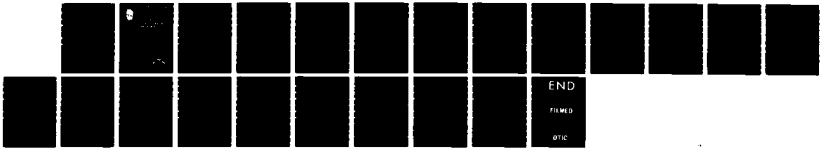
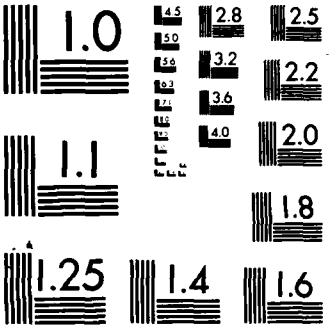
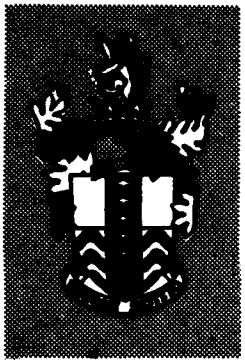
AD-A159 557 MODELLING TRANSMISSION GATES IN ELLA(U) ROYAL SIGNALS 1/1
AND RADAR ESTABLISHMENT MALVERN (ENGLAND)
K R MILNER ET AL. MAY 85 RSRE-MEMO-3825 DRIC-BR-96524
UNCLASSIFIED F/G 9/3 NL





(3)



RSRE
MEMORANDUM No. 3825

ROYAL SIGNALS & RADAR ESTABLISHMENT

MODELLING TRANSMISSION GATES IN ELLA

Authors: K R Milner
and T L Thorp

PROCUREMENT EXECUTIVE,
MINISTRY OF DEFENCE,
RSRE MALVERN,
WORCS.

RSRE MEMORANDUM No. 3825

UNLIMITED

ROYAL SIGNALS AND RADAR ESTABLISHMENT

Memorandum 3825

TITLE: MODELLING TRANSMISSION GATES IN ELLA

AUTHORS: K R Milner and T L Thorp

DATE: May 1985

SUMMARY

This Memorandum describes an algorithm for modelling transmission gates and gives details of its implementation in the Hardware Design and Description Language ELLA.

RSRE MEMORANDUM 3825

MODELLING TRANSMISSION GATES IN ELLA

K R Milner and T L Thorp

LIST OF CONTENTS

1 INTRODUCTION

The problem of modelling circuits in ELLA at the transistor level can be tackled either using unidirectional or bidirectional elements, depending on the level of sophistication required. Taking the latch circuit of Figure 1 as an example, if it is known that information flows through tg1 and tg3 from left to right and tg2 from right to left, then the circuit can be modelled using unidirectional elements. However, information could flow in the opposite directions - possibly unintentionally - and it would be more accurate to use a bidirectional model. A bidirectional model and a less sophisticated unidirectional model are both considered below.

In both cases, a method of giving a strength to each signal, as well as a value, is necessary for two main reasons:

i. Some circuits require drivers of different strengths. For example, in the latch circuit of Figure 2, n3 is a weaker inverter than n1, so that when tg1 is open, n1 overpowers n3, and consequently any information stored in the n2/n3 loop is replaced by the signal from n1.

ii. Modelling dynamic logic requires at least two strengths. This is because each node must "remember" its previous value and under certain conditions use it to calculate the next value of the node (this is discussed in more detail in section 3.1).

The remembered signal must be weaker than any current signal, and this will be achieved if the lowest value strengths are used only for the remembered signals.

Limitations of the current modelling algorithm are:

i. When the control input of a transmission gate is set by one of its own inputs, some states can become "locked in" incorrectly; this is particularly noticeable when trying to initialise circuits. For example, in the circuit of Figure 3, a weak unknown initially entering n1 produces a strong unknown on the control input of tg1, which in turn can produce a strong unknown on the control inputs of tg2 and tg3. The loop consisting of n1, tg1 and tg3 will thus be in a strong unknown state, and since this strong unknown is feeding tg1, which in turn feeds tg2 and tg3, the strong unknown becomes permanently locked in.

ii. Circuits with loops consisting entirely of transmission gates are simulated incorrectly, in the sense that the strength of a signal at a particular node can be wrong, even though the value is correct. For example, in the circuit of Figure 4, if TG1 is non-conducting, and TG2 and TG3 are conducting, then the signal in the loop should be weak, because it is undriven. In fact, in the algorithm below, the signal will incorrectly keep its initial strength.

2 SIGNALS

The basic signal used (TYPE sig) is slightly more sophisticated than the usual three or four value logic (t, f and x or t, f, x and z). It is related to the signals used by Flake et al (20th Design Automation Conference, Miami 1983, pp 615-8).

For example if two strengths are used with:

H corresponding to strong true

T corresponding to weak true

F corresponding to weak false

L corresponding to strong false

then the possible signals are given in Figure 5.

In the above example, H, T, F and L have ELLA representations (of TYPE sig) [2](s/2, tr) [2](s/1, tr), [2](s/1, fa) and [2](s/2, fa) respectively, where tr and fa are boolean values of TYPE bool2. The duplication of ordered pairs also allows all possible unknowns to be represented, for example, HT and FL can be represented by ((s/2, tr), (s/1, tr)) and ((s/1, fa), (s/2, fa)). These signs can readily be converted to three valued logic - if both boolean values are tr then the equivalent value is t, if both are fa then the equivalent value is f, otherwise it is x.

A high impedance state is not included in the above because it is not considered to be a signal in the same way as H, T, HF etc. It is generated when the control input of a transmission gate is false, and is used by the algorithm to decide when the previous value at a node should be used to calculate the new value. The high impedance state is one value of a TYPE data, and is introduced by the declaration:

```
TYPE data  =  NEW (info & sig |hi & sig| z)
```

The TYPE sig associated with the new values info and hi corresponds to the signals described previously. z is the high impedance state and hi & sig is an unknown which could be either z or info & sig. For example, hi & [2](s/5, tr) is interpreted as either z or info & [2](s/5, tr).

3 BIDIRECTIONAL MODEL

3.1 ALGORITHM

The bidirectional transmission gate (BTGATE) is written as two unidirectional gates (TGATEs) back to back, as shown in Figure 6. The ELLA FNTYPE mechanism is used to form a pair of wires with information flow in opposite directions.

For a unidirectional gate, if the control input is true, then the data input is passed through to the output. If it is false, then the high impedance signal, z, is produced; otherwise, the signal hi & ip is delivered, which is interpreted as "either z or info & ip".

When a number of signals meet at a node, possible conflicts are decided by the bus FNs, essentially by taking the "strongest" incoming signal. The detailed rules for evaluating "strongest" are:

i. If all inputs except one are undriven, then the bus will take up the value of the driven wire.

ii. If there are two driven inputs of different strengths, then the bus will take up the value of the stronger signal. For example, if the inputs are [2](s/3, tr) and [2](s/2, fa), then the bus will take up the value [2](s/3, tr).

iii. If there are two driven signals of equal strength, then the bus will take up a value which covers both possibilities. For example, if the incoming signals are [2](s/3, tr) and [2](s/3, fa), then the bus will take up the value (s/2, tr), (s/3, fa)).

iv. If the signals are of mixed strength, then the bus takes up a value which combines the stronger of the incoming signals. For example, if the incoming signals are ((s/3, tr), (s/2, fa)) and ((s/2, tr), (s/4, fa)) then the bus will take up the value ((s/3, tr), (s/4, fa)). SEE NOTE OVERLEAF.

v. If all the inputs are undriven, then the bus will take up the value z (high impedance).

NOTE: Mixed strength signals will not occur at this point in the algorithm (unless provided initially as inputs), so rule (iv) is usually redundant in practice. Sigs of the form ((s/3, tr), (s/3, fa)) are used instead of the simpler form (s/3, (tr, fa)) because of their greater flexibility.

Each bidirectional transmission gate connected to a bus attempts to "write" to the bus. It would seem natural for the value that the bus takes up to be fed back to each bidirectional transmission gate connected to the bus, but this feedback can create unintentional latch circuits. For example, in Figure 7, if tg1 and tg3 are non-conducting, and tg2 conducting, then the signal entering bus2 along 3 will leave along 4, enter bus1 along 2 and leave via 1, so creating an undesirable loop consisting of 1, 2, 3 and 4, in which a signal has become locked in. In this particular case, the problem can be overcome by allowing the incoming signals to the bus to pass straight through to the output, as shown in Figure 8. This can be generalised to a bus with any number of inputs, as shown by Monahan and Sangster (20th Design Automation Conference, Miami 1983, pp 94-99): in general, the signal fed back to one particular component connected to the bus is the "strongest" signal from all components except the one in question. The internal connections of a bus with four inputs is shown in Figure 9; note that each outgoing wire is paired with its corresponding incoming wire, using the ELLA FNTYPE mechanism, as shown in Figure 10.

The signals referred to so far are values of TYPE data, which may include high impedance. At various points in the circuit, such as the input of logic gates, all such values are converted to the form info & sig, which is easily interpreted in three-valued logic. Note that there is no such conversion inside the bus, so that nowhere inside the bus is its value explicitly calculated. There are three rules for this conversion, depending on the value of the TYPE data:

i. info & sig - the input value is the converted value.

ii. z - the incoming wire acts like a capacitor which stores the last value. The weakest strength corresponding to the last value is the converted value.

iii. hi & sig - this is a combination of (i) and (ii), since hi & sig is interpreted as either z or info & sig, so the sig associated with hi must be "extended" to include all possible signals covered by the weakened form of the last value.

To understand the concept of "extending" a signal, it is useful to consider pairs consisting of a strength and a boolean value to be ordered in the following way:

.
.
(s/5, tr) greatest
.
.
(s/1, tr)
(s/1, fa)
.
.
(s/5, fa) least
.
.

Now consider hi & ((s/1, fa), (s/3, fa)) being extended to include a last value of info & ((s/2, tr), (s/2, fa)). First, the last value is converted to its weakest form, in this case info & ((s/1, tr), (s/1, fa)).

Then, the first ordered pair of the answer is calculated; it is the greater of:

i. the first ordered pair associated with hi - (s/1, fa) in the above example; and

ii. the first ordered pair associated with the weakest form of the last value - (s/1, tr) in the example.

So, the first ordered pair of the answer in the above example would be (s/1, tr). The second ordered pair of the answer is calculated in a similar way; it is the lesser of:

i. the second ordered pair associated with hi - (s/3, fa) in the above example; and

ii. the second ordered pair associated with the weakest form of the last value - (s/1, fa) in the example. The second ordered pair of the answer in the above example is (s/3, fa).

The calculation is completed by making the answer an associated type of the form info & sig - info & ((s/1, tr), (s/3, fa)) in the above example.

3.2 PRIMITIVES

Note that a void output has not been implemented yet in ELLA, so a TYPE void has been declared:

```
TYPE void = (dummy|dummy1) .
```

3.2.1 Strengthmax

Specification: INT strengthmax = ...

Strengthmax must be set by the user, who can decide on the number of different strengths needed for a particular application. A minimum of two is required for dynamic logic to be implemented; the upper limit is restricted only by the ELLA integer range. For example, the circuit in Figure 2 needs at least three strengths; one for dynamic logic, and the other two for the different strength drivers. The latch circuit of Figure 1 can be simulated using two strengths.

3.2.2 BNOT

Specification: MAC BNOT{INT s} = ((data→data): ip1 ip2)→data:

Although the basic inverter is essentially unidirectional, it is useful to have such an element written so that it will plug directly into bidirectional elements. The convention used is that the input part of ip1 is used to calculate an answer, which is fed to the output part of ip2. The input part of ip2 is not used and the output part of ip1 is connected to high impedance and therefore does not affect the value of the bus connected to ip1.

An integer must be specified (as a MACro parameter), which will give a measure of the strength of the inverter. It clearly cannot exceed the maximum strength allowed (strengthmax) and must be greater than 1, since the lowest strength is used only for storing the last value.

3.2.3 BNOR

```
Specification:  MAC BNOR{INT s} = ((data→data):
                ip1 ip2 ip3)→data:
```

This has a very similar specification to BNOT, so the same remarks apply; in this case, though, the input parts of the first two FNTYPE inputs are used to calculate an answer which is fed to the output part of ip3, and the input part of ip3 is always connection to a high impedance value.

3.2.4 BNAND

```
Specification:  MAC BNAND{INT s} = ((data→data):
                ip1 ip2 ip3→data:
```

Similar to BNOR.

3.2.5 BTGATE

```
Specification:  FN BTGATE = ((data→data):
                ck ip1 ip2)→[2]data:
```

The convention used regarding the inputs is that ck is the control input, ip1 and ip2 are equivalent data inputs.

3.2.6 UBTGATE

```
Specification:  FN UBTGATE = ((data→data): ck ip1 ip2)→data:
```

UBTGATE is a unidirectional transmission gate written with the same specification as the bidirectional gate (BTGATE). This is so that a unidirectional gate can be plugged directly into a bidirectional circuit. The convention used is that the input part of ck is the control, the input part of ip1 is the data input and the input part of ip2 is a dummy input that takes no part in the calculation of the output.

3.2.7 Bus FNs

```
Specifications:  FN BUS2  =  FNSET [2](data)→data:
                 FN BUS3  =  FNSET [3](data)→data:
                 FN BUS4  =  FNSET [4](data)→data:
                 FN BUS5  =  FNSET [5](data)→data:
```

These arbitrate between 2, 3, 4 and 5 signals meeting at a point, as described in section 3.1. There is a MAC bus available, so that a user can create a bus with any number of inputs (see program documentation).

From the specifications given already, it can be seen that the logic elements BNOT, BNOR and BNAND and the transmission gate BTGATE require input parameters of the form (data→data); these are usually supplied by a BUS2, BUS3 etc as illustrated in the examples below.

3.2.8 BUFFER

```
Specification:  FN BUFFER = FNSET ((bool3)→void,
                                   (data) → data):
```

BUFFER converts a unidirectional input (either t, f or x) into a form that can be used as an input to a bidirectional circuit element, for example, BNOT or BTGATE. For an example of its use see LATCH1 below.

3.2.9 BUSEXT

```
Specification:  FN BUSEXT =  ([2](data→data))→void:
```

It is sometimes useful to be able to join two bus FNs together. This can be achieved by using an intermediate BUSEXT. For an example which illustrates the use of BUSEXT, see FN LATCH below.

3.2.10 VLINE

Specification: MAC VLINE{INT i,v} = (data) → data:

This is used when it is necessary to connect a bidirectional component either to a logical true or logical false. The first integer represents the strength of the output signal and cannot exceed strengthmax, the second must be either 0 or 1; 0 represents a logical false, 1 represents a logical true.

3.2.11 DATABOOL3

Specification: FN DATABOOL3 = (data) → bool3:

This performs a straightforward conversion from a TYPE data to a TYPE bool3; see LATCH1 below for an illustrative example.

3.2.12 BOOL3DATA

Specification: FN BOOL3DATA = (bool3) → data:

This performs a straightforward conversion from a TYPE bool3 to a TYPE data; see LATCH1 below for an illustrative example.

3.3 EXAMPLES

The ELLA description for the latch circuit of Figure 11 is as follows:

```
FN LATCH = ((data → data): ip1 ip2 ip3 ip4) → void:

(MAKE BTGATE:   tg1 tg2
      BNOT{5}:  n1 n2
      BUS3:     tg1tg2n1 tg2n2ip4
      BUS2:     n1n2
      BUSEXT:   ext

 JOIN (IOip2,IOip1,IOtg1tg2n1[1]) → tg1
      (IOtg1tg2n1[3],IO1n2[1]) → n1
      (IOip4,IOtg2n2ip4[3]) → ext
      (IOn1n2[2],IOtg2n2ip4[2]) → n2
      (IOip3,IOth2n2ip4[1],IOtg1tg2n1[2]) → tg2

 OUTPUT dummy).
```

This representation is illustrated in Figure 12.

Two such latches can be connected in series as shown in Figure 13, with ELLA description:

```
FN TWOLATCHES  =  ((data→data): ip1 ip2 ip3 ip4 ip5 ip6)→void:
(MAKE LATCH: lch1 lch2,
       BUS2: lch1lch2.
 JOIN (IOip1,IOip2,IOip3,IOlch1lch2[1])→lch1,
      (IOlch1lch2[2],IOip4,IOip5,IOip6)→lch2.
 OUTPUT dummy).
```

In turn, TWOLATCHES can be used as a building block.

3.4 SIMULATION

The outputs of the MACs BNOT, BNOR and BNAND can be monitored in the usual way. BTGATE and UBTGATE can be similarly monitored; BTGATE has two outputs: the first is calculated using ip2 (and is the outgoing wire paired with ip1), the second is calculated using ip1 (and is paired with ip2).

There is also a name "value" inside each bidirectional bus FN, which holds the "strongest" of all the inputs to the bus. This may include a high impedance value and therefore cannot be regarded strictly as the value that the bus takes up, but it can be useful for monitoring purposes. For examples which illustrate this, see CMOSNOT, CMOSNOR and CMOSNAND in the programmed examples.

The input TYPEs of LATCH and TWOLATCHES are fairly complex, so the simulator commands are lengthy. For example, a typical input command to LATCH would be:

```
pm→info s/5 tr s/5 tr→info s/5 tr s/5 tr
  →info s/5 tr s/5 tr→info s/5 tr s/5 tr
```

Under these circumstances, these input TYPEs can be avoided. For example, with the circuit of Figure 11, when the inputs to a particular FN consist entirely of signals of maximum strength, then BUFFER can be used as follows:

```
FN LATCH1  =  (bool3: ip1 ip2 ip3 ip4)→[3]bool3:
(MAKE   BTGATE: tg1 tg2,
        BNOT{5}: n1 n2,
        BUS3: tg1tg2n1 tg2n2ip4,
        BUS2: n1n2,
        BUSEXT: ext,
        BUFFER: ip1b ip2b ip3b ip4b.
 JOIN   ip1→ip1b[1],
        ip2→ip2b[1],
        (IOip2b[2],IOip1b[2],IOtg1tg2n1[1])→tg1,
        (IOtg1tg2n1[3],IOn1n2[1])→n1,
        ip4→ip4b[1],
        (IOip4b[2],IOtg2n2ip4[3])→ext,
        (IOn1n2[2],IOtg2n2ip4[2])→n2,
        ip3→ip3b[1],
        (IOip3b[2],IOtg2n2ip4[1],IOtg1tg2n1[2])→tg2
 OUTPUT ([INT k=1..3] DATABOOL3 tg1tg2n1[k])).
```

Now both inputs and outputs are bool3 TYPEs, so LATCH1 can be simulated more readily than LATCH. For example, the equivalent simulator command to the one above is:

```
pm t t t t
```

(assuming that the maximum strength used in this particular example is 5).

Note that BUFFER converts a TYPE bool3 to a sig of the same value, but of the maximum strength, so that a very restricted range of input values can be used with LATCH1 and its use is therefore limited.

4 UNIDIRECTIONAL MODEL

4.1 ALGORITHM

An algorithm closely related to the bidirectional one, which uses similar TYPEs and a similar transmission gate model, can be used to describe circuits in a unidirectional way. Clearly, there will be no FNTYPEs or FNSETs, but the signal strengths and unknowns can be represented as before.

4.2 PRIMITIVES

4.2.1 NOT

Specification: MACNOT{INT s} = (data)→data:

The basic inverter, with an INT parameter which specifies the strength of the output signal (as in the bidirectional case).

4.2.2 NOR

Specification: MAC NOR{INT s} = ([2]data)→data:

The basic nor, with INT parameter specifying the strength of the output signal.

4.2.3 NAND

Specification: MAC NAND{INT s} = ([2]data)→data:

The basic nand, with INT parameter specifying the strength of the output signal.

4.2.4 TGATE

Specification: TGATE = (data: ck ip)→data:

The unidirectional transmission gate with the first input corresponding to the control input and the second corresponding to the data input.

4.2.5 UBUS

Specification: MAC UBUS{INT n} = ([n]data)→data:

This is used when two or more transmission gates meet at a point. For an example which uses UBUS, see FN LATCH2 below.

4.2.6 BOOL3DATA

Specification: FN BOOL3DATA = (bool3)→data:

This is used to convert a TYPE bool3 to a TYPE data, and can be used for simulation in the same way as BUFFER in the bidirectional case.

4.2.7 BTOU

Specification: FN BTOU = FNSET (((data→data))→void,
(void)→data):

BTOU converts a TYPE (data→data), which would plug directly into a bidirectional bus, into a data which is suitable as an input to a unidirectional element.

4.2.8 UTOB

Specification: FN UTOB = FNSET ((data: ip1)→void,
(data→data): ip2)→void):

UTOB converts a unidirectional input, of TYPE data, into a TYPE (data→data) which can be used as an input to BNOT, BNOR, BNAND and BTGATE.

4.3 EXAMPLE

The ELLA description of the latch circuit of Figure 11, using the unidirectional model, is given below:

```
FN LATCH2  =  (data: ip1 ip2 ip3)→data:
(MAKE   TGATE: tg1 tg2,
        NOT{5}: n1 n2,
        UBUS{2}: tg1tg2.
 JOIN   (ip2,ip1)→tg1,
        tg1tg2→n1,
        n1→n2,
        (ip3,n2)→tg2.
 OUTPUT n2).
```

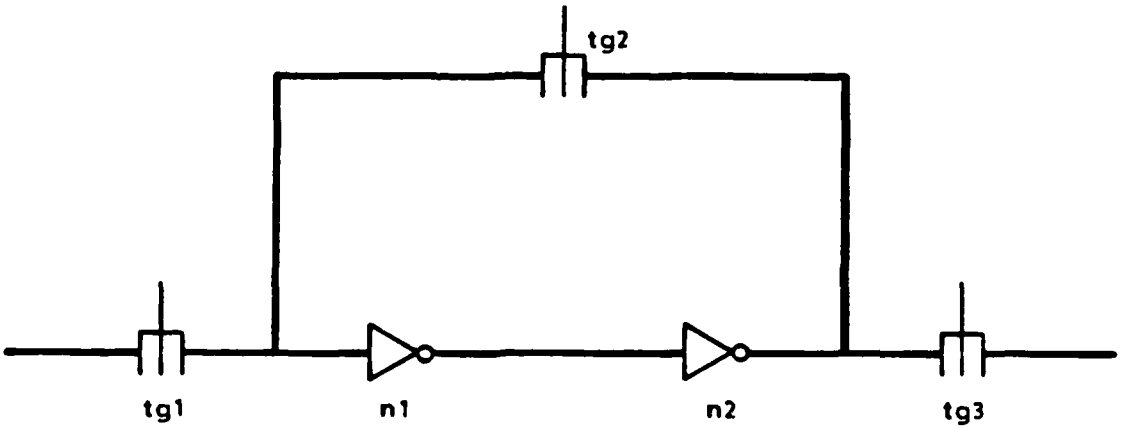


Figure 1

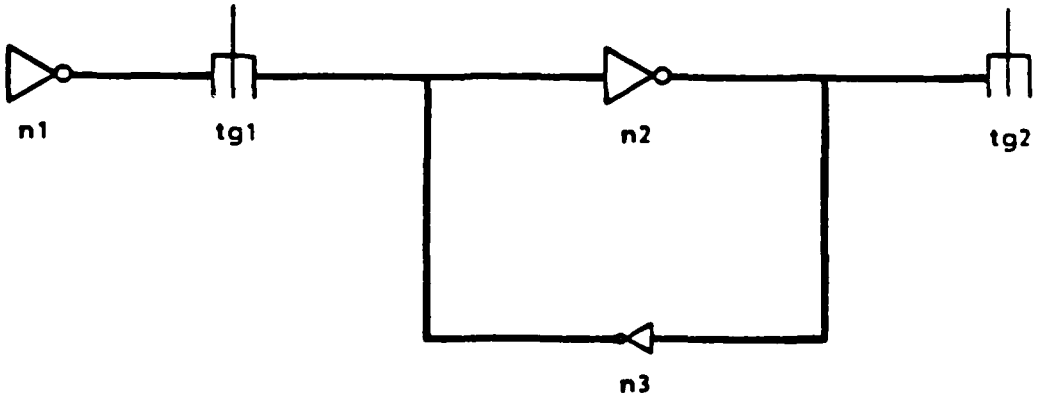


Figure 2

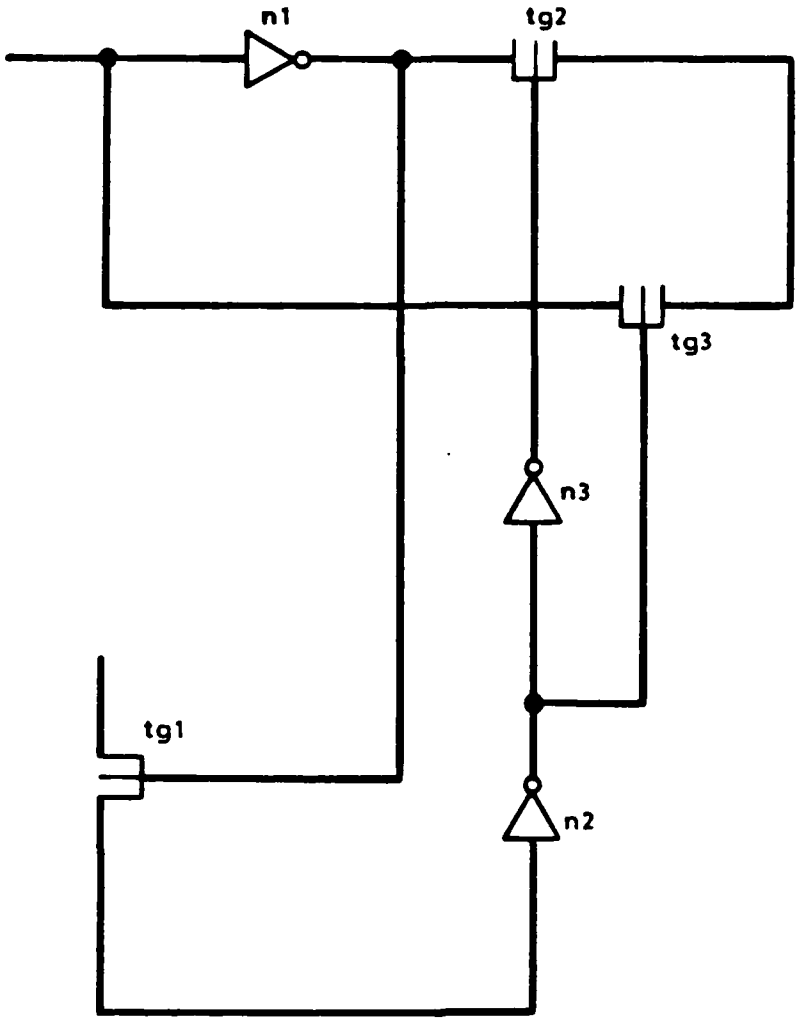


Figure 3

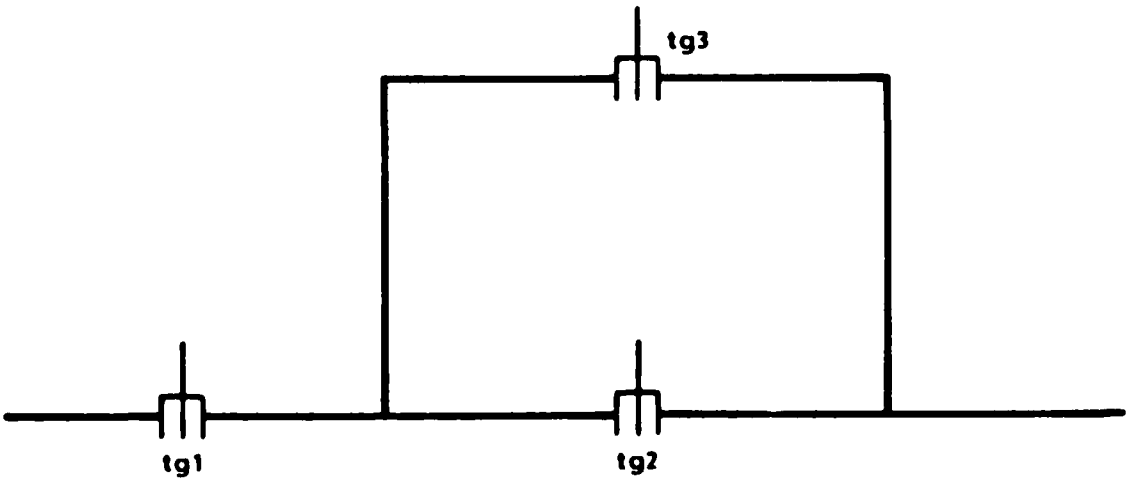


Figure 4

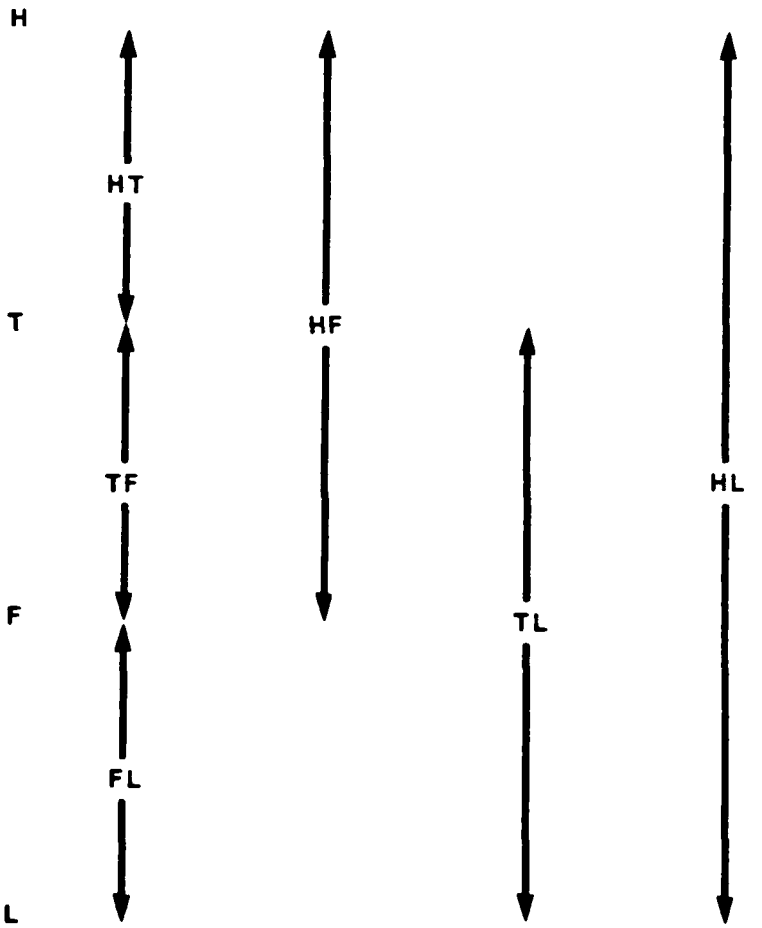


Figure 5

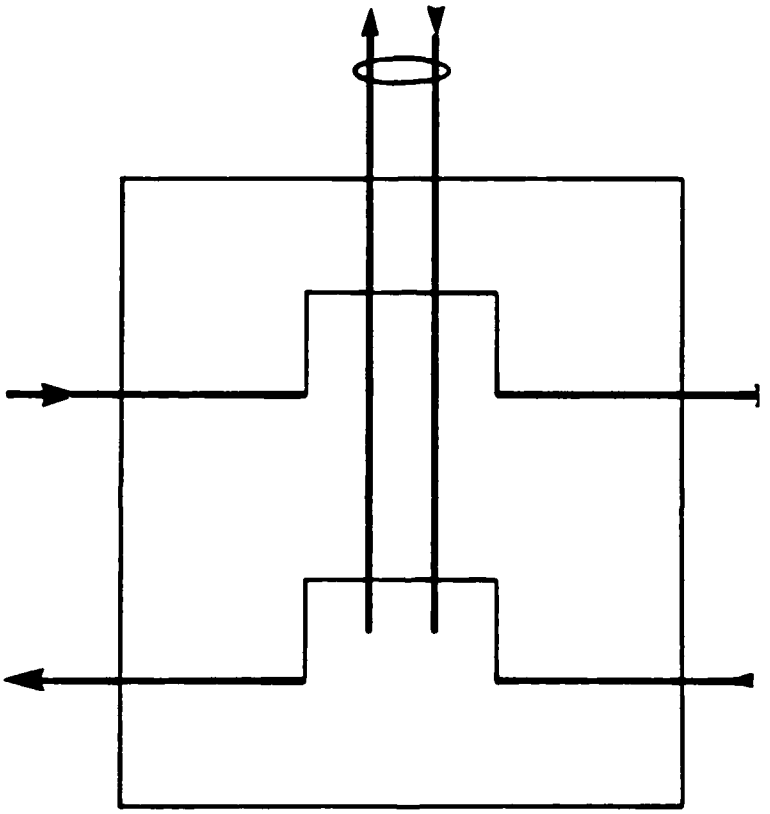


Figure 6

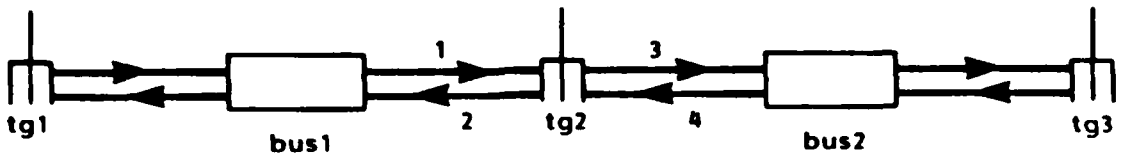


Figure 7

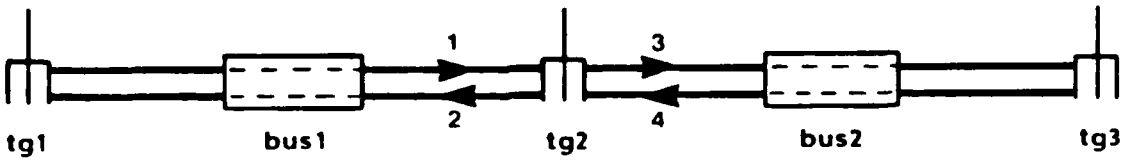


Figure 8

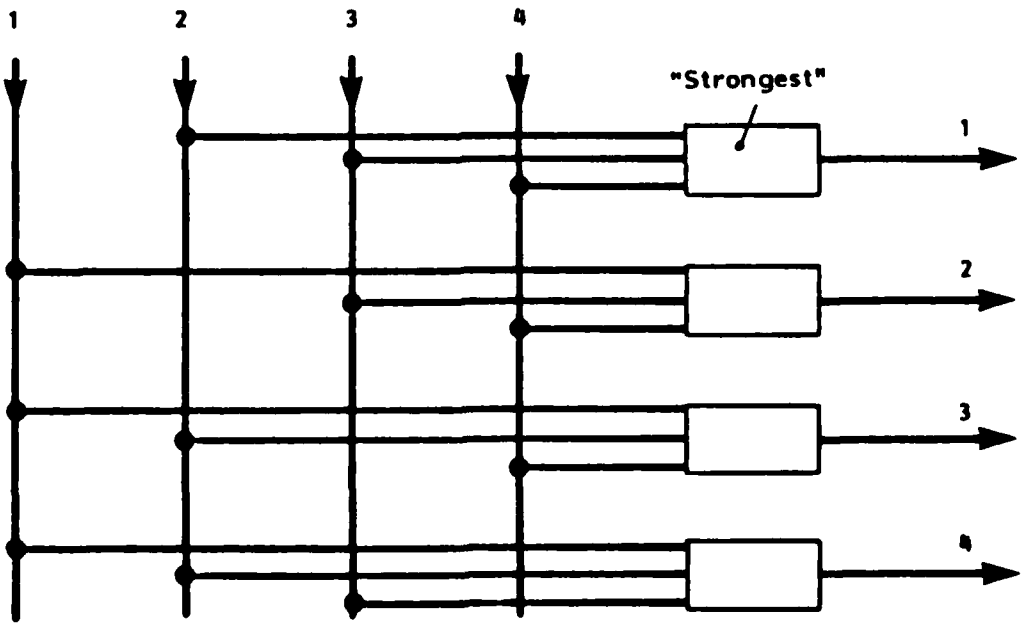


Figure 9

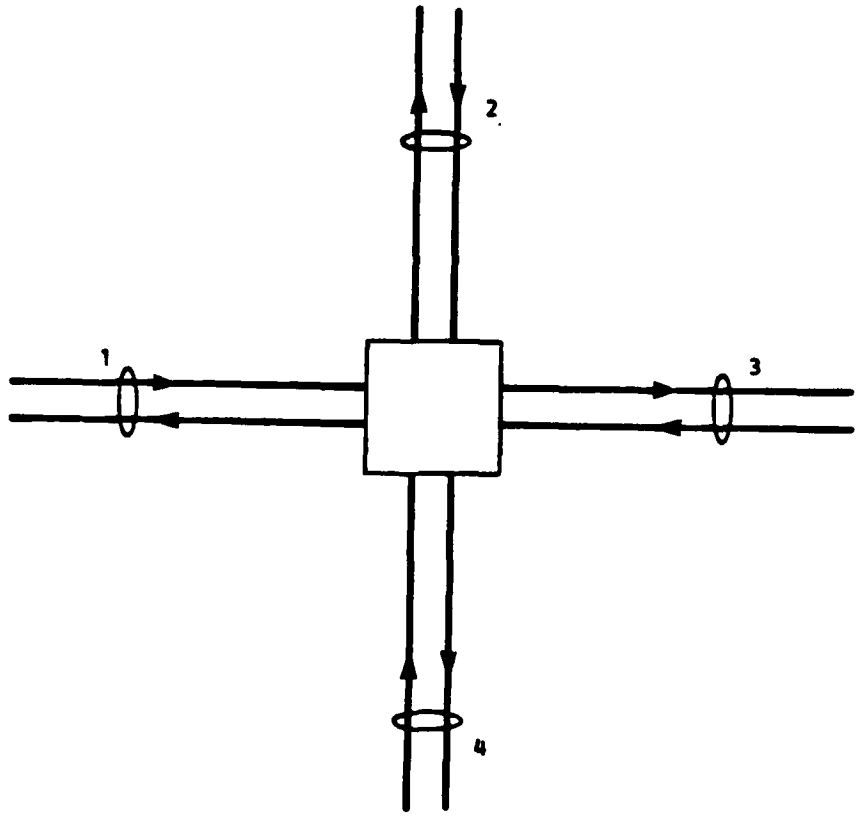


Figure 10

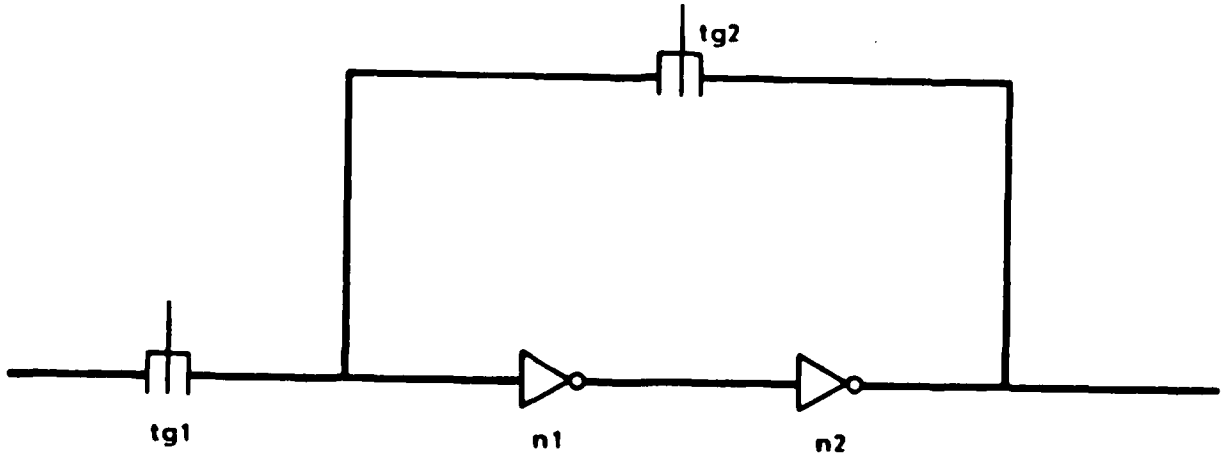


Figure 11

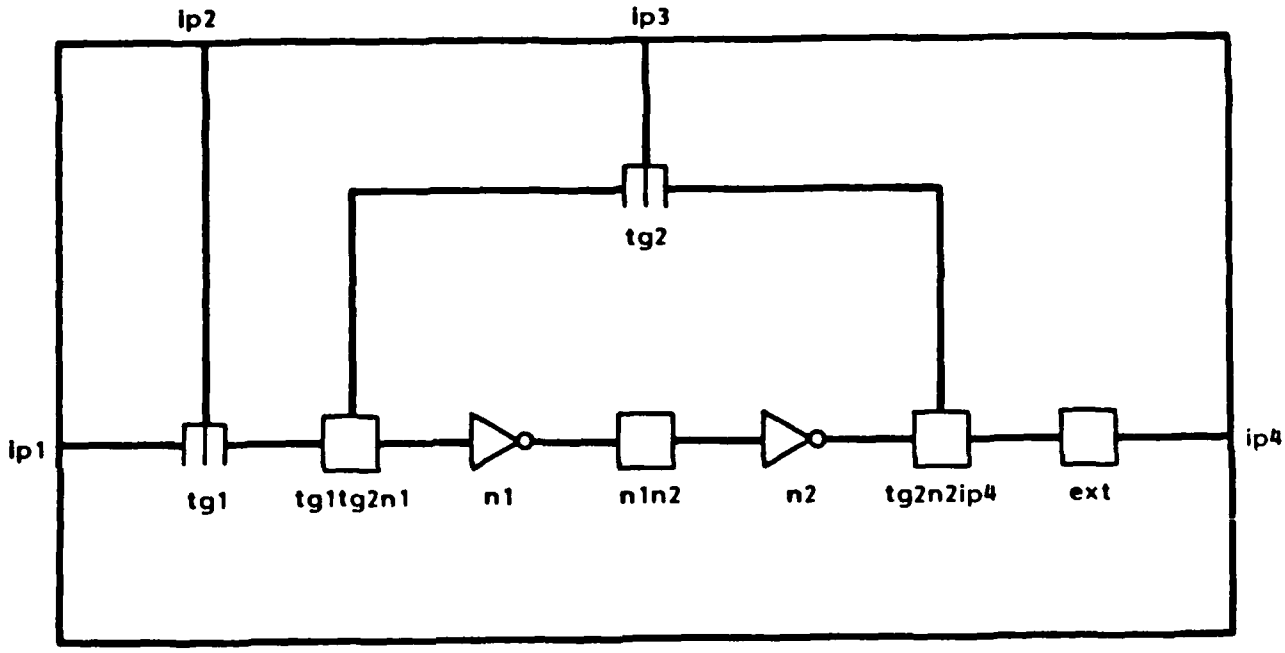


Figure 12 LATCH

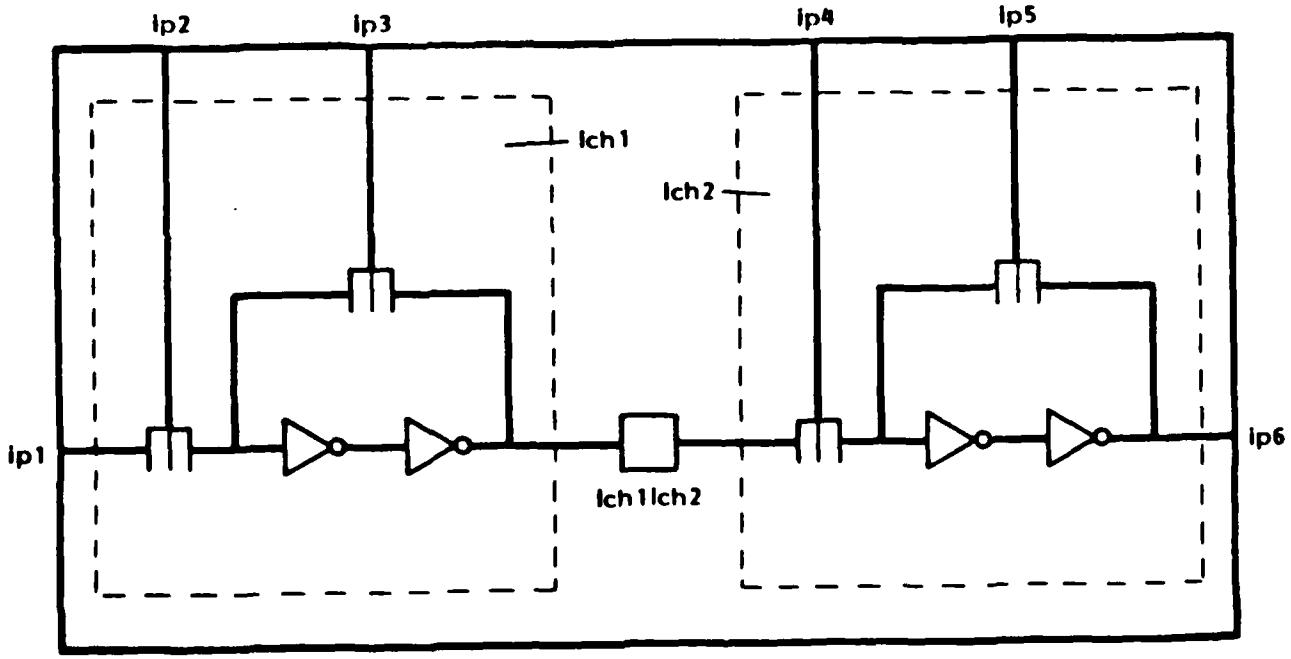


Figure 13 TWOLATCHES

DOCUMENT CONTROL SHEET

Overall security classification of sheet ..UNCLASSIFIED.....

(As far as possible this sheet should contain only unclassified information. If it is necessary to enter classified information, the box concerned must be marked to indicate the classification eg (R) (C) or (S))

1. DRIC Reference (if known)	2. Originator's Reference Memorandum 3825	3. Agency Reference	4. Report Security Classification UNLIMITED Unclassified
5. Originator's Code (if known)	6. Originator (Corporate Author) Name and Location Royal Signals and Radar Establishment		
5a. Sponsoring Agency's Code (if known)	6a. Sponsoring Agency (Contract Authority) Name and Location		

7. Title

MODELLING TRANSMISSION GATES IN ELLA

7a. Title in Foreign Language (in the case of translations)

7b. Presented at (for conference papers) Title, place and date of conference

8. Author 1 Surname, initials	9(a) Author 2	9(b) Authors 3,4...	10. Date	pp. ref.
Milner K R	Thorp T L			

11. Contract Number	12. Period	13. Project	14. Other Reference

15. Distribution statement

Unlimited

Descriptors (or keywords)

continue on separate piece of paper

Abstract

This Memorandum describes an algorithm for modelling transmission gates and gives details of its implementation in the Hardware Design and Description Language ELLA.

END

FILMED

11-85

DTIC