, and B.

Note:

A complete design may include more than one circuit definition. Nodes not in the terminal list are called “internal nodes,” e.g., **net6**.

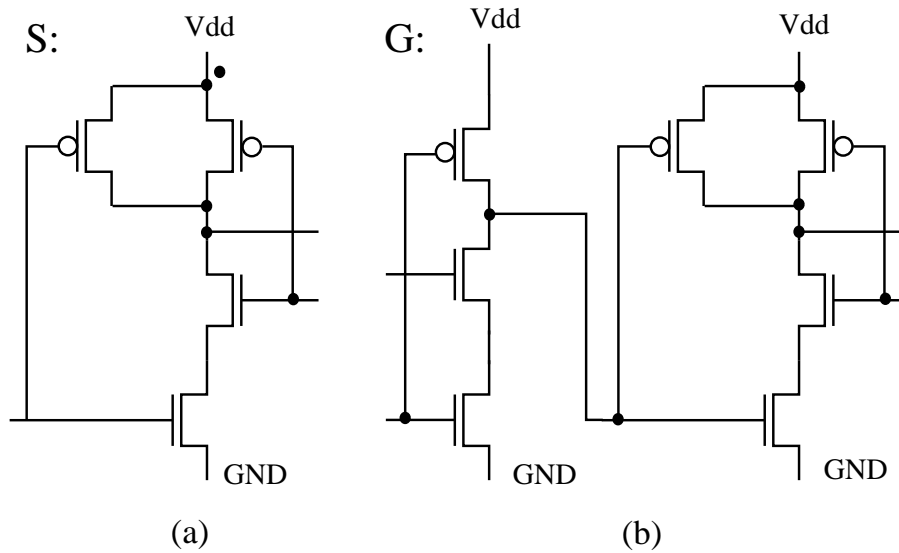


Figure 1: Transistor-level representation of an example circuit. There is one instance of the subcircuit in the main circuit at the right.

2 M_{xxx} drain gate source bulk model

Define a MOSFET device and its terminal connections.

xxx Device name.
drain The drain terminal node name.
gate The gate terminal node name.
source The source terminal node name.
bulk The bulk terminal node name.
model The device type name.

Example:

```
MN3 Z B net6 VSS n
```

An n-type MOS named N3 whose drain, gate, source, and bulk are connected to nodes Z, B, net6, VSS, respectively.

Note:

Drain and source are interchangeable, so they are in the same class “s” while gate is in class “g.” Since an n-type MOS typically connects bulk to VSS and a p-type MOS connects bulk to VDD, we can ignore the bulk connection in circuit graph. Here, VDD and VSS are called “global nodes,” meaning that they are all connected throughout the whole design.

3 X_{yyy} node_name_list subname

Define a circuit instance and its terminal connections.

yyy The instance name.
node_name_list The node names to which the instance terminals connect.
subname The circuit name of this instance

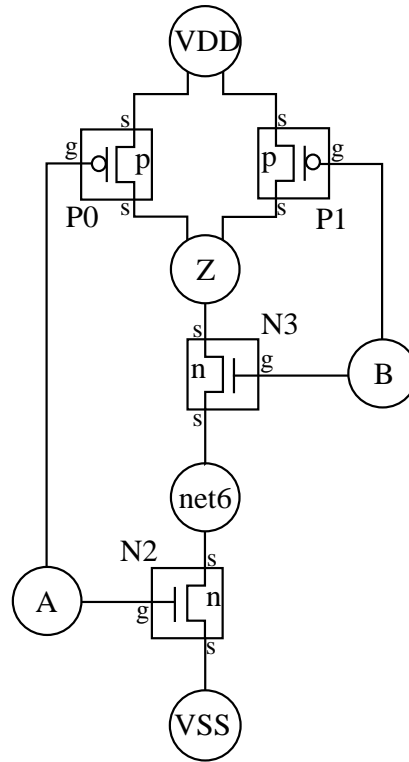
Example:

```

.subckt nand2 Z A B
MN3 Z B net6 VSS n
MN2 net6 A VSS VSS n
MP1 Z B VDD VDD p
MP0 VDDA Z VDD p
.ends nand2

```

(a)



(b)

Figure 2: (a) NAND gate description in the CDL format; (b) circuit graph of the NAND gate

```

.subckt nand2 Z A B
...
ends

.subckt TOP AA BB
XI1 net6 AA BB nand2
...
.ends

```

An instance (named I1) of nand2 whose three terminals Z, A, B are connected to nodes net6, AA, and BB in the circuit definition of TOP.

2.3 Hierarchical Structure and Flattening

With the ability of defining instance of one circuit within another circuit, the whole chip design can be represented by a tree or hierarchical structure. If we replace all the instances with their original circuit from bottom to top in a hierarchical design, we can end up with a flat circuit which consists of only primitive MOS devices. This process is called **flattening**.

Figure 4 gives a circuit flattened from the example hierarchical design shown in Figure 3. Note that the device and node names in Figure 4 were reassigned numbers. A mapping between the node numbers and

.subckt	adder	3	4	5	6	7	M21	1	6	25	1	N	M42	30	5	2	2	P
M1	20	17	19	1	N		M22	1	5	25	1	N	M43	32	7	31	1	N
M2	1	7	20	1	N		M23	25	6	26	2	P	M44	1	16	32	1	N
M3	2	17	19	2	P		M24	26	5	2	2	P	M45	2	7	31	2	P
M4	19	7	2	2	P		M25	1	11	4	1	N	M46	31	16	2	2	P
M5	1	19	9	1	N		M26	4	11	2	2	P	M47	1	31	14	1	N
M6	9	19	2	2	P		M27	1	12	3	1	N	M48	14	31	2	2	P
M7	22	5	21	1	N		M28	3	12	2	2	P	M49	34	10	33	1	N
M8	1	6	22	1	N		M29	1	9	11	1	N	M50	1	12	34	1	N
M9	2	5	21	2	P		M30	1	13	11	1	N	M51	2	10	33	2	P
M10	21	6	2	2	P		M31	11	9	27	2	P	M52	33	12	2	2	P
M11	1	21	17	1	N		M32	27	13	2	2	P	M53	1	33	13	1	N
M12	17	21	2	2	P		M33	1	15	12	1	N	M54	13	33	2	2	P
M13	1	23	10	1	N		M34	1	14	12	1	N	M55	36	5	35	1	N
M14	10	23	2	2	P		M35	12	15	28	2	P	M56	1	6	36	1	N
M15	2	7	23	1	N		M36	28	14	2	2	P	M57	2	5	35	2	P
M16	1	18	23	1	N		M37	1	29	16	1	N	M58	35	6	2	2	P
M17	23	7	24	2	P		M38	16	29	2	2	P	M59	1	35	15	1	N
M18	24	18	2	2	P		M39	1	6	29	1	N	M60	15	35	2	2	P
M19	1	25	18	1	N		M40	1	5	29	1	N	.ends					
M20	18	25	2	2	P		M41	29	6	30	2	P						

Figure 4: Description of the flattened circuit from that listed in Figure 3.

1:	VSS	5:	A
2:	VDD	6:	B
3:	CARRY	7:	C
4:	SUM		

Figure 5: Example node number mapping.

5 Output

Report instances of the subcircuit found from the given flat circuit.

nand2:	instance	1				
M1		20	17	19	1	N
M2		1	7	20	1	N
M3		2	17	19	2	P
M4		19	7	2	2	P
nand2:	instance	2				
M7		22	5	21	1	N
M8		1	6	22	1	N
M9		2	5	21	2	P
M10		21	6	2	2	P
nand2:	instance	3				
M43		32	7	31	VSS	N
M44		VSS	16	32	VSS	N
M45		VDD	7	31	VDD	P
M46		31	16	VDD	VDD	P
nand2:	instance	4				
M49		34	10	33	VSS	N
M50		VSS	12	34	VSS	N
M51		VDD	10	33	VDD	P
M52		33	12	VDD	VDD	P
nand2:	instance	5				
M55		36	5	35	VSS	N
M56		VSS	6	36	VSS	N
M57		VDD	5	35	VDD	P
M58		35	6	VDD	VDD	P

6 Language/Platform

- Language: C or C++.
- Platform: Sun OS/Solaris.

7 Evaluation

- Correctness, effectiveness (number of subcircuit instances found), performance, time and memory usage.
- Note that subcircuit identification in general is an NP-hard problem, and thus it is computationally expensive to extract all instances of a subcircuit from a general circuit. You shall consider the trade-off between efficiency and effectiveness during the tool development.

- Please expect that the problem size could reach 10K.

8 Questions

Please report any question regarding this problem to `cad@cis.nctu.edu.tw` with the email subject “CAD Contest: Problem 8.” Your question(s) will be answered in two weeks, and the Q&A’s will be posted at the contest web site. Buffers

References

- [1] Miles Ohlrich, “*SubGemini: Identifying SubCircuits using a Fast Subgraph Isomorphism Algorithm*“, *30th ACM/IEEE Design Automation Conference*, pp. 31-37, June 1993.