Computer Science
University of Manchester

Designing C-elements for Testability

Oleg Petlin, Steve Furber

Technical Report UMCS-95-10-2

Designing C-elements for Testability

Oleg Petlin, Steve Furber

Department of Computer Science
University of Manchester
Oxford Road, Manchester, U.K.
`{oleg, sfurber}@cs.man.ac.uk`

Designing C-elements for Testability

O. A. Petlin, S. B. Furber

Department of Computer Science, University of Manchester, Manchester, M13 9PL,
UK

`{oleg,sfurber}@cs.man.ac.uk`

Abstract**:** C-elements are used widely in asynchronous VLSI circuits. Fabrication faults in some CMOS C-elements can be undetectable by logic testing. Testable designs of static symmetric and asymmetric C-elements are given in this report which provide for the detection of single line stuck-at and stuck-open faults. We show that driving feedback transistors in the proposed testable static C-elements transforms their sequential functions into combinational AND and OR functions or into repeaters of one of their inputs depending on the driving logic value. This simplifies the testing of asynchronous circuits which incorporate a large number of state holding elements. The scan testable C-element described can be used in scan testing of the asynchronous circuit making the states of its memory elements controllable and observable.

Key words: Asynchronous circuit, CMOS VLSI circuit, symmetric C-element, asymmetric C-element, stuck-at fault, stuck-open fault, logic testing.

1 Introduction

Asynchronous or self-timed circuits have already demonstrated advantages over their synchronous counterparts. Some of the advantages are design flexibility, the absence of clock skew, the potential for lower power consumption and performance at the average speed rate rather than at the worst case [1,2]. A successful attempt to produce an asynchronous version of the ARM microprocessor has been reported [3]. However, before producing a fully commercial asynchronous chip one must be sure that it is fault-free after its fabrication. Testing asynchronous circuits is aggravated by the following factors [4]:

- the presence of a large number of state holding elements in asynchronous circuits makes the generation of tests harder or even impossible;
- detecting hazards and races is complicated;
- the absence of synchronization clocks decreases the level of test control over the circuit.

It has been observed that some classes of asynchronous circuits, such as delay-insensitive and speed independent circuits, are testable for a certain class of stuck-at faults. These circuits use handshaking protocols where each signal transition on a circuit line is acknowledged by another signal transition. In the presence of stuck-at faults such circuits halt. Asynchronous circuits are self-checking or self-diagnostic circuits if they exhibit no activity in the presence of stuck-at faults [5-7].

The components used to design asynchronous circuits are complicated. Some of the fabrication faults inside asynchronous components are hard or even impossible to detect by logic testing. The main building block widely used in asynchronous VLSI circuits is the Muller C-element. *Brzozowski* and *Raahemifar* showed that the testing of line stuck-at faults in different implementations of the C-element is not trivial [8]. It has been observed that line stuck-at faults of the C-element fall into one of the following categories:

- faults that are detectable by logic testing since they halt the circuit or change its function;
- faults that are detectable by delay measurements;
- faults that may result in an oscillation;
- faults that may destroy the speed-independence of the circuit;
- faults that are detectable by measuring the current.

In this report we consider different CMOS implementations of static symmetric and asymmetric C-elements for testability. The C-element designs described in this report provide for the detection of line stuck-at and transistor stuck-open faults using logic testing. The structure of the report is as follows: Section 2 discusses the testing of line stuck-at faults and transistor stuck-open faults in CMOS circuits; different implementations of static symmetric C-elements are examined in Section 3; Sections 4 and 5 present CMOS implementations of the symmetric C-element which are testable for stuck-open and stuck-at faults respectively; CMOS implementations of static asymmetric C-elements are considered in Section 6; Sections 7 and 8 present CMOS designs of asymmetric C-elements which are testable for stuck-open and stuck-at faults respectively; the C-element with scan features is considered in Section 9; cost implementation comparisons are made in Section 10; and, finally, Section 11 summarises the principal conclusions of the report. Simulation results for the extracted layouts of all the CMOS C-element designs presented in this report can be found in the Appendix.

2 Testing for fabrication faults in CMOS circuits

Stuck-at and stuck-open fault models are used to describe the effects of the majority of fabrication faults in CMOS circuits [9-11]. The stuck-at fault model assumes that a fabrication failure causes the wire to be stuck permanently at a certain logical value. Consider a fragment of a CMOS design (in Figure 1a) with possible locations of line stuck-at faults. For instance, the stuck-at one fault in node 1(1-SA1) is interpreted as a break on line 1 with the gate of *n*-type transistor N_2 connected permanently to the power supply voltage (in Figure 1b). The application of a constant voltage is marked with a cross. Fault 2-SA can be represented in three ways:

- the disconnection of transistor N_1 from node 2 and setting its source to a logical value (fault 2*'*-SA in Figure 1b);
- the disconnection of transistor N_3 from node 2 and setting its source to a logical value (fault 2*"*-SA in Figure 1c);
- the disconnection of transistor N_2 from node 2 and setting its drain to a logical value (fault 2*"'*-SA in Figure 1d).

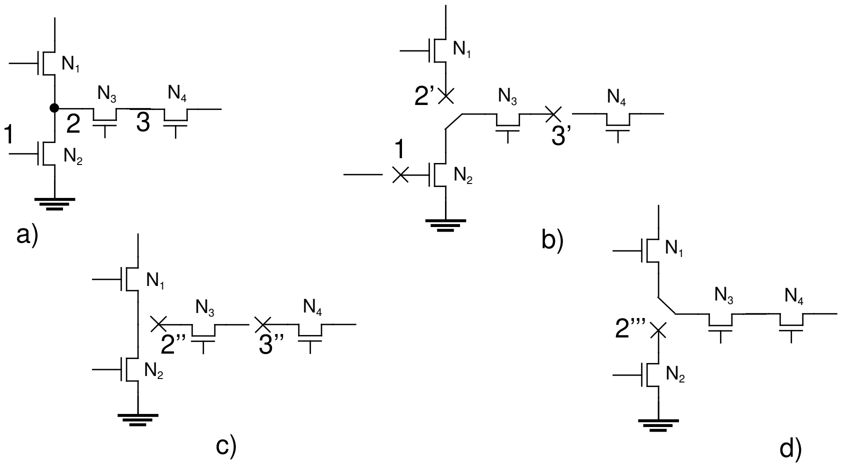


Fig. 1 : *Locations of line stuck-at faults and their interpretation in a fragment of CMOS design*

Note that fault 2’”-SA is equivalent to fault 1-SA0 when transistor N_2 is permanently off. Thus, fault 2’”-SA can be excluded for the sake of simplicity. Notations 3’-SA or 3”-SA denote a break on the left side of line 3 and setting a permanent logical value on its right side or a break on the right side of line 3 and setting a permanent logical value on its left side respectively (compare Figures 1b and 1c).

The basic CMOS inverter shown in Figure 2a consists of two types of transistors: *p*-type and *n*-type transistors [9]. When input *x* is low the *n* transistor is off and the *p* transistor is on. Output *y* is connected to the power supply voltage (V_{dd}) which corresponds to a logical one. If input *x* is high the *n* transistor is on and the *p* transistor is off. Output *y* is connected to ground (V_{ss}) which is a logical zero. Figure 2a shows line stuck-at fault locations in the CMOS inverter. For example, fault 1-SA1 of the inverter sets its output *y* to a constant logical zero. Input *x* of the inverter must be set to low to detect this fault, whereupon output *y* remains low whereas the fault-free response is high.

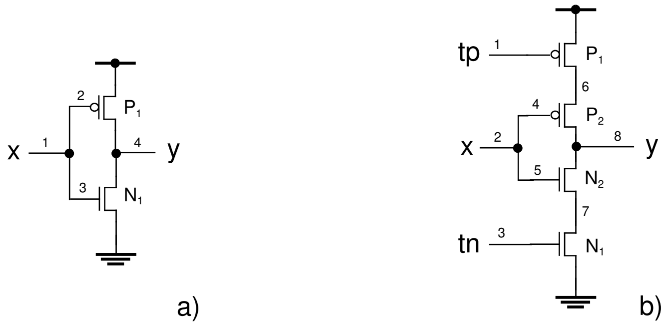


Fig. 2 : *CMOS inverters: a) inverter with two logically untestable stuck-at faults; b) testable inverter*

Table 1 : *Tests for stuck-at and stuck-open faults of the inverter in Figure 2b*

Single SA0 faults	Single SA1 faults	Single SO faults	Test sequences			Fault-free output	Faulty output
			x	tn	tp	y	y
2	8		1	1	0	0	1
8	2		0	1	0	1	0
	1,4	P_1,P_2	1	1	0	0	0
			0	1	0	1	0’
3,5		N_1,N_2	0	1	0	1	1
			1	1	0	0	1’
4			1	1	1	0	0
			1	0	0	0’	1
	5		0	0	0	1	1
			0	1	1	1’	0
1	6		1	1	1	0	0
			0	1	1	0’	1
7	3		0	0	0	1	1
			1	0	0	1’	0
6			1	1	0	0	0
			0	1	0	1	0_w
	7		0	1	0	1	1
			1	1	0	0	1_w

Consider fault 2-SA0 in the inverter illustrated in Figure 2a. This fault sets transistor P_1 permanently on. If input x is high both transistors P_1 and N_1 are on. This leads to an uncertain situation when a logical one or zero can be registered by the test circuitry depending on the strengths of the transistors. As a consequence, the detection of fault 2-SA0 cannot be guaranteed by logic testing. Similar observations can be made for fault 3-SA1.

A stuck-open fault model represents a fault effect caused by a fabrication failure which permanently disconnects the transistor pin from the circuit node. Stuck-open faults can be opens on the gates, sources or drains of transistors. In the presence of a single stuck-open fault (SO) there is no path from the output of the circuit to either V_{dd} or V_{ss} through the faulty transistor. For example, in the presence of fault P_1-SO (Figure 2a) output y cannot be set high since there is no connection between V_{dd} and node y. This fault can be identified by a set of two test patterns <T_1=1,T_2=0> applied sequentially to input x. As a result, the output of the faulty inverter remains low whereas the output of the fault-free inverter is high. Fault N_1-SO is detectable by a test set <T_1=0,T_2=1>.

Figure 2b shows a CMOS inverter which is testable for all single stuck-at and stuck-open faults. Two additional transistors (P_1 and N_1) controlled by two separate inputs are inserted into the inverter shown in Figure 2a. Table 1 contains tests for line stuck-at faults and transistor stuck-open faults in the inverter. Note that faults 4-SA0 and 5-SA1 are detectable by logic testing. For instance, a test sequence <T_1=111,T_2=100> applied to the inputs of the inverter detects fault 4-SA0. The effect of fault 6-SA0 or 7-SA1 is a 'weak zero' (0_w) or a 'weak one' (1_w) output signal respectively. These voltage levels are very close to the corresponding logical 1 and 0 voltage levels since output *y* was previously set to the same logical values. Faults 1-SA1 or 3-SA0 result in a 'floating zero' (0') or 'floating one' (1') output signal respectively. The output capacitance of the inverter can be considered as a dynamic memory element which keeps its precharged value for a certain time. It is assumed that the time between the application of two test vectors is small enough not to allow a floating output voltage level to reach the CMOS threshold level [9,11]. Hereafter we will treat weak and floating logical values as normal ones. A stuck-at fault on the gate of a CMOS transistor keeps the transistor on or off permanently depending of the type of fault. Thus transistor stuck-open faults in CMOS designs can be represented by their correspondent gate stuck-at faults. For instance, fault 5-SA0 on the gate of transistor N_2 is equivalent to fault N_2-SO (see Figure 2b). As a result, testing for stuck-at faults of the inverter illustrated in Figure 2b guarantees the detection of all its stuck-open faults.

3 Symmetric C-element CMOS designs

Figure 3a shows a symbolic representation of the two-input symmetric C-element with inputs *a*, *b* and output *c*. The symmetric C-element is a state holding element the output of which is high when its inputs are high and low when its inputs are low. Any other input combinations do not change the state of the C-element. The function performed by the two-input symmetric C-element can be written as follows:

$$c_t = a_t \cdot b_t + a_t \cdot c_{t-1} + b_t \cdot c_{t-1} \quad , \tag{1}$$

where c_t and c_{t-1} are the states of the C-element at time *t* and *t-1* respectively.

There are different ways to implement static symmetric C-elements in CMOS technology. Figure 3b shows a CMOS C-element which performs according to equation 1. For example, when *a*=1 and *b*=1 there is a path between V_{ss} and the input of inverter *inv*. As a result, output *c* of the C-element is high and feedback *n*-type transistor N_5 is on. If the inputs of the C-element are different there is always a connection between V_{ss} and the input of inverter *inv*.

Equation 1 can be rewritten in the following form:

$$c_t = (a_t + b_t) \cdot c_{t-1} + a_t \cdot b_t \quad . \tag{2}$$

A CMOS implementation of the symmetric C-element which performs according to equation 2 is illustrated in Figure 3c. This C-element works in a similar way as the one shown in Figure 3b. Both CMOS implementations of the symmetric C-element require 12 transistors.

The symmetric C-element shown in Figure 3d is a pseudo-static C-element which performs according to equation 1 but in a way similar to that of a dynamic C-element. The only difference is that the weak feedback inverter *inv2* is inserted into the symmetric C-element to create a CMOS memory. If c_{t-1}=0, a_t=1 and b_t=1 then the input of inverter *inv1* is driven to low since the strength of *n*-type input stack (transistors N_1 and N_2) is higher than that of *p*-type stack of weak inverter *inv2*. As a consequence, output *c* of the C-element goes high keeping the input of inverter *inv1* in low. If the input transistor stacks are disabled by different input signals the current state of the C-element is kept unchanged. The implementation of the pseudo-static C-element requires 8 transistors.

For test purposes we assume that the inputs of the C-element are controllable and its outputs are observable. It has already been shown in Section 2 that some of single stuck-at faults in the CMOS inverter are not detectable by logic testing. Therefore, such faults are not detectable in the CMOS designs depicted in Figure 3. There is a fundamental problem in the testing of static C-elements for stuck-open faults. Stuck-open faults in the feedback transistors of the static symmetric C-element transform it into a dynamic one. For instance, if the output of the weak inverter in the C-element shown in Figure 3d is disconnected from the input of inverter *inv1* the faulty C-element still performs according to equation 1, but as a dynamic circuit. This kind of fault can be identified only by 'slow' testing which 'waits' until the output of the faulty C-element is discharged completely. This degrades the test performance. CMOS structures of the symmetric C-element which provide for detection of single line stuck-at and transistor stuck-open faults are considered in the following sections.

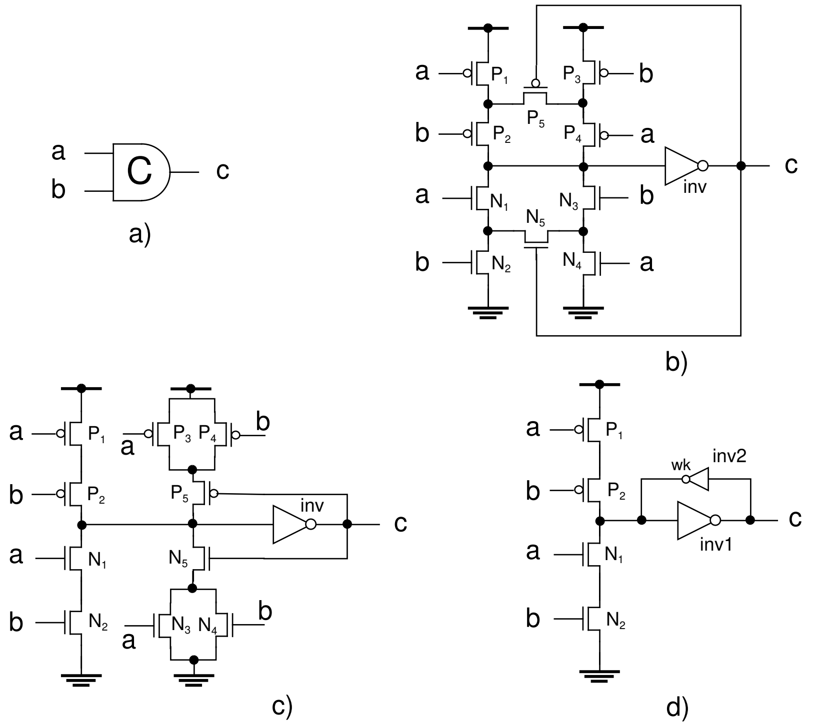


Fig. 3 : *Symmetric C-elements: a) symbol of the two-input C-element; b) and c) static C-elements; d) pseudo-static C-element*

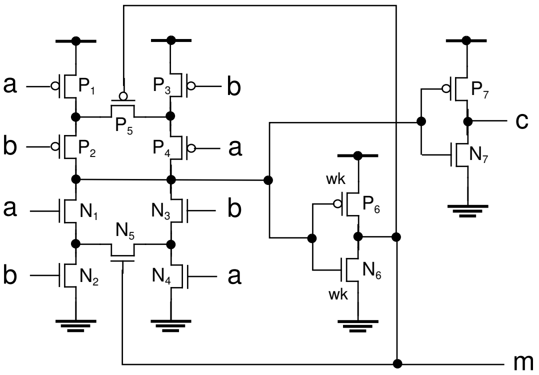


Fig. 4 : Locations of stuck-open faults in the static symmetric C-element

4 Testing for stuck-open faults

The testing of stuck-open faults in the feedback transistors of the static symmetric C-element is implemented by driving them using an extra input and observing test results on its output. The structure shown in Figure 3b is most suitable as the starting point for the testable implementation. Figure 4 illustrates a CMOS design of the symmetric C-element where single transistor stuck-open faults are detectable. The C-element contains an additional weak inverter (transistors P_6 and N_6) the output of which can be overdriven by a logical value applied to pin *m*. The proposed implementation of the symmetric C-element requires 14 transistors.

Table 2 : *Tests for stuck-open faults of the symmetric C-element in Figure 4*

Stuck-open faults	Test sequences			Outputs of the fault-free C-element		Outputs of the faulty C-element	
	a	b	m_i	c	m_o	c	m_o
P_2,P_3,P_5,N_7	1	1	0	1	-	1	-
	1	0	0	0	-	1	-
P_1,P_4,P_5,N_7	1	1	0	1	-	1	-
	0	1	0	0	-	1	-
N_2,N_3,N_5,P_7	0	0	1	0	-	0	-
	0	1	1	1	-	0	-
N_1,N_4,N_5,P_7	0	0	1	0	-	0	-
	1	0	1	1	-	0	-
P_6	0	0	z	0	0	0	0
	1	1	z	1	1	1	0
N_6	1	1	z	1	1	1	1
	0	0	z	0	0	0	1

Tests for the transistor stuck-open faults of the C-element are shown in Table 2. If m_i=0 or m_i=1 node m is used to drive feedback transistors P_5 and N_5 with a logical zero or one respectively. If m_i=z then node m is in a high impedance mode. A hyphen in the column headed m_o means that node m does not carry any diagnostic information since it is driven by an external logical value. It can be seen from Table 2 that faults P_5-SO and N_5-SO, which transform the static behaviour of the symmetric C-element into a dynamic one, are detectable by sets of two tests with no need for 'slow' testing.

Driving the feedback transistors of the symmetric C-element transforms its sequential function into a combinational one depending on the logical value applied to pin m. Table 3 contains the operation modes of the symmetric C-element illustrated in Figure 4. When m_i=0 or m_i=1 the C-element is transformed into an AND or OR gate respectively. This transformation of the sequential function of the C-element allows a reduction in the number of state holding elements in the asynchronous circuit under test and makes its testing easier. On the other hand, once the circuit (in Figure 4) performs as the symmetric C-element output m can be used as a test point increasing its observability inside the asynchronous circuit.

The use of a weak inverter in the symmetric C-element illustrated in Figure 4 is not efficient in terms of power consumption in test mode. For instance, if during the test pin m is kept to one and a=b=0 then there is a path between V_{dd} and V_{ss} which increases the power dissipation of the C-element. The power consumption can be reduced if the weak inverter is replaced by a tristate inverter which is disabled during the test by an extra control input. It is easy to show that in this case such a C-element remains testable for transistor stuck-open faults.

Table 3 : *Operation modes of the C-element in Figure 4*

Function	Inputs			Outputs	
	a	b	m_i	c_t	m_o
AND	0	0	0	0	-
	0	1	0	0	-
	1	0	0	0	-
	1	1	0	1	-
OR	0	0	1	0	-
	0	1	1	1	-
	1	0	1	1	-
	1	1	1	1	-
Symmetric Muller C	0	0	z	0	0
	0	1	z	c_{t-1}	c_{t-1}
	1	0	z	c_{t-1}	c_{t-1}
	1	1	z	1	1

5 Testing for stuck-at faults

The symmetric C-element illustrated in Figure 4 is not testable for line stuck-at faults since it has inverters which are not fully testable for all single line stuck-at faults (see Section 2). The implementation of the symmetric C-element shown in Figure 5 incorporates inverters which are testable for single stuck-at faults using two additional test inputs *tp* and *tn*. Two extra transistors controlled by inputs *tp* and *tn* are inserted in the feedback paths of the C-element for testability purposes. In normal operation mode, when *tp*=0 and *tn*=1 the circuit performs in the same manner as the one in Figure 4.

It is assumed that all the inputs and outputs of the C-element are controllable and observable during the test. Table 4 contains tests which are derived to detect its single line stuck-at faults. As was observed previously in Section 2 the detection of all single line stuck-at faults in a CMOS design guarantees the detection of all its single stuck-open faults. For instance, fault 18-SA1 is equivalent to keeping the appropriate *p*-type transistor off permanently which means its permanent disconnection from V_{dd}. As a result, the proposed CMOS design is testable for both stuck-at and stuck-open faults.

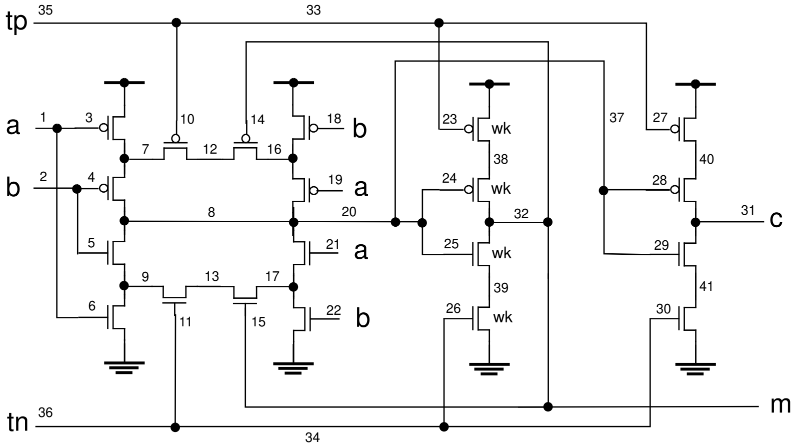


Fig. 5 : Locations of stuck-at faults in the static C-element

__Table 4 :__ Tests for stuck-at faults of the C-element in Figure 4

No	Single stuck-0 faults	Single stuck-1 faults	Test sequences			Outputs of the fault free circuit		Outputs of the faulty circuit	
			a&b	t&tp	m_i	c	m_o	c	m_o
1	31,32,38,40	8,20,37	11	10	z	1	1	0	0
2	8,20,37	31,32,39,41	00	10	z	0	0	1	1
3	7’,12’,16’,30	2,4,7”,10,12”, 14,18	11	10	0	1	-	1	-
			10	10	0	0	-	1	-
4	7”,12”,16”	1,3,10,14,12’, 16’,19	11	10	0	1	-	1	-
			01	10	0	0	-	1	-
5	2,5,9”,11,13”, 15,22	9’,13’,17’,27	00	10	1	0	-	0	-
			01	10	1	1	-	0	-
6	1,6,11,13’,15, 17’,21	9”,13”,17”	00	10	1	0	-	0	-
			10	10	1	1	-	0	-
7	25,26,29,30, 34,36		11	10	z	1	1	1	1
			00	10	z	0	0	1	1
8		23,24,27,28, 33,35	00	10	z	0	0	0	0
			11	10	z	1	1	0	0
9	4,10,18	16”	11	10	0	1	-	1	-
			01	11	0	1	-	0	-
10	3,10,19	7’	11	10	0	1	-	1	-
			10	11	0	1	-	0	-
11	17”	5,11,22	00	10	1	0	-	0	-
			10	00	1	0	-	1	-
12	9’	6,11,21	00	10	1	0	-	0	-
			01	00	1	0	-	1	-
13	14		11	00	1	1	-	1	-
			10	00	1	1	-	1	-
			10	11	0	1	-	0	-
14		15	00	11	0	0	-	0	-
			10	11	0	0	-	0	-
			10	00	1	0	-	1	-
15	23,27,33,35	38,40	00	10	z	0	0	0	0
			11	11	z	0	0	1	1
16	39,41	26,30,34,36	11	10	z	1	1	1	1
			00	00	z	1	1	0	0
17	24,28		00	11	z	0	0	0	0
			00	00	z	0	0	1	1
18		25,29	11	00	z	1	1	1	1
			11	11	z	1	1	0	0

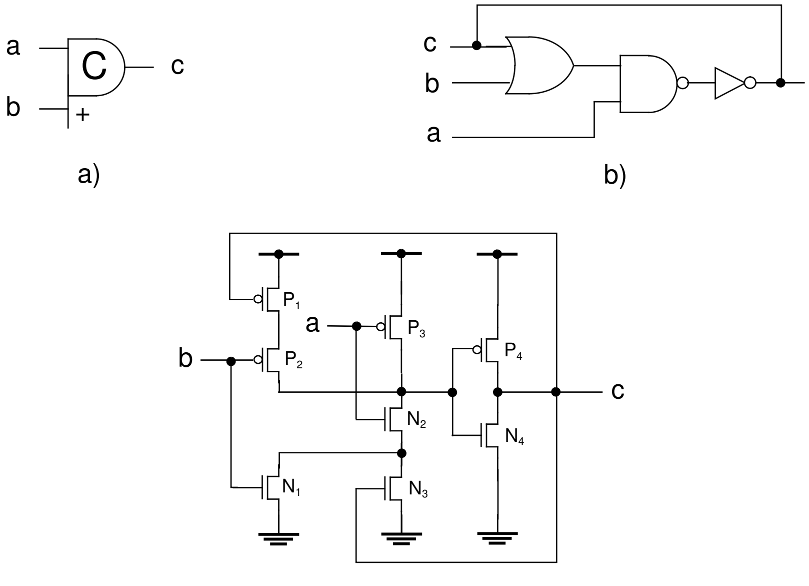
6 Static asymmetric C-elements

So-called asymmetric C-elements are used widely to improve the performance of the asynchronous control logic in micropipelines [8,12]. These C-elements are set to a particular state when both signals are one or zero and set to the negated state only by one input. Figures 6a and 6b demonstrate a symbolic representation and a gate level implementation of the OR-AND type asymmetric C-element. The OR-AND type asymmetric C-element performs according to the following equation:

$$c_t = a \cdot (b + c_{t-1}) \quad . \tag{3}$$

The output of such an asymmetric C-element is set to high when both its inputs are high and set to low if its input *a* is low. It keeps its current state when its input *a* is high and input *b* is low. A static CMOS implementation of the OR-AND type asymmetric C-element is illustrated in Figure 6c. Figures 7a and 7b show a symbolic representation and a gate level implementation of the AND-OR type asymmetric C-element. This C-element performs according to the following equation:

$$c_t = b + a \cdot c_{t-1} \quad . \tag{4}$$



a) b)

c)

Fig. 6 : *Static OR-AND type asymmetric C-element: a) symbol; b) gate level representation; c) CMOS implementation*

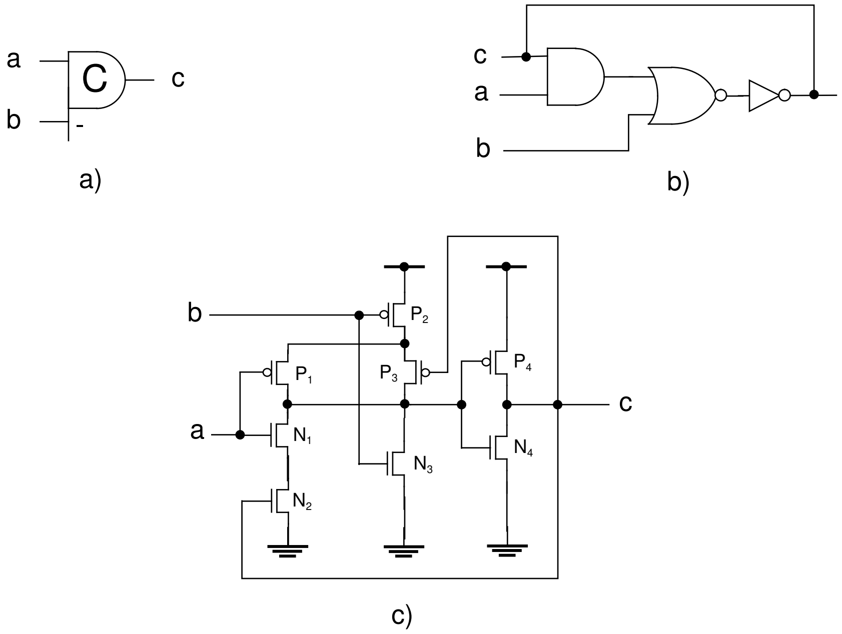


Fig. 7 : *Static AND-OR type asymmetric C-element: a) symbol; b) gate level representation; c) CMOS implementation*

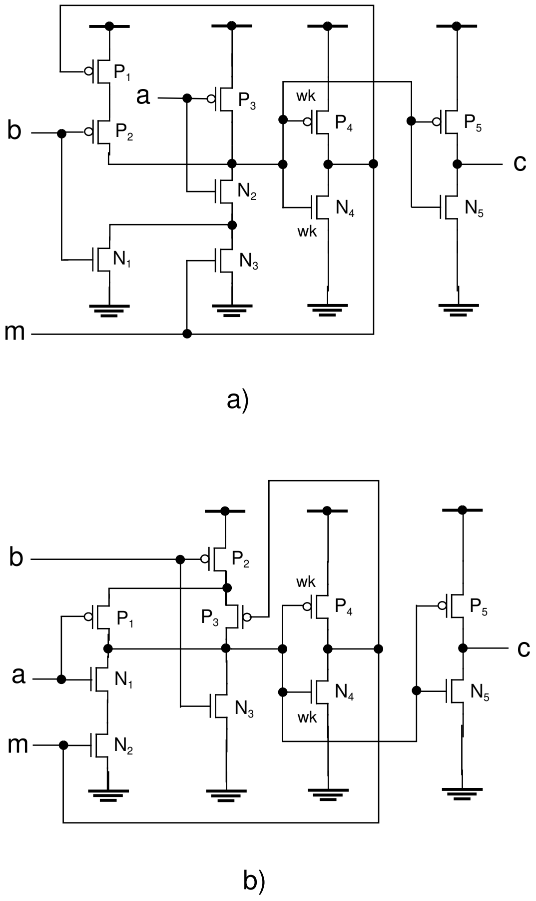
The output of the AND-OR type asymmetric C-element is set to high if its input *b* is high and set to low when both its inputs are low. It preserves its current state when input *a* is high and input *b* is low. A static CMOS implementation of the AND-OR type asymmetric C-element is shown in Figure 7c.

These two types of asymmetric C-elements were implemented in CMOS technology on a 1 μm , double layer metal CMOS process and simulated using *SPICE* analysis in *Cadence* CAD environment. Simulation results obtained from their extracted layouts can be found in Appendix.

Testing for stuck-at and stuck-open faults in asymmetric C-elements is not easy. For instance, such stuck-open faults as N_3-SO (in Figure 6c) and P_3-SO (in Figure 7c) can be identified only by 'slow' testing since they transform the correspondent static asymmetric C-elements into dynamic ones. As it was previously mentioned a stuck-at-0 fault on the gate of transistor P_4 and stuck-at-1 fault on the gate of transistor N_4 of the asymmetric C-elements are not detectable by logic testing. Thus, extra design efforts must be involved to make the asymmetric C-elements testable.

7 Testing for stuck-open faults in asymmetric C-elements

Figures 8a and 8b illustrate the designs of asymmetric C-elements testable for transistor stuck-open faults. The main approach to testing of stuck-open faults in asymmetric C-elements is the same: the feedback transistors are driven by external control input *m* making them controllable. For this purpose an additional weak inverter (transistors P_4 and N_4) is inserted into the original designs of asymmetric C-elements. Its output can be overdriven by applying a logical value to its output *m*. If no logical values are applied to output *m* then it can be used as a test point during the test.



a)

b)

Fig. 8 : *Static asymmetric C-elements testable for stuck-open faults: a) OR-AND type asymmetric C-element; b) AND-OR type asymmetric C-element*

Table 5 : *Tests for stuck-open faults of the OR-AND type asymmetric C-element*

Stuck-open faults	Test sequences			Outputs of the fault-free C-element		Outputs of the faulty C-element	
	a	b	m_i	c	m_o	c	m_o
P_1,P_2,N_5	1	1	0	1	-	1	-
	1	0	0	0	-	1	-
N_1,N_2,P_5	0	1	0	0	-	0	-
	1	1	0	1	-	0	-
N_3	0	0	1	0	-	0	-
	1	0	1	1	-	0	-
P_4	0	1	z	0	0	0	0
	1	1	z	1	1	1	0
P_3,N_4	1	1	z	1	1	1	1
	0	1	z	0	0	1	1

Table 6 : *Tests for stuck-open faults of the AND-OR type asymmetric C-element*

Stuck-open faults	Test sequences			Outputs of the fault-free C-element		Outputs of the faulty C-element	
	a	b	m_i	c	m_o	c	m_o
P_1,P_2,N_5	1	0	1	1	-	1	-
	0	0	1	0	-	1	-
P_3	1	1	0	1	-	1	-
	1	0	0	0	-	1	-
N_1,N_2,P_5	0	0	1	0	-	0	-
	1	0	1	1	-	0	-
N_3,P_4	0	0	z	0	0	0	0
	0	1	z	1	1	0	0
N_4	0	1	z	1	1	1	1
	0	0	z	0	0	0	1

Tests for the transistor stuck-open faults of the asymmetric C-elements (in Figure 8) are shown in Tables 6 and 7. Denotations of these tables have the same meanings as the ones used in Table 2. As it follows from Tables 6 and 7 all stuck-open faults of the asymmetric C-elements are detectable by only five tests which include two sequential tests each.

***Table 7 :** Operation modes of asymmetric C-elements shown in Figure 8*

Function of OR-AND type C-element	Function of AND-OR type C-element	Inputs			Outputs of OR-AND type C-element		Outputs of AND-OR type C-element	
		a	b	m_i	c_t	m_o	c_t	m_o
AND	B-repeater	0	0	0	0	-	0	-
		0	1	0	0	-	1	-
		1	0	0	0	-	0	-
		1	1	0	1	-	1	-
A-repeater	OR	0	0	1	0	-	0	-
		0	1	1	0	-	1	-
		1	0	1	1	-	1	-
		1	1	1	1	-	1	-
OR-AND type asymmetric C-element	AND-OR type asymmetric C-element	0	0	z	0	0	0	0
		0	1	z	0	0	1	1
		1	0	z	c_{t-1}	c_{t-1}	c_{t-1}	c_{t-1}
		1	1	z	1	1	1	1

The designs of asymmetric C-elements testable for transistor stuck-open faults were implemented on a 1 μm , double layer metal CMOS process and simulated using *SPICE* analysis in *Cadence* CAD environment. Simulation results obtained from their extracted layouts can be found in Appendix.

Driving the feedback transistors allows the sequential functions of the asymmetric C-elements to be changed into combinational ones. Table 7 shows the operation modes of the C-elements illustrated in Figure 8. If output *m* of the OR-AND type asymmetric C-element is driven by a logical zero or one then its function is transformed into an AND gate or a repeater of its input *a* respectively. When output *m* of the AND-OR type asymmetric C-element is overdriven by a logical one or zero its sequential function is transformed into an OR gate or a repeater of its input *b* respectively. These properties of the asymmetric C-elements (in Figure 8) can be used to simplify the testing of asynchronous circuits by reducing the number of their state holding elements.

8 Testing for stuck-at faults in asymmetric C-elements

As was mentioned previously the stuck-open testability of CMOS circuits does not guarantee the detection of all their line stuck-at faults. In order to make the asymmetric C-elements illustrated in Figure 8 testable for line stuck-at faults two additional inputs *tp* and *tn* are required. Figure 9 shows a CMOS implementation of the static OR-AND type asymmetric C-element testable for stuck-at faults. In normal operation mode inputs *tp* and *tn* are set to zero and one respectively and the circuit exhibits the same behaviour as the one demonstrated in Figure 8a.

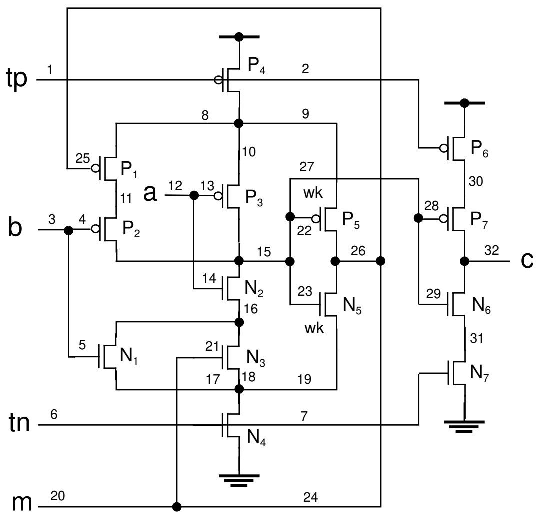


Fig. 9 : *Static OR-AND type asymmetric C-element testable for stuck-at faults*

Table 8 contains tests to detect the single line stuck-at faults whose locations are shown in Figure 9. It is presumed that all the inputs and outputs of the asymmetric C-element are controllable and observable during the test. The C-element illustrated in Figure 9 is testable for its stuck-open faults since it is fully testable for its stuck-at faults.

The asymmetric C-element testable for single line stuck-at faults was implemented on a 1 μm , double layer metal CMOS process and simulated using *SPICE* analysis. Simulation results obtained from its extracted layout can be found in Appendix.

Table 8 : *Tests for stuck-at faults of the asymmetric C-element in Figure 9*

No	Single stuck-0 faults	Single stuck-1 faults	Test sequences			Outputs of the fault-free circuit		Outputs of the faulty circuit	
			a& b	tp& tn	m_i	c	m_o	c	m_o
1	9,20,24,26 ,30,32	15,27	11	01	z	1	1	0	0
2	15,27	19,20,24,26,31, 32	00	01	z	0	0	1	1
3	7,23,29	1	11	01	z	1	1	1	1
			01	01	z	0	0	1	1
4	6	2,22,28	01	01	z	0	0	0	0
			11	01	z	1	1	0	0

***Table 8** : Tests for stuck-at faults of the asymmetric C-element in Figure 9*

No	Single stuck-0 faults	Single stuck-1 faults	Test sequences			Outputs of the fault-free circuit		Outputs of the faulty circuit	
			a&b	tp&tn	m_i	c	m_o	c	m_o
5		3,4,25	11	01	0	1	-	1	-
			10	01	0	0	-	1	-
6		12,13	1x	01	1	1	-	1	-
			0x	01	1	0	-	1	-
7	3,5		01	01	0	0	-	0	-
			11	01	0	1	-	0	-
8	12,14,21		00	01	1	0	-	0	-
			10	01	1	1	-	0	-
9	1		11	11	0	1	-	1	-
			01	11	0	1	-	0	-
10	2		01	01	0	0	-	0	-
			11	11	0	0	-	1	-
11		6	00	01	1	0	-	0	-
			10	00	1	0	-	1	-
12		7	11	01	1	1	-	1	-
			01	00	1	1	-	0	-
13	22,28		00	01	z	0	0	x	x
			00	11	z	0	0	0	0
			00	00	z	0	0	1	1
14		23,29	11	01	z	1	1	x	x
			11	00	z	1	1	1	1
			11	11	z	1	1	0	0
15	25		11	01	1	1	-	1	-
			10	00	1	1	-	1	-
			00	11	1	1	-	0	-
16	4,13		11	01	0	1	-	1	-
			11	00	0	1	-	1	-
			01	11	0	1	-	0	-
17		14	00	01	1	0	-	0	-
			00	11	1	0	-	0	-
			10	00	1	0	-	1	-
18		5,21	00	01	0	0	-	0	-
			10	11	0	0	-	0	-
			11	00	0	0	-	1	-
19		8,10,11	11	01	0	1	-	1	-
			00	11	0	1	-	0	-
20		9,30	01	01	z	0	0	0	0
			11	11	z	0	0	1	1

Table 8 : Tests for stuck-at faults of the asymmetric C-element in Figure 9

No	Single stuck-0 faults	Single stuck-1 faults	Test sequences			Outputs of the fault-free circuit		Outputs of the faulty circuit	
			a&b	tp&tn	m_i	c	m_o	c	m_o
21	8,11		01	01	0	0	-	0	-
			10	01	0	0	-	1	-
22	10		01	01	1	0	-	1	-
23	19,31		11	01	z	1	1	1	1
			01	00	z	1	1	0	0
24	16,17,18		01	01	1	0	-	0	-
			11	00	1	0	-	1	-
25		17	01	01	z	0	0	0	0
			11	01	z	1	1	0	0
26		16,18	10	01	1	1	-	0	-

9 Scan testing of C-elements

Scan testing has already become a standard methodology for testing VLSI circuits [13]. Several attempts to implement a scan test in asynchronous circuits have been published [14-16]. Scan testing presumes that the circuit is set to scan test mode where all its state holding elements are connected together forming a united scan chain. The scan path can be controlled either synchronously or asynchronously [16]. As a consequence, the states of all memory elements are controllable and observable.

State holding elements of a scan testable circuit must operate at least in two modes: normal and scan test modes. In normal operation mode, the circuit performs according to its specification. During the scan test, the test patterns are loaded into the state holding elements and the test results are shifted out of the circuit. Figure 10 illustrates a CMOS implementation of the pseudo-static symmetric C-element with scan features. It contains two additional control inputs: clock (*Clk*) and scan test (*T*) signals. Inputs *Sin* and *Sout* are used to scan the test pattern in and scan the state bit out of the C-element. Output *Sout* of each scan testable C-element (or any other scan testable memory block) is connected to input Sin of its successor forming the scan chain.

In normal operation mode, when *T*=0 and *Clk*=0 the C-element performs as the pseudo-static C-element depicted in Figure 3d. In scan mode, the input transistor stack is disabled by input *T* set to high. Clock signals are generated on input *Clk* to shift the test pattern from input *Sin* into the C-element. When signal *Clk* goes high the output transistor stack of the C-element is disabled and nodes *c* and *nc* are controlled from input *Sin*. Once the clock signal is low the negated bit loaded from input *Sin* is stored in the C-element and is passed to its output *Sout*. Clock signals generated on input *Clk* are used to shift the state bit of the C-element through the scan path to the test circuitry. When *Clk*=1 output *Sout* keeps its current logical value creating a dynamic memory and

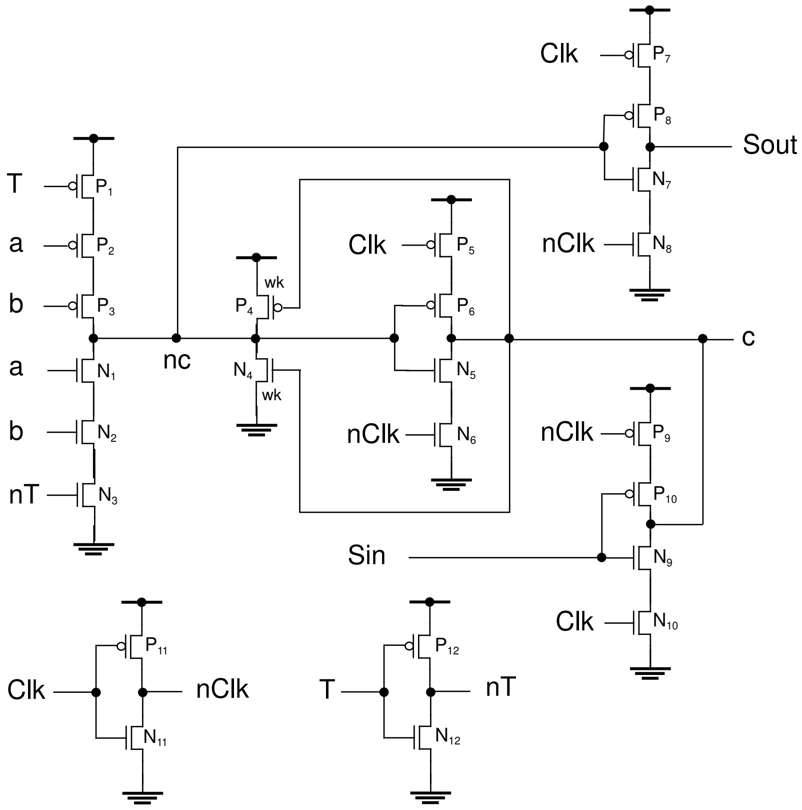


Fig. 10 : *Pseudo-static symmetric C-element with scan features*

supplying input *Sin* of the following memory element. Clock signals must be kept high for enough time to guarantee the proper transmission of logical voltage levels.

An analysis of the symmetric C-element shown in Figure 10 reveals that it is testable for single transistor stuck-open faults. Table 9 contains tests for detecting stuck-open faults of the C-element. Symbol ‘x’ denotes a ‘don’t care’ signal. A hyphen means that the appropriate output is not used to observe the test results. Note that the fundamental problem of testing stuck-open faults in the weak feedback inverter of the pseudo-static C-element no longer exists since the weak transistors of the scan testable C-element participate in the scanning of the test data.

Consider an implementation of the scan testable CALL element in order to demonstrate how the C-element shown in Figure 10 can be used to build more complex scan testable asynchronous blocks. The CALL element is an event driven logic block [17]. It remembers which of its inputs received an event first (*R1* or *R2*) and acknowledges the completion of the called procedure by an appropriate event on the matching output (*D1* or

D2). The CALL element with scan features shown in Figure 11 performs using the two-phase signalling protocol where each signal transition denotes an event. All the inputs and outputs are initialized to zero. *T*=0 and *Clk*=0 in normal operation mode. When a rising request signal is generated on input *Ri* it primes C-element *Ci* and passes through the XOR gate producing a rising request signal on its output *R*. Once the required procedure completed an appropriate acknowledge event is generated on input *D*. As a result, C-element *Ci* is set to one and a rising acknowledge signal is passed to output *Di* (i=1,2). The performance of the CALL element is identical for falling request events.

If the CALL element shown in Figure 11 is incorporated into an asynchronous VLSI circuit its internal states can be controllable and observable through the scan path. A test bit sent to input *Sin* of the C-element is negated on its outputs *Sout* and *c* (see Figure 10). The CALL element is tested by setting test control signal *T* to one. For instance, the clocked sequence 01 must be applied to input *Sin* of the CALL element in order to set its C-elements to one. The CALL element can perform its specified function when signals *T* and *Clk* are returned to zero. The state bits of the C-elements are shifted out of the CALL element and compared with known responses when its input *T* is set one and clocks are produced on its input *Clk*.

Table 9 : *Tests for stuck-open faults of the C-element in Figure 6*

Single stuck-open faults	Test sequences				Outputs of the fault-free C-element		Outputs of the faulty C-element	
	a&b	T	Clk	Sin	c	Sout	c	Sout
P_1-P_3,N_5,N_6	11	0	0	x	1	-	1	-
	00	0	0	x	0	-	1	-
P_5,P_6,P_{12},N_1-N_3	00	0	0	x	0	-	0	-
	11	0	0	x	1	-	0	-
P_7-P_{10},N_4,N_{11}	xx	1	1	1	-	x	-	x
	xx	1	0	1	-	0	-	0
	xx	1	1	0	-	1	-	0
	xx	1	0	0	-	1	-	0
P_4,P_{11},N_7-N_{10}	xx	1	1	0	-	x	-	x
	xx	1	0	0	-	1	-	1
	xx	1	1	1	-	0	-	1
	xx	1	0	1	-	0	-	1
N_{12}	00	0	0	x	0	0	0	0
	11	1	0	x	0	0	1	1

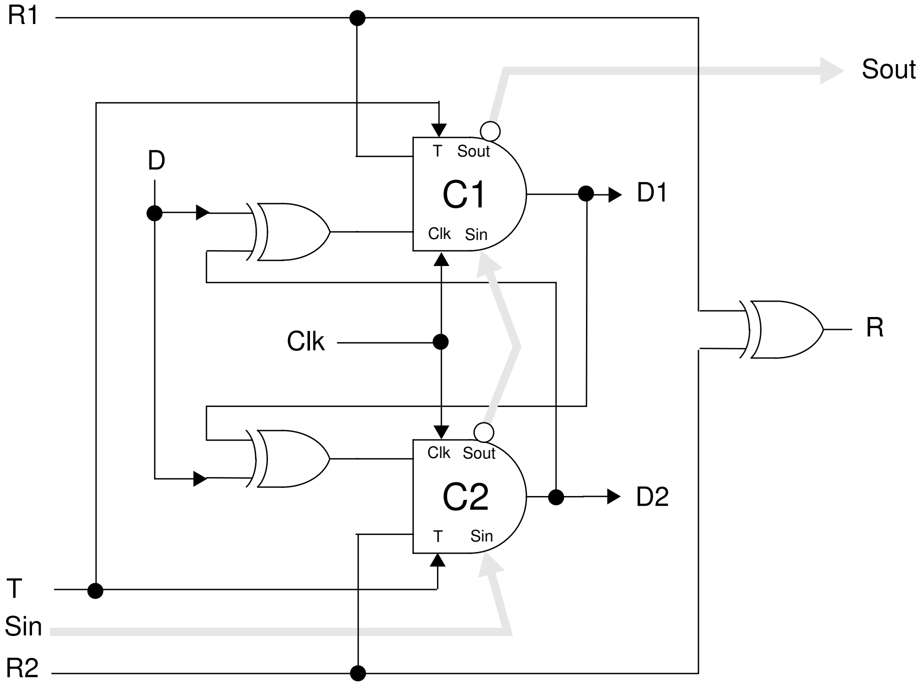


Fig. 11 : CALL element with scan features

10 Cost comparisons

The testable structures of the symmetric and asymmetric C-elements presented in this report require different overheads depending on the fabrication faults to be detected. The designs of testable C-elements have been implemented on a 1 μm CMOS process and their extracted layouts have been investigated using SPICE analyses. Table 10 contains a summary of cost implementation comparisons of the CMOS C-elements and their testability. The largest number of transistors is required to implement the scan testable C-element. This is because scanning the data through the C-element can be implemented only in a master-slave manner which requires at least two memory elements. The implementation of the symmetric C-element shown in Figure 3 has just 17% layout overhead with one extra control input and guarantees the detection of all its stuck-open faults. The sequential function of such a C-element can be changed into a combinational one (AND or OR) which simplifies the testing of other components incorporated in the asynchronous circuit. The asymmetric testable C-elements illustrated in Figure 8a and 8b guarantee the detection of all their transistor stuck-open faults and require 19% and 32% layout overheads respectively.

Table 10 : *Summary of costs of the testable C-elements*

Design	No. of transistors	No. of extra inputs/ outputs	Transistor overhead	Layout overhead	Output nodal capacitance $\times 10^{-14} F$	Testability
Figure 3	14	1	17%	17%	2.07	SO
Figure 4	20	3	67%	45%	3.21	SA& SO
Figure 8a	10	1	25%	19%	2.45	SO
Figure 8b	10	1	25%	32%	2.55	SO
Figure 9	14	3	75%	41%	2.18	SA& SO
Figure 10	24	4	200%	115%	11.22	SO

11 Conclusions

CMOS C-element designs for testability have been presented. Tests for detection of all single line stuck-at and transistor stuck-open faults of the proposed testable implementations of the symmetric and asymmetric C-elements have been derived. It has been shown that although the CMOS C-element testable for line stuck-at faults allows the detection of all its single stuck-open faults its implementation requires more overhead than that of the C-element testable only for stuck-open faults. The fundamental problem of testing feedback transistors in static C-elements has been resolved. We illustrated that the sequential function of the symmetric C-element can be transformed into a combinational one (AND or OR) by driving its feedback transistors with an additional input. The asymmetric C-elements can be transformed into repeaters of one of their inputs, OR or AND gates. A scan test can be implemented by using the scan testable C-element design described in the report. An analysis of the costs required to implement different testable C-element structures has been presented. It shows that the largest number of transistors and area overhead are required to implement the scan testable C-element. The CMOS structures of the C-element considered in the report can be used to design asynchronous VLSI circuits for testability.

Acknowledgments

We would like to express our gratitude to Jim Garside and David Jackson for their useful comments and discussions on the topic of this report. Special thanks to Craig Farnsworth for his invaluable help and consultation with *Cadence* CAD tools.

References

[1] S. Hauck, “Asynchronous design methodologies: An overview,” *Proc. IEEE*, Vol. 83, No. 1, Jan. 1995, pp. 69-93.

[2] L. Lavagno, A. Sangiovanni-Vincentelli, “Algorithms for synthesis and testing of asynchronous circuits,” Kluwer Academic Publishers, 1993.

[3] S. B. Furber, P. Day, J. D. Garside, N. C. Paver, J. V. Woods, “AMULET1: A micropipelined ARM,” *Proc. IEEE Comput. Conf.*, March 1994.

[4] H. Hulgaard, S. M. Burns, G. Borriello, “Testing asynchronous circuits: A survey,” *TR-FR-35*, Department of Computer Science, University of Washington, Seattle, WA, USA, 1994.

[5] P. Hazewindus, “Testing delay-insensitive circuits,” Ph.D. thesis, *Caltech-CS-TR-92-14*, California Institute of Technology, 1992.

[6] P. A. Beerel, T. H. Y. Meng, “Semi-modularity and testability of speed-independent circuits,” *Integration, The VLSI Journal*, 13, 1992, pp. 301-322.

[7] I. David, R. Ginosar, M. Yoeli, “Self-timed is self-diagnostic,” *TR-UT-84112*, Department of Computer Science, University of Utah, Salt Lake City, UT, USA, 1990.

[8] J. A. Brzozowski, K. Raahemifar, “Testing C-elements is not elementary,” *Proc. 2nd Working Conf. on Asynchronous Design Methodologies*, South Bank University, May 30-31 1995, pp. 150-159.

[9] N. H. E. Weste, K. Eshraghian, “Principles of CMOS VLSI design: A systems perspective,” Addison-Wesley Publishing Co., 1993.

[10] R. L. Wadsack, “Fault modelling and logic simulation of CMOS and MOS circuits,” *Bell System Tech. Journal*, vol. 57, May-June 1978, pp. 1449-1474.

[11] M. K. Reddy, S. M. Reddy, “Detecting FET stuck-open faults in CMOS latches and flipflops,” *IEEE Design & Test of Computers*, vol. 3, no. 5, Oct. 1986, pp. 17-26.

[12] C. Farnsworth, D. A. Edwards, Jianwei Liu, S. S. Sikand, “A hybrid asynchronous system design environment,” *Proc. 2nd Working Conf. on Asynchronous Design Methodologies*, South Bank University, May 30-31 1995, pp. 91-98.

[13] G. Russell, I. L. Sayers, “Advanced simulation and test methodologies for VLSI design,” Van Nostrand Reinhold (International), 1989.

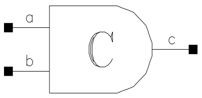
[14] M. Roncken, “Partial scan test for asynchronous circuits illustrated on a DCC error corrector,” in *Proc. Int. Symposium on Advanced Research in Asynchronous Circuits and Systems* (*Async94*), Nov. 1994, pp. 247-256.

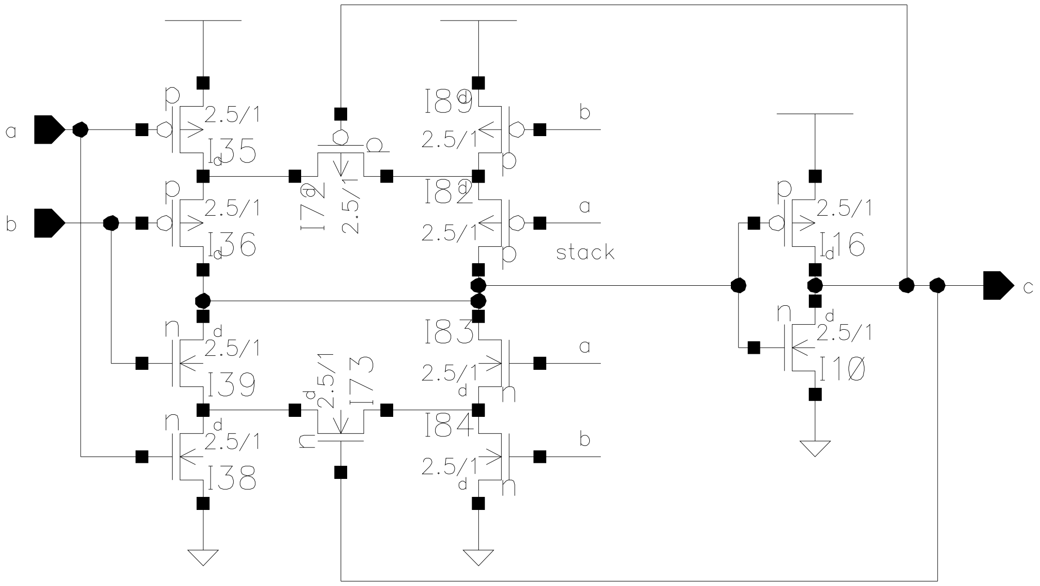
[15] A. Khoche, E. Brunvand, “Testing micropipelines,” *Proc. Int. Symposium on Advanced Research in Asynchronous Circuits and Systems* (*Async94*), Utah, Nov. 1994, pp. 239-246.

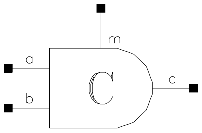
[16] O.A.Petlin, S.Furber, “Scan testing of micropipelines”, *Proc. 13th IEEE VLSI Test Symposium*, Princeton, New Jersey, USA, May 1995, pp. 296-301.

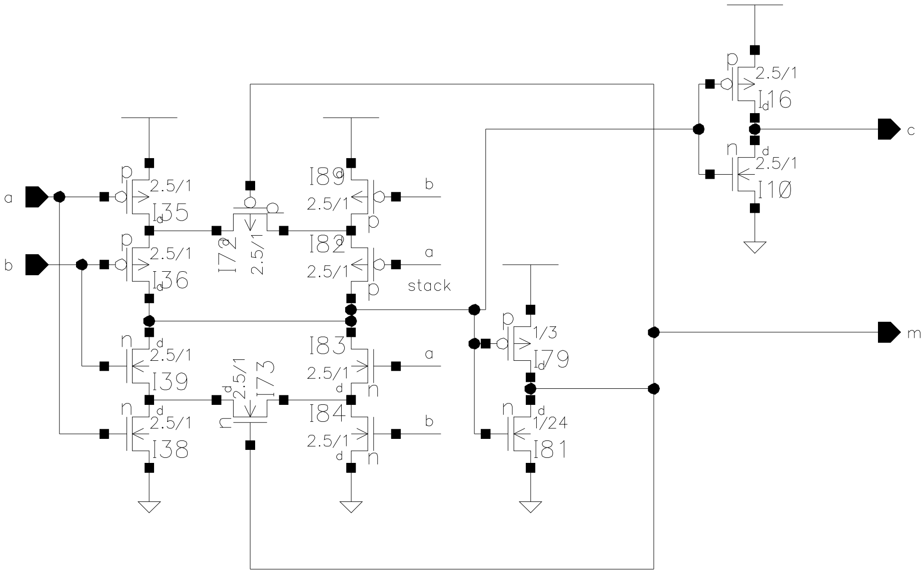
[17] I. E. Sutherland, “Micropipelines,” *Communications of the ACM*, Vol.32, no.6, June 1989, pp. 720-738.

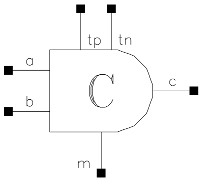
Appendix: C-element cells

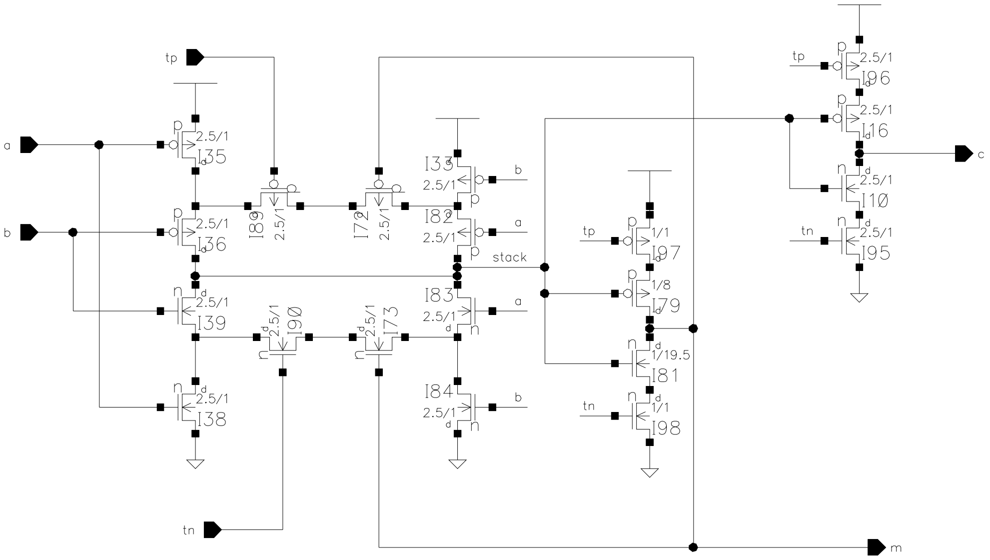

a
b
C
c

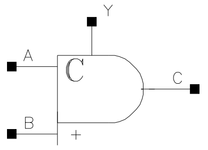

a
b
p
2.5/1
I35
d
p
2.5/1
I36
d
n
d
2.5/1
I39
n
d
2.5/1
I38
I72
2.5/1
p
d
I73
2.5/1
n
I89
2.5/1
p
b
I82
2.5/1
p
a
stack
I83
2.5/1
d
n
a
I84
2.5/1
d
n
b
p
2.5/1
I16
d
n
d
2.5/1
I10
c

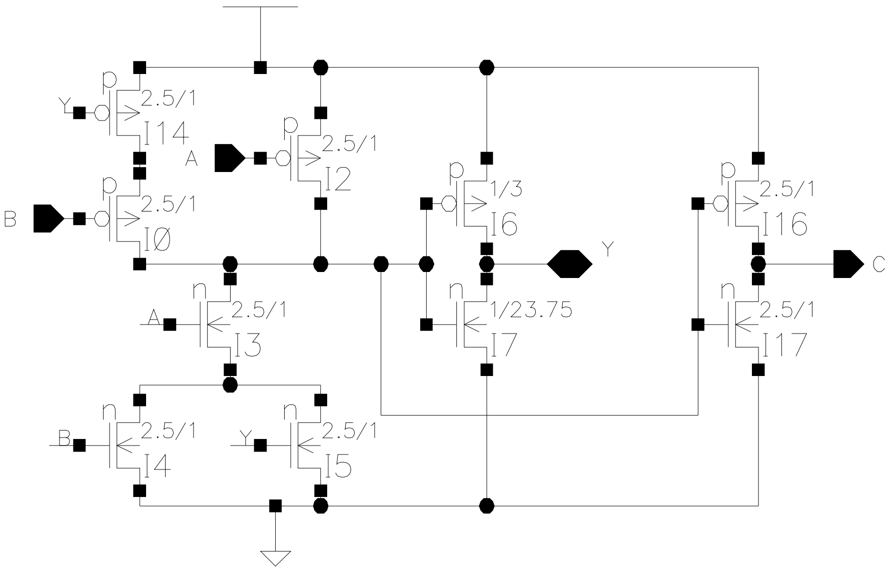

m
a
b
C
c

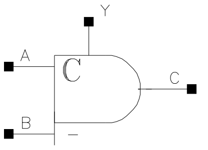

a
b
p
2.5/1
I35
I36
n
I39
I38
I72
2.5/1
I73
2.5/1
I89
2.5/1
b
I82
2.5/1
a
stack
I83
2.5/1
a
I84
2.5/1
b
p
1/3
I79
n
1/24
I81
p
2.5/1
I16
n
2.5/1
I10
c
m

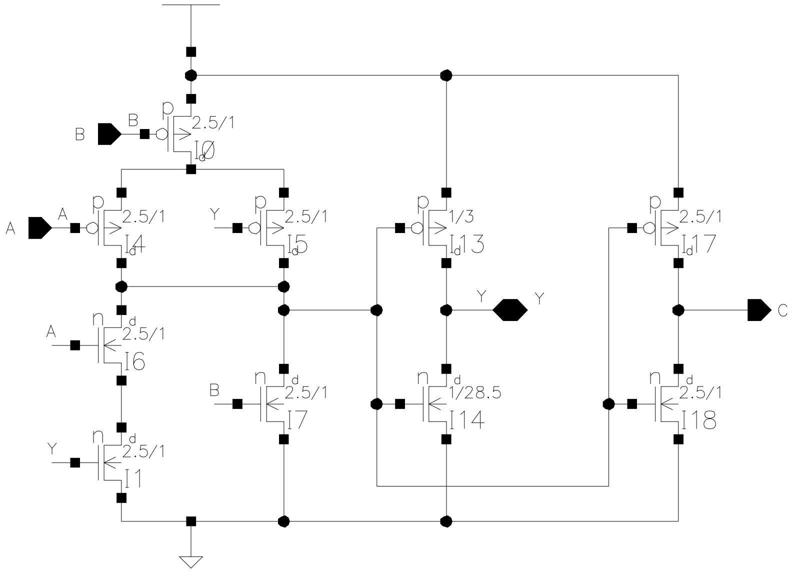

tp
tn
a
b
C
c
m

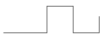

tp
a
b
tn
m
c
stack
I35
I36
I39
I38
I89
I90
I72
I73
I33
I82
I83
I84
I97
I79
I81
I98
I96
I16
I10
I95
2.5/1
1/1
1/8
1/19.5
p
n
d

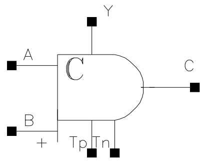

Y
A
C
C
B
+

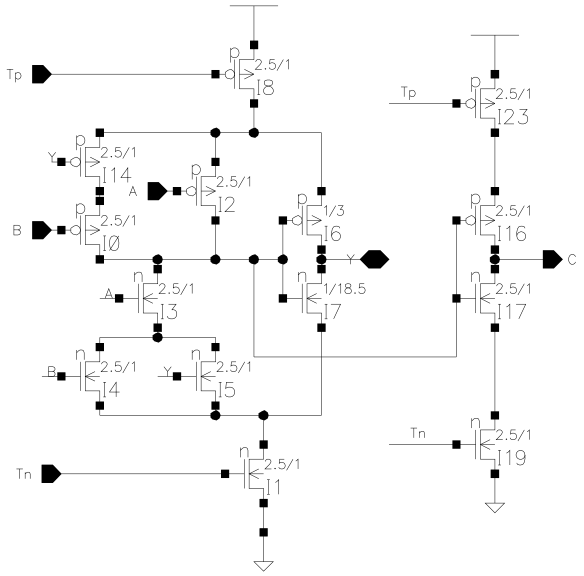

p 2.5/1 I14
Y
p 2.5/1 I2
A
B
p 2.5/1 I0
p 1/3 I6
Y
n 1/23.75 I7
p 2.5/1 I16
C
n 2.5/1 I17
n 2.5/1 I3
A
n 2.5/1 I4
B
n 2.5/1 I5
Y

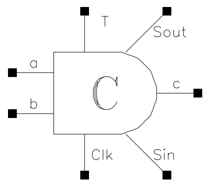

Y
A
C
C
B
−

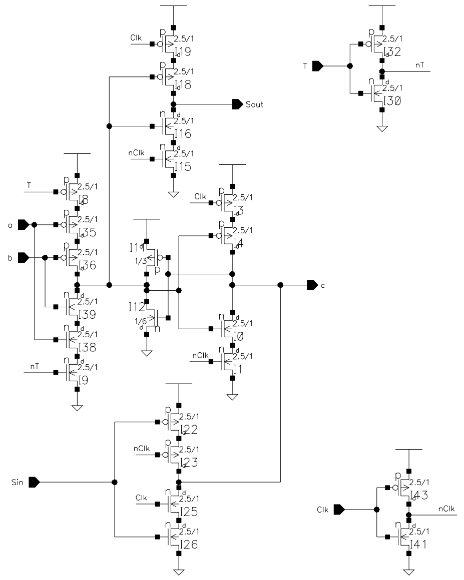

B
B
p
2.5/1
I0
A
A
p
2.5/1
I4
Y
p
2.5/1
I5
p
1/3
I13
p
2.5/1
I17
Y
Y
C
A
n
2.5/1
I6
B
n
2.5/1
I7
n
1/28.5
I14
n
2.5/1
I18
Y
n
2.5/1
I1




Y
A
C
C
B
+
Tp
Tn


Tp
p
2.5/1
I8
Tp
p
2.5/1
I23
Y
p
2.5/1
I14
A
p
2.5/1
I2
p
1/3
I6
B
p
2.5/1
I0
Y
p
2.5/1
I16
C
A
n
2.5/1
I3
n
1/18.5
I7
n
2.5/1
I17
B
n
2.5/1
I4
Y
n
2.5/1
I5
Tn
n
2.5/1
I19
Tn
n
2.5/1
I1


T
Sout
a
C
c
b
Clk
Sin


Clk
I19
I18
Sout
I16
nClk
I15
T
I8
a
I35
b
I36
I39
I38
nT
I9
I11
1/3
I12
1/6
Clk
I3
I4
c
I0
nClk
I1
T
I32
nT
I30
I22
nClk
I23
Sin
Clk
I25
I26
Clk
I43
nClk
I41
2.5/1