Minimal realization of a multiple output AND/OR combination circuit

Orhan UÇAR[1] and Ahmet DERVİŞOĞLU[2]

Abstract

In this paper we present an algorithm to realize m Boolean functions with n variables by the use of a two level AND/OR combination circuit. The algorithm is implemented in an efficient computer program MORP(Multiple Output Reduction Program), which gives the minimal expressions of the given Boolean functions. The algorithm always gives a minimal solution or all minimal solutions. However, if it takes too long time to obtain a minimal solution then the algorithm may give a near-minimal solution in a shorter time.

The algorithm first determines the prime implicants. Then a cover C_i is determined which contains as few prime implicants as possible and gives a near-minimal solution. In the next step a cover C_j with fewer prime implicants is searched; if such a cover does not exist the solution obtained from C_i is a minimal solution. If C_j is obtained then C_k which contains fewer prime implicants than C_j is searched. Continuing this way, a minimal solution is always obtained. However search can be terminated at the end of a given time duration in which case a near-minimal solution is obtained.

MORP is run on the PC and tested on some two level circuits, including some MCNC benchmarks and the minimal solutions are obtained in reasonable time duration.

1 Introduction

Multiple output two level minimal realization problem is a well known problem. However well known reduction programs like ESPRESSO, SCHERZO and others give a near-minimal solution rather than a minimal solution. However for some purposes such as to test other programs which give a near-minimal solution, a program which gives minimal or all minimal solutions is needed.

The synthesis of a two level combinational circuit is important in the realization of sequential circuits. Because the synthesis of a sequential circuit can be reduced to the synthesis of a combinational circuit and most of the time, multi-level combinational circuits are obtained from the two-level combinational circuits.

In this paper we present an algorithm to realize m Boolean functions with n variables by the use of a two level AND/OR combination circuit. The algorithm is implemented in an efficient computer program MORP(Multiple Output Reduction Program), which gives the minimal expressions of the given Boolean functions. The algorithm always gives a minimal solution or all minimal solutions. However, if it takes too long time to obtain a minimal solution then the algorithm may give a near-minimal solution in a shorter time.

The algorithm first determines the prime implicants. Then a cover C_i is determined which contains as few prime implicants as possible and gives a near-minimal solution. In the next step a cover C_j with fewer prime implicant is searched; if such a cover does not exist the solution obtained from C_i is a minimal solution. If C_j is obtained then C_k which contains fewer prime implicants than C_j is searched. Continuing this way, a minimal solution is always obtained. However search can be terminated at the end of given time duration then, a near-minimal solution is obtained.

MORP is run and tested using the PC (Pentium 200MMX with 32MB RAM), on some two level circuits, including some MCNC benchmarks and the results are given in Table 1. From Table 1, it can be seen that despite of the limited computing resources, MORP is efficient on most of the benchmarks.

II The Algorithm

The algorithm given below deals with covering problem. Therefore it can be used in other related

[1]NETAŞ, Alemdağ Caddesi, Ümraniye, 81244 Istanbul, TURKEY
[2]Istanbul Technical University, Electrical and Electronics Engineering Faculty, Department of Electronics and Communication Engineering, 80626 Maslak, Istanbul, TURKEY

design problems in which covering techniques are used.

Step-1: Determination of Prime Implicants

This algorithm finds the prime implicants by the classic Quine-McCluskey method. In this method minterms of the functions are ordered according to the weights. The weight of a minterm is the number of 1s it has. Minterms that have equal weights are grouped and adjacent groups of minterms are compared. If two minterms are different only at one position, a new product term which has a '-' at this position is produced and these minterms are signed by '*'. This procedure ends if a new product term can not be obtained. Product terms or minterms that are not signed by the '*' are the prime implicants of the related functions.

Step-2: Determination of a near-minimal solution

In this step covering table is obtained. Covering table has a row for each prime implicant and a column for each minterm of the output functions.

In the first part of this step covering table is searched for an essential prime implicant. If found, these prime implicants must be included in the minimal cover. If we add these prime implicants to the cover, we can delete these rows and the columns they cover. Thus essential prime implicants are added to the near-minimal solution and related rows and columns are deleted.

In the second part of this step row domination procedure is applied to the remaining covering table. Dominated rows are deleted and first part of this step is applied to the remaining covering table.

First and second parts of this step are applied repeatedly. If any essential prime implicant and any dominated row can not be found, this procedure stops.

When the procedure ends then greedy method is applied to the remaining table. This is the third part of this step. By this method a prime implicant which covers maximum number of minterms of the remaining table is selected. Selected prime implicant is added to the near-minimal solution. Related row and columns are deleted from the covering table. After this greedy procedure, first and second part of this step are applied. Thus first, second and third parts of this step are applied until the table has no row. Selected prime implicants give a near-minimal solution. Also number of the prime implicants of this solution give an upper bound.

Step-3: Determination of minimal solutions

In this step, results of the first and second part of the Step-2 described above is used. Thus essential prime implicants are added to the minimal solution. Then prime implicant combinations which cover the minterms of the remaining covering table are checked. The prime implicant combination which cover the minterms of the remaining covering table and has a minimum number of prime implicants are added to the minimal solution. Thus these prime implicants and the essential prime implicants give the minimal solution. Since all combinations are checked for a minimal solution, all minimal solutions are obtained.

Example 1:

Consider the functions f_1 and f_2 given below.

$f_1 (X_1, X_2, X_3) = \Sigma(4,5,7)$
$f_2 (X_1, X_2, X_3) = \Sigma(0,1,4)$

These functions are described to the reduction program MORP as follows.

```
.i3
.o2
.p5
000 01
001 01
100 11
101 10
111 10
```

The table which shows the minterms according to the weights is given below.

Minterm	$X_3 X_2 X_1$	$f_2 f_1$
0*	0 0 0	1 0
1*	0 0 1	1 0
4	1 0 0	1 1
5*	1 0 1	0 1
7*	1 1 1	0 1

If we compare minterms that are in the adjacent groups we obtain the following table.

Minterm	$X_3 X_2 X_1$	$f_2 f_1$
(0*-1*)	0 0 -	1 0
(0*-4)	- 0 0	1 0
(1-5)	- 0 1	0 0
(4-5*)	1 0 -	0 1
(5*-7*)	1 - 1	0 1

Repeating the operation we find,

minterm	$X_3X_2X_1$	f_2f_1
(0-1)(4-5)	- 0 -	0 0

From the tables above we obtain the prime implicant table which is given below.

Prime Implicant			f_2 f_1
PI_1 :	X_3 X_2' X_1'	(1 0 0)	1 1
PI_2 :	X_3'X_2'	(0 0 -)	1 0
PI_3 :	X_2' X_1'	(- 0 0)	1 0
PI_4 :	X_3 X_2'	(1 0 -)	0 1
PI_5 :	X_3 X_1	(1 - 1)	0 1

Then the covering table which has a row for each prime implicant and a column for each minterm of the functions f_1 and f_2 is ,

	f_1			f_2		
	4	5	7	0	1	4
PI_1	X					X
PI_2				X	X	
PI_3				X		X
PI_4	X	X				
PI_5		X	X			

If we look at the table above, we see that second prime implicant PI_2 and fifth prime implicant PI_5 are the essential prime implicants. These prime implicants must be included in the minimal cover. If we add these prime implicants to the cover, we can delete these rows and the columns they cover. The resulting table is,

	f_1	f_2
	4	4
PI_1	X	X
PI_3		X
PI_4	X	

From the table above we see that the row corresponding to PI_1 dominates the remaining two rows. If we aelete the rows corresponding to PI_3 and PI_4, we have,

	f_1	f_2
	4	4
PI_1	X	X

This table has only one row. Therefore the row indicated by the prime implicant PI_1 is selected. Thus minimal cover for the functions f_1 and f_2 is obtained as given below.

$$f_1 = PI_5 + PI_1 = X_3\,X_1 + X_3\,X_2'\,X_1'$$
$$f_2 = PI_2 + PI_1 = X_3'\,X_2' + X_3\,X_2'\,X_1'$$

Example 2:

Assume that the covering table is,

	f_1			f_2		
	m_0	m_1	m_2	m_3	m_4	m_5
PI_1	X					X
PI_2			X		X	
PI_3				X		X
PI_4	X	X		X	X	
PI_5		X	X			

As seen from the table above, there are no essential prime implicants. Therefore in the first part of the Step-2, a prime implicant can not be selected.

In the second part of the Step-2, a row, which is dominated by another row is searched; but there are no row dominated by another rows. Therefore in the second part of the Step-2, a row can not be deleted from the covering table.

In the third part of this step, greedy method is applied to this table. As a result of this method PI_4 which covers maximum number of minterms is selected. By deleting the row PI_4 and columns it covers, the following table is obtained.

	f_1	f_2
	m_2	m_5
PI_1		X
PI_2	X	
PI_3		X
PI_5	X	

As seen from the table above, there are no essential prime implicants. Therefore a prime implicant can not be selected; but the row PI_1 dominates PI_3 and the row PI_2 dominates PI_5. Removing the corresponding rows we have,

	f_1	f_2
	m_2	m_5
PI_1		X
PI_2	X	

Now PI_1 and PI_2 are essential prime implicants. Therefore, these prime implicants are added to the near-minimal solution and the following near-minimal solution is obtained.

$$f_1 = PI_4 + PI_2$$
$$f_2 = PI_4 + PI_1$$

Since there are 3 prime implicants in the cover, the upper bound upper bound for this example is 3.

In the next step, a minimal cover which has minimum number of the prime implicants is searched.

Because of the upper bound is three, a search tree with the level of two is created. To do this, two minterms, which has lower weights, are selected. The weight of a minterm is the number of prime implicants, which cover it. In other words, number of 'X' signs in the the corresponding column gives the weight.

Covering table is shown below.

	f_1			f_2		
	m_0	m_1	m_2	m_3	m_4	m_5
PI_1	X					X
PI_2			X		X	
PI_3				X		X
PI_4	X	X		X	X	
PI_5		X	X			

Hence the weights are,

minterms	*weight*
m_0	2
m_1	2
m_2	2
m_3	2
m_4	2
m_5	2

Since the weights of all minterms are equal, randomly selected two minterms will be used to create the search tree. If we select the minterms m_0 and m_1, the following search tree is obtained.

$$\begin{tikzcd}
 & & & \text{root} \arrow[dll] \arrow[drr] & & & \\
 & PI_1 \arrow[dl] \arrow[dr] & & & & PI_4 \arrow[dl] \arrow[dr] & \\
PI_4 & & PI_5 & & PI_4 & & PI_5
\end{tikzcd}$$

From the search tree the combinations of the prime implicants and the minterms covered by each combination are given below.

PI_1 , PI_4	: m_0 , m_1 , m_3 , m_4 , m_5
PI_1 , PI_5	: m_0 , m_1 , m_2 , m_5
PI_4 , PI_4	: m_0 , m_1 , m_3 , m_4
PI_4 , PI_5	: m_0 , m_1 , m_2 , m_3 , m_4

Since there is no combination which covers all minterms, the near-minimal solution, which has three prime implicants, is a minimal solution.

3 Conclusion

In this paper we present an algorithm to realize m Boolean functions with n variables by the use of a two level AND/OR combinational circuit. The algorithm is implemented with an efficient computer program MORP(Multiple Output Reduction Program), which gives the minimal expressions for Boolean functions. The algorithm always gives a minimal solution or all minimal solutions. However, if it takes too long time to obtain a minimal solution then the algorithm may give a near-minimal solution in a shorter time.

MORP is run and tested using the PC (Pentium 200MMX with 32MB RAM), on some two level circuits, including some MCNC benchmarks and the results are displayed in Table 1. From Table 1, it can be seen that despite limited computing resources, MORP is efficient on most of the benchmarks. The near-minimal solution determination algorithm found minimal solutions for all benchmarks except last three benchmarks apex4, sao2 and ex5.

Table 1: Experimental Results

					MORP (near-minimal)		MORP (minimal)		MOP		ESPRESSO	
Benchmark	**i**	**o**	**NM**	**NPI**	**NP$_s$**	**Time (s.)**	**NP$_s$**	**Time (s.)**	**NP$_s$**	**Time (s.)**	**NP$_s$**	**Time (s.)**
xor5	5	1	16	16	16	0	16	0	16	0	16	0
rd53	5	3	42	51	31	0	31	0	31	0	31	0
squar5	5	8	85	71	25	0	25	9	29	0	25	1
con1	7	2	156	24	9	0	9	0.1	9	0.1	9	0
rd73	7	3	192	211	127	0	-	-	127	1.5	127	1
rd84	8	4	410	632	254	1	-	-	248	9.9	255	1
rd84v3	8	4	79	97	50	0	50	73	77	0.3	50	0
misex1	8	7	548	28	12	0.3	12	0.3	17	0.2	12	0
5xp1	7	10	524	345	63	0.3	-	-	101	0.1	65	0
5xp1v2	7	10	412	169	25	0.3	25	0.4	43	0.1	25	0
apex4	9	19	2770	2336	501	120	-	-	464	4.6	436	18
sao2	10	4	746	191	65	26	-	-	79	26	58	0
ex5	8	63	7588	2489	77	57	-	-	-	-	74	2

i: number of inputs o: number of outputs NM: number of minterms NPI: number of prime implicants
NP$_s$: number of prime implicants in the solution

References

[1] E.J.McCluskey, Logic Design Principles, Prentice-Hall Inc., Englewood Cliffs, N.J., 1986.

[2] Puri R. and Gu J., "An Efficient Algorithm to Search for Minimal Closed Covers in Sequential Machines", IEEE Transactions on CAD of Integrated Circuits and Systems, v.12, n.6, June 1993, pp.737-745.

[3] Dervişoğlu A., Hacıoğlu H. and Uçar O., "A New Method for State Assignment in Incompletely Specified Asynchronous Sequential Machines", Proceedings ECCTD'97, pp.690-694, Budapest, September 1997.

[4] Uçar O. and Dervişoğlu A., "State Reduction of Incompletely Specified Finite Sequential Machines by the Use of Closed Compatible Pairs", Proceedings ECCTD'99, pp.1375-1378, Stresa, Italy, September 1999.

[5] Dervişoğlu A., "Logic Design Course Notes", Istanbul Technical University, Electric and Electronics Faculty, 1999.