Some improved designs for the digital summation threshold logic (DSTL) gate

C. R. Edwards

School of Electrical Engineering, University of Bath, Claverton Down, Bath BA2 7AY

Some computer-generated improved designs for the digital summation threshold logic gate, first described by Hurst, are presented. These designs require considerably fewer logic elements than the original designs and have improved gate-delay parameters. The improved designs are shown to have delay times comparable to those of the 'bifurcated' gates of Reddy and Swamy, whilst enjoying a more elegant structure. Some implementations for these circuits are given.

(Received November 1976)

Downloaded from https://academic.oup.com/comjnl/article/21/1/73/348881 by guest on 19 April 2024

1. Introduction

Recently Hurst (1973) has proposed a digital summation threshold logic gate which overcomes the threshold tolerancing problems associated with more conventional designs. The circuits, which comprise the DSTL gates proposed heretofore, are 2-input AND/OR modules connected as a cellular array.

Edwards (1975a) has shown that, under the Rademacher/Walsh transform, an optimal DSTL gate having a greatly simplified structure may be used for the synthesis of functions of order n, $n \leqslant 4$ (and certain functions of higher-order). Such gates may be employed in the synthesis of higher-order functions when combined with exclusive-OR/NOR gates.

Reddy and Swamy (1974) have proposed a 'bifurcated' DSTL gate which is of comparable complexity to, but has a smaller propagation delay than, the original DSTL gate. It suffers the disadvantage of having a large number of signal path 'crossovers'.

This communication introduces some new designs for the DSTL gate which have more elegant structures than the gates detailed above.

2. Original gate parameters

The number of 2-input AND/OR logic modules required in the original (Hurst, 1973), n-input, DSTL gate is given by:

$$\sum_{a=1}^{n-1} (n-a) \text{ or } {}^nC_2 \qquad (1)$$

The maximum gate propagation delay* is:

$$2n - 3 \qquad (2)$$

The original DSTL gate design, for the case where $n = 8$, is shown in **Fig. 1.**

The bifurcated DSTL gate (Reddy and Swamy, 1974) is not wholly cellular in structure. An n-input bifurcated gate comprises two DSTL sub-arrays having p and q inputs respectively, where $p + q = n$, together with a 'sum' array.

The total number of 2-input AND/OR modules in the two DSTL sub-arrays is given by:

$$\left(\frac{n}{2}\right)\left(\frac{n}{2} - 1\right) + \frac{D^2}{4} \qquad (3)$$

where $D = |p - q|$

The 'sum' array comprises

$$p \times q \qquad (4)$$

2-input AND gates

together with

$$n - 1 \qquad (5)$$

OR gates with multiple inputs.

In Reddy and Swamy (1974) only gates for which $0 \leqslant D \leqslant 1$ are considered (this gives rise to designs with optimal delay times). For $0 \leqslant D \leqslant 1$ these OR gates have an average number of inputs given by:

$$1 + \frac{p \times q}{n - 1} \qquad (6)$$

and the maximum propagation delay† is:

$$n - 1 \qquad (7)$$

The design of a bifurcated DSTL gate, for the case where $n = 8$, is shown in **Fig. 2.**

Because the bifurcated gate is not composed wholly of 2-input AND/OR modules, no direct comparison with the original DSTL gate in terms of the number of modules is possible. The following points may be made:

1. If it is assumed that the number of active devices required to implement an i-input AND or OR gate is proportional to i, then the total number of active devices required in the bifurcated and original DSTL gates is approximately the same. The number of signal path 'crossovers' generated by the bifurcated gate is, however, very large. This may prove a significant problem when such a gate is fabricated in integrated-circuit form. The advantage of the bifurcated gate is its relatively small propagation delay, see also Section 4.
2. The optimised DSTL gate, first described by Edwards (1975a), is a special form of DSTL gate and is shown in **Fig. 4(*a*)**. This gate, together with inverting and exclusive-OR/NOR gates, will synthesise the great majority of functions of order n, $n \leqslant 4$ (and certain higher order functions). It may also be employed to synthesise functions of order $n > 4$ in an elegant fashion. It has six inputs and output weights of 1 (or more) and 2 (or more). The original design of this gate comprises five 2-input AND/OR modules and one 5-input OR gate.

3. New designs

The new designs of DSTL gate retain the modular (2-input AND/OR) components of the original DSTL gate but have essentially different cellular array structures.

The complexity of these designs is significantly less than that of either the original or bifurcated gates. Results to date indicate that the propagation delay of the new designs is

*Note the typographical error in Hurst (1973): $2m - 1$ should read: $2m - 3$.

†Note the typographical error in Reddy and Swamy (1974): $2(p - 3) + 2$ should read: $(2p - 3) + 2$.

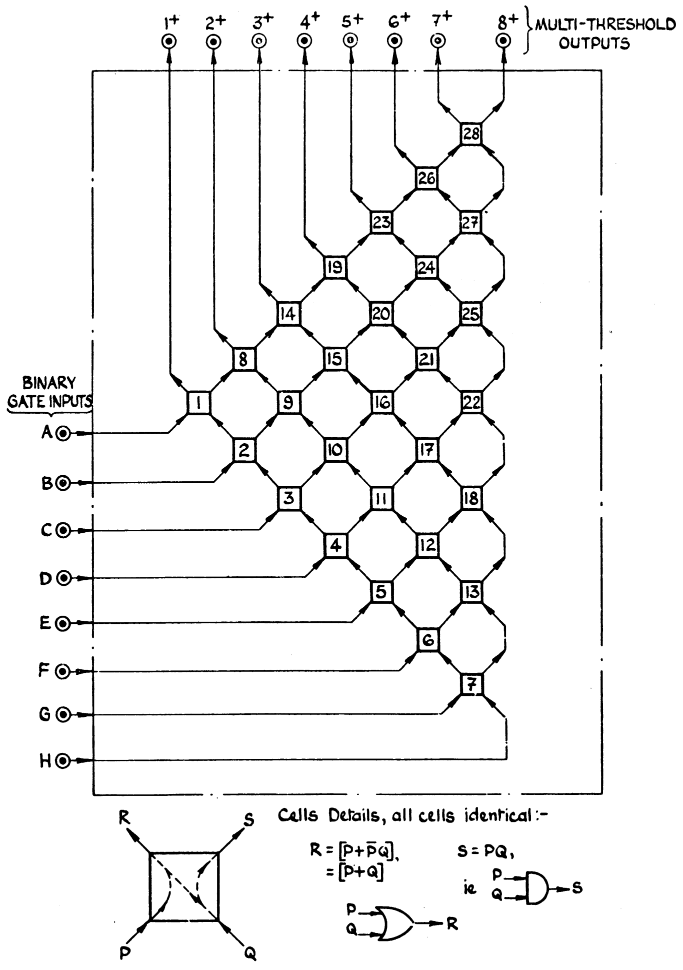


Fig. 1 Original DSTL gate (Hurst), n = 8

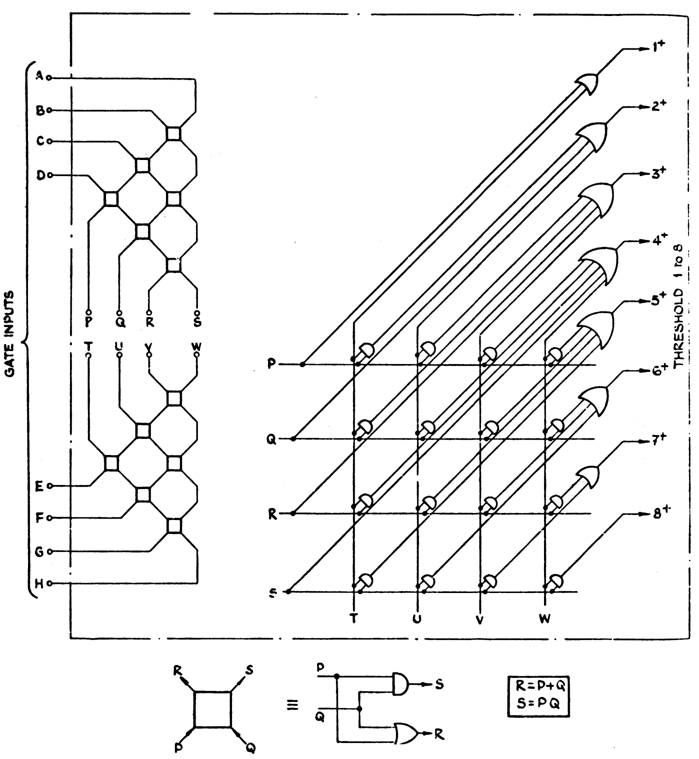


Fig. 2 Bifurcated DSTL gate (Reddy, Swamy), n = 8

comparable with that of the bifurcated gate but that the number of signal path 'crossovers' is much smaller.

The method used to design the new gate configurations is based upon Boolean function symmetries. Specifically, the synthesis proceeds from input to output under the interactive control of a suite of symmetry orientated computer programs. At each stage of synthesis the *optimal* symmetry of the form $f(x_1, \ldots, \bar{x}_i, x_j, \ldots, x_n) = f(x_1, \ldots, x_i, \bar{x}_j, \ldots, x_n)$ is detected for all output functions. *Optimal* here refers to the symmetry which maps onto the maximum number of true/false minterms in the spaces $\bar{x}_i, x_j$ and $x_i, \bar{x}_j$. This symmetry is then exploited by using the OR/AND module:

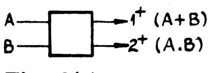


Fig. 3(a)

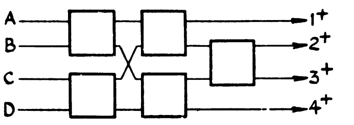


Fig. 3(b)

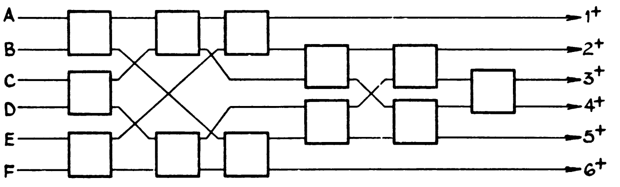


Fig. 3(c)

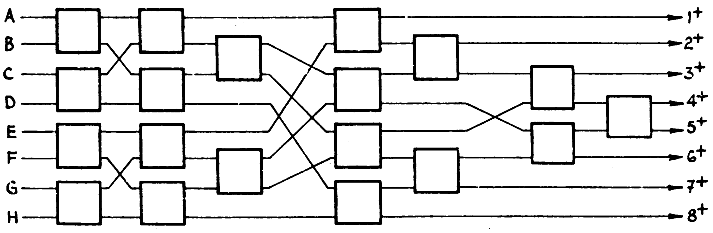


Fig. 3(d)

Fig. 3 New designs of DSTL gates, 2 $\leqslant n \leqslant$ 8, n even

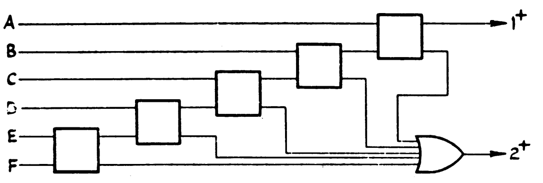


Fig. 4(a) Optimised DSTL gate

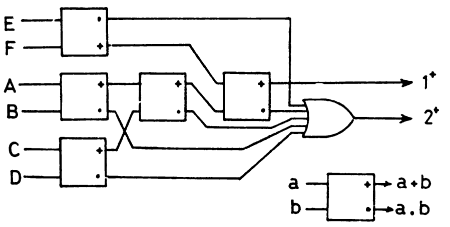


Fig. 4(b) Improved design of optimised gate

x_i	x_j	$(x_i + x_j)$	$(x_i \cdot x_j)$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

Downloaded from https://academic.oup.com/comjnl/article/21/1/73/445683 by guest on 19 April 2024

Note that this module is itself a universal 2-input threshold gate which maps $(x_i, x_j) = (0, 1)$ to $(1, 0)$; the space $(x_i, x_j) = (0, 1)$ is then allocated the 'don't care' value. Using the *optimal* criterion above it follows that the maximum number of 'don't care' states are generated at each stage of synthesis. To reduce propagation delay $\langle i, j \rangle$ is chosen to be *disjoint* whenever possible. See also the Appendix.

Eventually the problem is reduced to a set of functions having states: 'don't care' and 'one', which corresponds to a complete solution (such functions are simple throughput connections).

As a synthesis aid a visual display of the functions in n-space, using a nested Karnaugh-map format, is employed.

Some of the results of employing this method are shown in **Fig. 3** for $2 \leqslant n \leqslant 8$, n even. As may be expected, this design method gives rise to symmetric implementations. For cases where n is odd several solutions may be formulated, each essentially of a semi-symmetric form, the different designs having different *distributions* of propagation delays at the outputs.

It should be noted that, unlike the original DSTL gate, the propagation delay to each of the outputs is related to the complexity of the function at that output, e.g. for $n = 6$, delay to 1^+ (OR) and 6^+ (AND) is 3 whereas delay to 2^+ (any 2 or more of 6) is 5, etc.

Inspection of **Table 1** shows that

(*a*) The new DSTL gate employs significantly fewer AND/OR modules than the original DSTL gate and is less complex than the bifurcated gate (with the provisos outlined in Section 2).

(*b*) The new gate has maximum propagation delay times which are significantly smaller than the original DSTL gate and comparable with the bifurcated gate.

(*c*) The new gate has significantly fewer signal path 'crossovers' than the bifurcated gate (The original DSTL gate has no 'crossovers').

The modified design for the optimal DSTL gate is shown in Fig. 4(*b*). The number of components required is exactly the same as for the original design (Fig. 4(*a*)) but the maximum propagation delay is now 4 (as opposed to 6 in the original), and 2 'crossovers' have been generated.

4. Further notes

It is clear that the delay times of the bifurcated gate may be further reduced by 'bifurcating' each of the two DSTL sub-arrays. Indeed this process may be repeated until a 'completely bifurcated' design is generated. **Fig. 5** shows an example of a gate of this type having $n = 8$, delay $= 5$. The propagation delays for gates of this type are shown in **Table 2.** This gate suffers the disadvantage however of having a very large number of 'crossovers' which may preclude its efficient I/C fabrication.

Another possible gate may be constructed by replacing the DSTL portions of the bifurcated gate by new DSTL arrays described above. This will reduce propagation delay without significantly increasing the number of 'crossovers' already present in the bifurcated design; the structure of the gate would also be simplified.

Table 1 Table of comparisons

Parameter	*Order n*	*Original DSTL*	*Bifurcated DSTL*	*New DSTL*
Number of modules	2	1	*	1
	4	6	*	5
	6	15	*	12
	8	28	*	19
Maximum delay (gates)	2	1	1	1
	4	5	3	3
	6	9	5	6
	8	13	7	7
Signal path 'crossovers'	2	0	0	0
	4	0	7	1
	6	0	22	6
	8	0	56	9

*Approximately same complexity as original DSTL (see text)

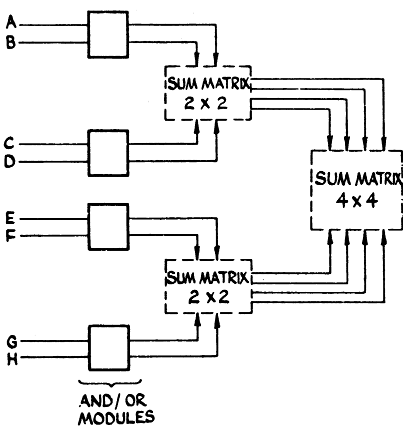


Fig. 5 Completely Bifurcated DSTL gate, $n = 8$

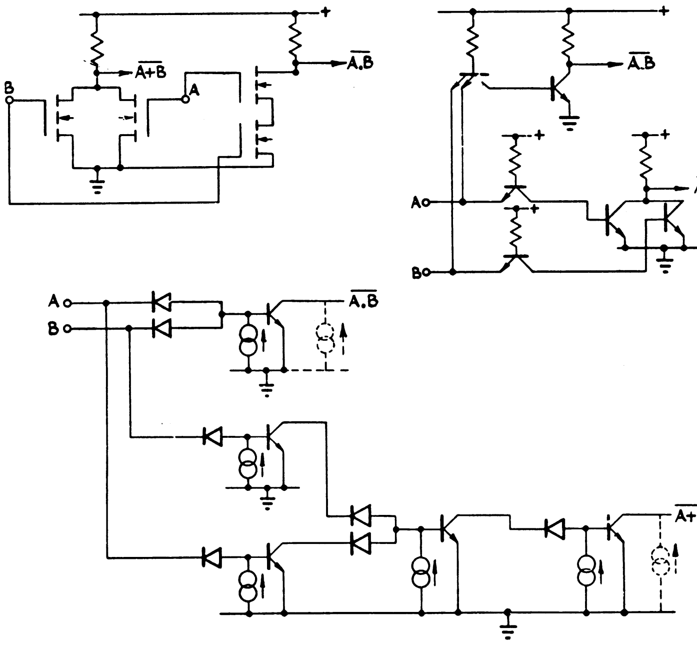


Fig. 6 Possible MOS, T^2L and SFL 2-input NAND/NOR modules

Table 2

n	2	3	4	5	6	7	8	9	10	11	12
Propagation delay	1	3	3	5	5	5	5	7	7	7	7

Downloaded from https://academic.oup.com/comjnl/article/21/1/73/445583 by guest on 19 April 2024

5. Gate fabrication*

In most technologies a 2-input AND/OR module is a cumbersome vehicle, a NAND/NOR module being preferred for reasons of propagation delay. In fact DSTL gates may be fabricated using NAND/NOR modules if certain connection conventions are obeyed, but even then large numbers of active devices must be employed. See **Fig. 6.** To overcome these problems some new designs are presented.

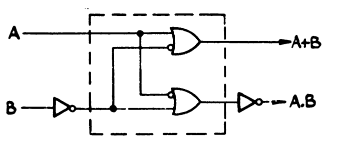
An alternative fabrication, which is a direct replacement for the 2-input AND/OR module, suitable for T^2L and MOS is shown schematically in **Fig. 7**(*a*). Corresponding implementations in T^2L and MOS appear in Fig. 7(*b*) and (*c*). (See Edwards (1975b) for a description of the operation of this type of circuit.)

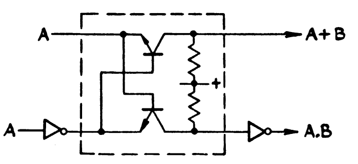
An alternative fabrication for SFL (an improved form of I^2L) is shown in **Fig. 8**(*a*) and (*b*). Other configurations are shown in Figs. 7(*d*) and 8(*c*) (see Blatt *et al*, 1974; Hart and Slob, 1973).

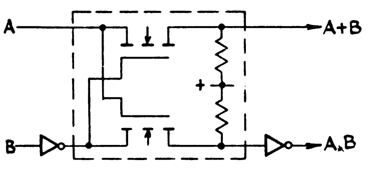
In these figs. the inverters have been deliberately excluded from the 'cells' (shown dotted) because, as shown below, when these modules are used to implement the type of array shown in Fig. 3, most of these inverters are rendered redundant. (This is also true of the original designs of Hurst.)

When the modules of Figs. 7(*a*)-(*c*) and 8(*a*), (*b*)† are used to implement the arrays of Fig. 3 it is noted that:

1. In some cases inverters cascade, in which case they may be eliminated.
2. Inverters which appear at the inputs and outputs of the array implement a weight negation at the input and output. If these inverters are removed, therefore, the gate may be redefined in terms of -ve weights as shown in **Fig. 9**(*b*). (Since there is a probability of $\frac{1}{2}$ of requiring a -ve weight at any input or output this gate is no less 'valid' than that of Fig. 3.)
3. The 'cells' of Fig. 9 correspond to the 'cells' of Figs. 7 and 8.







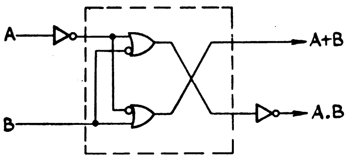
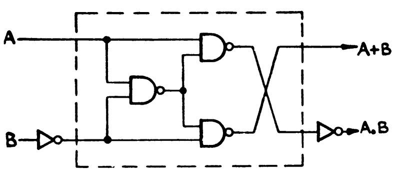
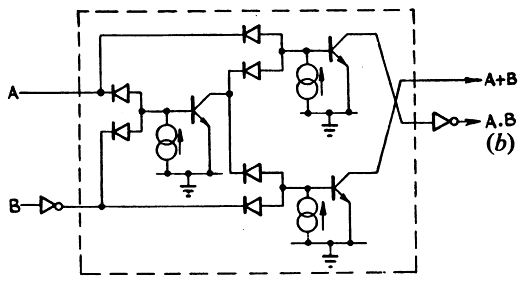


Fig. 7 Modified AND/OR modules for T^2L and MOS





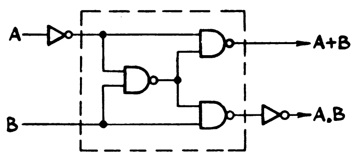
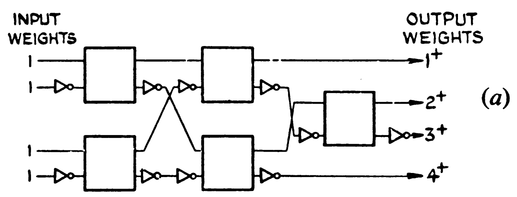


Fig. 8 Modified AND/OR modules for SFL



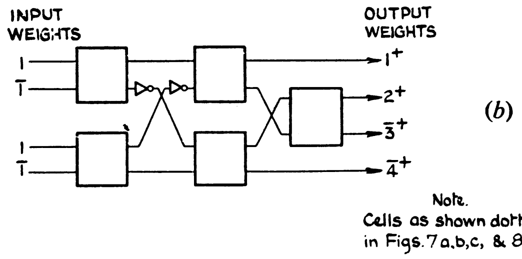


Fig. 9 Reduction of inter-module inverters

Fig. 10 shows the implementation of an 8th order threshold gate after inverter elimination. Cell type 1 corresponds to Fig. 7(*a*), 8(*a*) and type 2 to Fig. 7(*d*), 8(*c*).

When circuits of the type shown in Fig. 7 are cascaded, signal paths, not including inverters, are subject to voltage drops at each stage ($V_{CE(SAT)}$ for T^2L and $V_{DS(ON)}$ for MOS). In addition it is possible that devices which appear at the beginning of such a cascade will have to sink current from resistive loads of subsequent stages; thus increasing likely propagation delay. Of course signal paths which include inverters are 'ground referenced' at each inverter and so only cascade paths between inverters contribute to the above effects. It is clear from the above that the maximum permitted cascade length will be determined from the required maximum switching speed and noise immunity parameters. These in turn will depend upon the technology employed; for example in the T^2L case Schottky diode fabrication may be used.

*The following abbreviations are employed:
T^2L transistor-transistor logic
MOS metal-oxide-silicon (Field effect devices)
SFL substrate fed logic (Blatt *et al.*, 1974)
I^2L integrated injection logic (Hart and Slob, 1973).

†Although Fig. 7(*a*) and (*d*); Fig. 8(*a*) and (*c*) are functionally identical their topological differences permit inverter elimination when they are employed in the final gate designs.

Downloaded from https://academic.oup.com/comjnl/article/21/1/73/445683 by guest on 19 April 2024

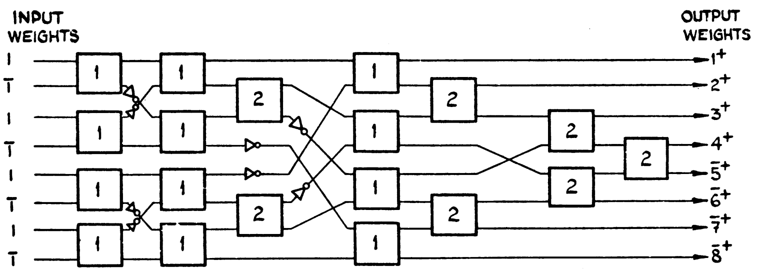


Fig. 10 An 8th order threshold gate fabrication

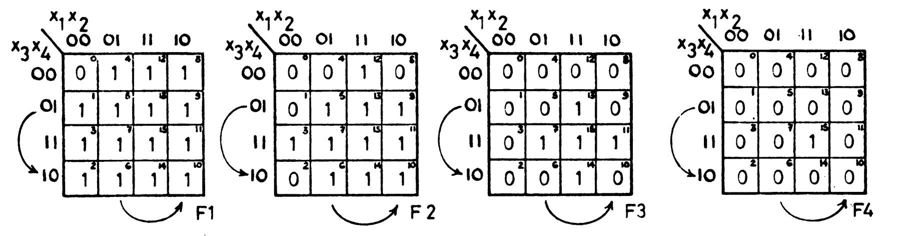


Fig. 11

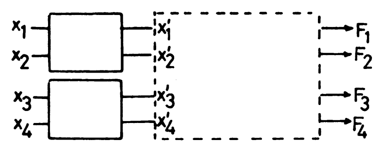


Fig. 12

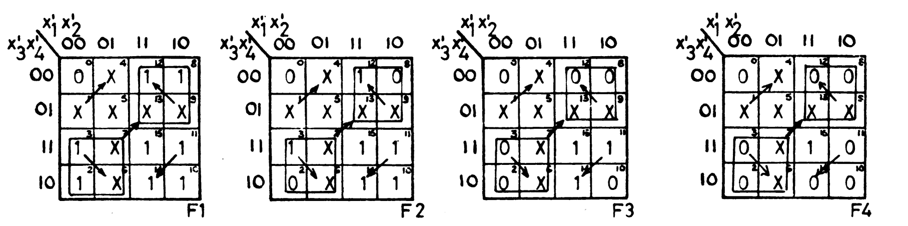


Fig. 13

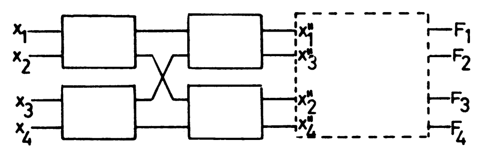


Fig. 14

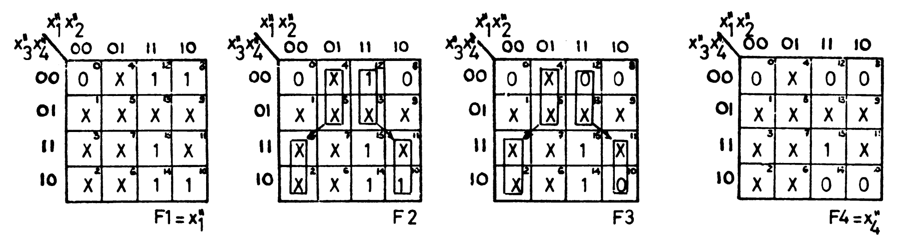


Fig. 15

Cascaded circuits of the type shown in Fig. 8 do not have the same noise-immunity problems as those of Fig. 7, however a gate delay of 2 is generated by each stage of any cascade (a gate delay of 1 is also present for each inverter employed). The predicted high speed of SFL, together with its low power consumption and efficient utilisation of silicon area, makes the fabrication using circuits of the type shown in Fig. 8 very attractive.

6. Conclusions

Some new designs of DSTL gate have been presented which have a more simple structure than the original designs of Hurst and the bifurcated designs of Reddy and Swamy. The propagation delays of this type of gate have been shown (from the given results) to be comparable with that of the bifurcated gate.

Further methods of reducing the propagation delay and complexity of these gates have been outlined.

Some novel methods of fabricating the new gate designs have been proposed which, within certain practical limits, give very elegant implementations.

Appendix

The step-by-step synthesis of an $n = 4$ input universal threshold gate (Fig. 3(*b*)) is given. The four required output functions F1-F4 are shown in **Fig. 11**; each input has unity weighting, and

Output:	*Function*	*Output weighting*	
	F1	1	(OR)
	F2	2	
	F3	3	
	F4	4	(AND)

Inspection of these functions shows that they all have a symmetry $x_1 \neq x_2$, $x_3 \neq x_4$ (among others), these two symmetries involve disjoint variables. These symmetries may be otherwise expressed as:

> the functions are identical for $x_1 = 1, x_2 = 0$ and $x_1 = 0, x_2 = 1$ for any values of x_3, x_4 and the functions are also identical for $x_3 = 1, x_4 = 0$ and $x_3 = 0, x_4 = 1$ for any value of x_1, x_2. The equivalent n-spaces are indicated by arrows in Fig. 11.

Suppose that we employ OR/AND modules as shown in **Fig. 12** to exploit these symmetries, viz.

Downloaded from https://academic.oup.com/comjnl/article/21/1/73/445683 by guest on 19 April 2024

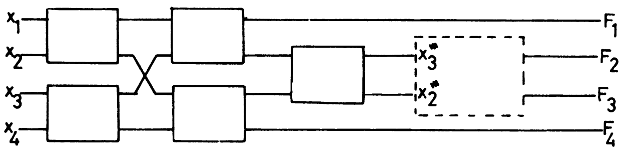
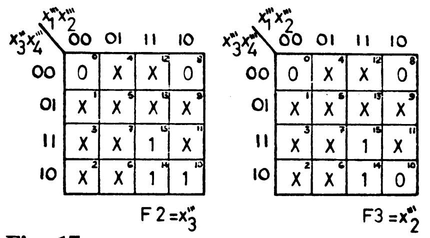


Fig. 16



$x_3'''x_4'''$ \ $x_1'''x_2'''$	00	01	11	10
00	0	X	X	0
01	X	X	X	X
11	X	X	1	X
10	X	X	1	1

F 2 $=x_3'''$

$x_3'''x_4'''$ \ $x_1'''x_2'''$	00	01	11	10
00	0	X	X	0
01	X	X	X	X
11	X	X	1	X
10	X	X	1	0

F3 $=x_2'''$

Fig. 17

$$x_1' = x_1 + x_2, \qquad x_2' = x_1 \cdot x_2, \qquad x_3' = x_3 + x_4$$
$$x_4' = x_3, x_4$$

This has the effect of *mapping* n-space $x_1 = 0$, $x_2 = 1$ *onto* $x_1 = 1$, $x_2 = 0$ and n-space $x_3 = 0$, $x_4 = 1$ *onto* $x_3 = 1$, $x_4 = 0$ as indicated in Fig. 11.

The n-spaces $x_1' = 0$, $x_2' = 1$ and $x_3' = 0$, $x_4' = 1$ may then be allocated the 'don't care' state 'X' since these n-spaces *cannot* be addressed by the inputs x_1, x_2, x_3, x_4. The four functions remaining to be synthesised after this step are shown in **Fig. 13.** These four functions also have common disjoint symmetries, viz. $x_1' \neq x_3'$ and $x_2' \neq x_4'$ (shown boxed and arrowed respectively in Fig. 13).

It should be noted that the importance of choosing *disjoint* symmetries is that propagation delays are minimised.

Fig. 14 shows the implementation of these symmetries by OR/AND modules viz.

$$x_1'' = x_1' + x_3', \qquad x_3'' = x_1' \cdot x_3',$$
$$x_2'' = x_2' + x_4', \qquad x_4'' = x_2' \cdot x_4'$$

The four functions remaining to be synthesised are shown in **Fig. 15.**

Now F1 is solved completely by letting F1 $= x_1''$ and F4 is solved completely by letting F4 $= x_4''$. Functions F2 and F3 have a common symmetry $x_3'' \neq x_2''$ (shown boxed in Fig. 15).

The implementation of this symmetry is shown in **Fig. 16**, viz

$$x_3''' = x_3'' + x_2'', \qquad x_2''' = x_3'' \cdot x_2''$$

The two functions (F2, F3) remaining are shown in **Fig. 17.** These are solved completely by letting F2 $= x_3'''$ and F3 $= x_2'''$. This gives the solution shown in Fig. 3(*b*).

In practice this procedure is carried out by a computer aided design method. Edwards (1977) should be consulted for a more detailed discourse on this design method.

References

Blatt, V., *et al.* (1974). Substrate Fed Logic—An improved form of injection logic, International Electron. Dev. Meeting (IEEE), *Technical Digest*, Washington DC, p. 511 fol.

Edwards, C. R. (1975(a)). The application of the Rademacher-Walsh transform to Boolean function classification and threshold logic synthesis, *IEEE Trans. Electron. Comput.*, Vol. C-24, pp. 48-62.

Edwards, C. R. (1975(b)). Some novel exclusive—OR/NOR circuits, *Electronics Lett.*, Vol. 11, No. 1, pp. 3-4.

Edwards, C. R. (1977). Design of Easily Tested Circuits using Mapping and Spectral Techniques, *The Radio and Electronic Engineer*, Vol. 47, No. 7, pp. 321-342.

Hart, C. M., and Slob, A. (1973). Integrated Injection Logic (I²L), *Philips tech Rev.*, Vol. 33, No. 3, pp. 76-85.

Hurst, S. L. (1973). Digital-Summation Threshold-Logic gates: a new circuit element, *Proc.IEE*, Vol. 120, No. 11, pp. 1301-1307.

Reddy, V. C. V., and Swamy, P. S. N. (1974). Note on Digital-Summation Threshold-Logic gates, *Proc.IEE*, Vol. 121, No. 10, pp. 1085-1086.

Book review

Computer-Aided Design of Digital Systems, by D. Lewin, 1977; 313 pages. (*Edward Arnold*, £15·00)

This book will be of interest mainly to students of computer science at a postgraduate level, and to those practising engineers who are already familiar with 'formal' logic design methods. The book surveys the current status of computer aids in the fields of logic-network synthesis, logic simulation and logic testing. The subject of system specification, both by means of register transfer languages and by graph-theoretic models is also discussed.

The main barrier to the more widespread acceptance of computer aided logic design is the lack of sufficiently powerful algorithms, especially methods applicable to circuits using MSI and LSI components. Most of the algorithms described in this book are orientated toward design using flipflops and discrete NAND and NOR gates; this limits its usefulness to designers of CAD systems in industry, who have to work with the current technology.

The longest chapter in this book (117 pages) covers the topic of logic-network synthesis. A large number (over 20) of algorithms are described, for state reduction, state assignment and for implementation of the resulting switching functions in a particular 'logic family'. The algorithms described first appeared in a variety of journals: Ph.D theses, etc.; this book serves the useful purpose of collecting and comparing such a diversity of methods. The algorithms are described in Professor Lewin's usual lucid style, and a large number of helpful worked examples are provided. Many of these algorithms suffer severe limitations on the size of problem which they can handle. Unfortunately, little numeric information is provided in this book to indicate to the reader the limitations of each technique.

The chapter on system specification contains an up-to-date survey of hardware description languages, and also describes more recent developments, e.g. the use of Petri nets. The subjects of logic simulation for design verification and for test-program validation are treated in rather less detail; for example, the techniques of deductive fault simulation and the use of worst case timing are mentioned but not described in detail. The book concludes with a review of the subjects of logic-circuit testing and testable logic design. The book provides a large number (nearly 300) of references, and a useful subject and authors index.

D. Bumstead (Poole)

Downloaded from https://academic.oup.com/comjnl/article/21/1/73/445683 by guest on 19 April 2024