EXPLANATION AND IMPLEMENTATION

OF A 112 - b TRANSMISSION

GATE ADDER

By

XIAOBO LIU

Master of Science

Beijing Polytechnic University

Beijing China

1985

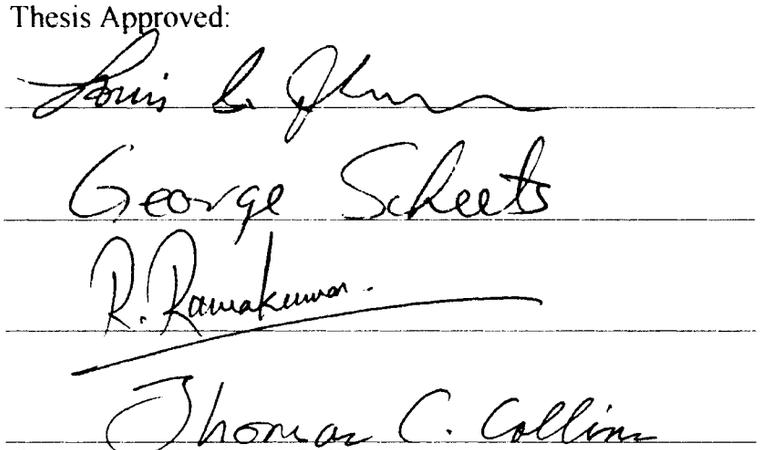
Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 1994

EXPLANATION AND IMPLEMENTATION

OF A 112 - b TRANSMISSION

GATE ADDER

Thesis Approved:



Dean of the Graduate College

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Louis G. Johnson for his guidance and advice throughout my research work. Without his guidance completion of this thesis would have been difficult. My sincere appreciation extends to my other committee members Dr. R.G. Rumakumar and Dr. George M. Sheets.

Finally, I would like to thank the Department of Electric and computer engineering for supporting during these my study at Oklahoma State University.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

Adders are very often in the critical path of the computer, so it is very important that their performance will not limit the cycle time of the machine. In VLSI applications, area and power are also important factors which must be taken into account in the design of a fast adder. High-speed floating-point arithmetic circuits are required for high performance computer systems. One choice is the carry - skip adder, which, because of its topological regularity and layout simplicity, is considered a good compromise in terms of area and performance. But in a carry select adder, two ripple carry adder structures are built, one with a zero carry-in and the other with a one carry-in. The previous carry then selects the appropriate sum using a multiplexer or tri-state adder gates. The stage carries and the previous carry are gated to form the carry for the succeeding stage. In other words, the carry select adder uses two sum signals instead of one sum signal, thus wasting transistors[1]. Another choice is a kind of improved carry - skip adder which uses one carry adder structure to implement the function of carry - skip adder[2]. Because there are only half the numbers of the transistors in the latter adder, it is faster than a conventional carry select adder.

The transmission gate is a pair of transistors connected to function as complementary switch. It consists of an n-channel transistor and a p-channel transistor. The control signal is applied to the gate of the n-device, and its complement is applied to

the gate of the p-device. The operation of the transmission gate can be best explained by considering the characteristics of both the n-device and p-device as pass transistors individually. One can address this by treating the charging and discharging of a capacitor via a transmission gate. Comparing with NAND, NOR AND XOR gates, using transmission gates to build an adder reduces both the propagation delay time and the number of transistors.

High performance computer systems need both fast integer and floating-point operations. Floating-point operations require several cycles. The floating-point multiplier can operate directly on a normalized or wrapped number. The first stage of the multiplier contains a parallel multiplier, and a long bit adder is used for the second stage to execute a carry propagate addition[3]. A double precision floating point parallel multiplier based on the IEEE standard needs a long bit adder with more the 108-b[2].

However, as the number of adder sections increases, the carry-skip circuits become more costly and more complex. In large adders, bypass networks have been used to speed up the propagation signals. The problems of a conventional bypass circuit are conflicting signals. Since the two transmission gates output different signal levels in their transition phases, some node voltages are at intermediate voltage. This causes a long propagation delay time and wastes power.

A novel conflict - free bypass circuit has been introduced by T. Sato, M. Sakate, H. Okada, T. Sukemura, and G, Goto[2]. They use their conflict-free bypass circuit to speed up the propagation of bypassed signals of a ***112 - b*** transmission gate adder.

Although they have given the block diagrams for their circuit, they do not address exactly how that bypass network works, and they failed to specify how the bypass control signals control the adder. They give an example of the conflict - free control signal instead of the general way how to build the conflict free bypass system. In short, they do not want to explain their circuit in detail so that no one can implement their adder.

This research word is aimed at accomplishing the following:

1. Explanation of the basic circuit of the improved carry-skip adder.
2. Explanation of the general law of the conflict - free bypass signals.
3. Design the conflict-free bypass circuit for a ***16 - b*** adder and then design the conflict-free bypass circuit for a ***112 - b*** adder.
4. Implementation and layout design for a ***112 - b*** transmission gate adder with the conflict - free bypass circuit.
5. Simulation of the ***16 - b*** adder and the ***112 - b*** transmission gate adder.
6. Preparation for the fabrication this circuit with 64-pad package.
7. Comparison of the result obtained with previous published results.

CHAPTER 2

EXPLANATION OF CIRCUIT

2.1 Basic circuit

The addition of two binary numbers A and B can be obtained by means of the relations

$$C_{-1} = 0$$

$$SUM_i = A_i \oplus B_i \oplus C_{i-1} = P_i \oplus C_{i-1}$$

$$C_i = A_i B_i + P_i C_{i-1} = G_i + P_i C_{i-1}$$

Where

$P_i = A_i \oplus B_i$ [Propagate signal]

$G_i = A_i B_i$ [Generate signal]

C_i [Carry bit i]

In a simple carry ripple adder, the worst case delay is proportional to its size, because if a carry is generated, it will ripple through the entire structure.

If we divide the total number of bits into groups, the following rules apply to each group:

1. If each $A_i \neq B_i$ in a group, then we do not need to compute the new value of C_i for the block. The carry in of the block can be propagated directly to the next block.
2. If $A_i = B_i = 1$ for some i in the group, a carry is generated which may be propagated up to the output of that group.
3. If $A_i = B_i = 0$, a carry will not be propagated by that bit location.

The ***112-b*** adder here is divided into seven 16-b adder blocks. Fig. 1 shows the 1-b adder circuit diagram of 16-b adder[2][4][5].

There are four parts in the 1 - b adder of Fig. 1. We re-draw the circuit diagram on the right side. From the right side diagram, we can see the basic circuit unit for this adder is the two input multiplexer. A multiplexer function is formed by using the complementary switches to select between a number of inputs. As the switches have to pass ***zero***s and ***one***s equally well, complementary switches with n - and p - transistors are used. The truth table for the two input multiplexer is shown in Table 1. The complementary switch is also called a transmission gate or pass gate (complementary). A commonly used circuit symbols is two triangles as shown in Fig. 1.

TABLE 1 Two input multiplexer truth table

Input A	Input B	Control Signal S	Complementary of S	Output
X	0	0	1	0 (B)
X	1	0	1	1 (B)
0	X	1	0	0 (A)
1	X	1	0	1 (A)

The basic idea of a carry-skip adder is to assess if in each group all A_i do not equal B_i and enable the block's carry-in to skip the block when it happens. Our circuit works

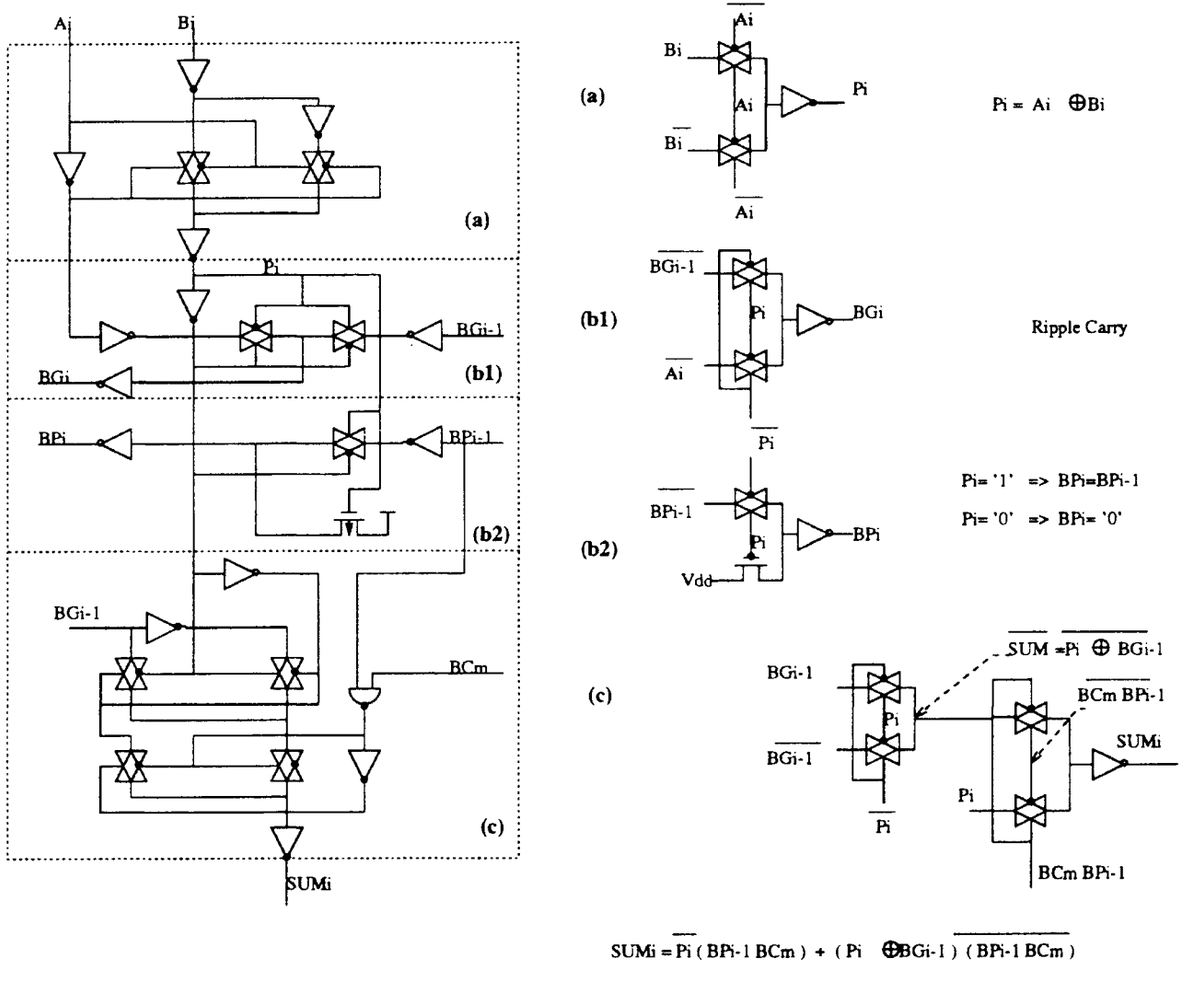


Fig. 1 One - bit Adder

like a carry select adder but with some difference. In a carry select adder, there are two summation signals and they are selected by block carry signals, thus wasting transistors. By contrast, the sum signal in this design is selected by logically multiplied block carry and block carry propagate signals based on the logical equation:

$$SUM_i = \overline{P_i} \bullet (BP_{i-1} \bullet BC_m) + (P_i \oplus BG_{i-1}) \overline{\bullet (BP_{i-1} \bullet BC_m)}$$

(m = 15,31,47,63,79,95,111).

Where: BP_i is a block-carry propagate signal, BG_i is a block-carry generate signal and BC_m is a block-carry signal. We will find the detail meanings of them from the following chapter.

Now we look back to the Fig. 1. In the part (a), two inputs of the multiplexer are Bi and complementary of Bi. And the control signal here is Ai. In this way, the out put of the multiplexer Pi = $A \oplus B$.

The signal Pi is very useful in the circuit. Not only both the carry propagate signal and the carry generate signal depend upon this signal Pi, but the complementary of Pi is also a signal which is directly sent out to the summation whenever both block propagate signal BPi and block carry signal BCm are ***one***s.

The part (b1) is also a multiplexer with previous block generate signal BGi-1 and one of the adder input, say Ai, as the two inputs. The control signal is Pi. The output of this part is current block generate signal. The output signal of the part (b1) is like the normal carry signal of the ripple adder.

The part (b2) of the Fig. 1 is not exactly a multiplexer, since there is only one pair of transmission gate with the other input connected to a p- transistor. When the

control signal Pi equals ***one***, the output signal BPi (block propagate signal) of this part equals the previous block propagate signal BPi-1. When Pi equals ***zero***, the output of this part is ***zero***. This means once the BPi becomes ***zero***, it will remain ***zero*** until the very end of this block.

The part (c) of this circuit consists two multiplexers. The output of the first multiplexer is a normal summation like a normal adder. The second multiplexer use the signal BPi-1BCm as the control signal. Just like the equation we addressed above, the summation of the adder will remain the same as the normal adder's summation except both BPi-1 and BCm are ***one***s. When that happened, i.e. BPi-1 equals BCm equals ***one***, the summation of this adder equals to the complementary of Pi.

2.2 Conflict-Free Bypass Circuit

The bypass circuit is added to propagate the carry signal as fast as possible. Since the carry signal of the conventional bypass circuit is generated by a wired-OR at some node, two transmission gates are activated at the same time. Then, a signal conflict is generated. One of the examples of the conventional bypass circuit used in the carry propagation path of a 4-b Manchester adder is shown in Fig. 2 (a). Fig. 2 (b) shows its waveform with all carry propagate signals (P_i's) equal to ***one*** and all carry generate signals (G_i's) equal to ***zero.*** A carry signal is generated by a wired-OR at the node S and two transmission gates are activated at the same time, a signal conflict is generated. Since the two transmission gates output different signal levels in their transition phases, the node S is at intermediate voltage. The dark areas on the Fig. 2(b) shows the intermediate voltage stage. This causes a long propagation delay time and wastes power.

Some other bypass circuits are used to solve the conflict problem. The paper [6] uses a NAND or NOR gate in place of a wired-OR. This circuit solves the power consumption problem, however, there are some transition phases in which the bypass circuit does not provide any performance improvement. Another type of bypass cicuit[2] has been used, but it only works when the signal changes from low to high, not when the signal changes from high to low.

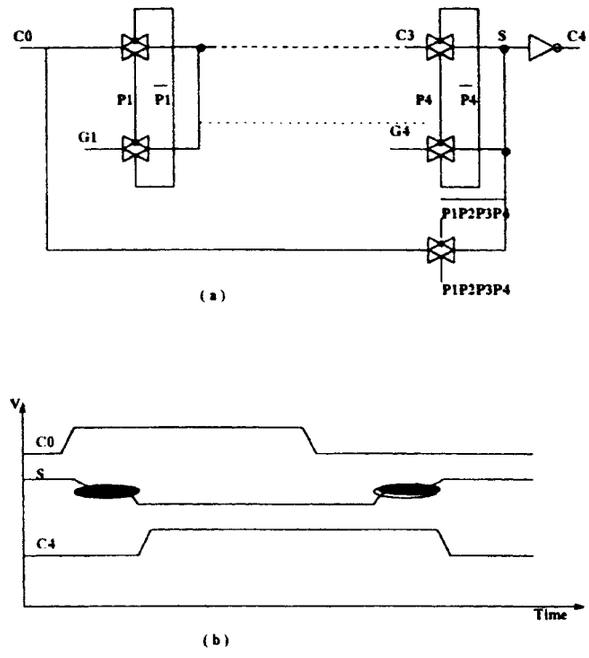


Fig. 2 (a) Manchester adder (wired-OR type)

(b) Waveform (with pi=1 and Gi=0, i=1,2,3,4)

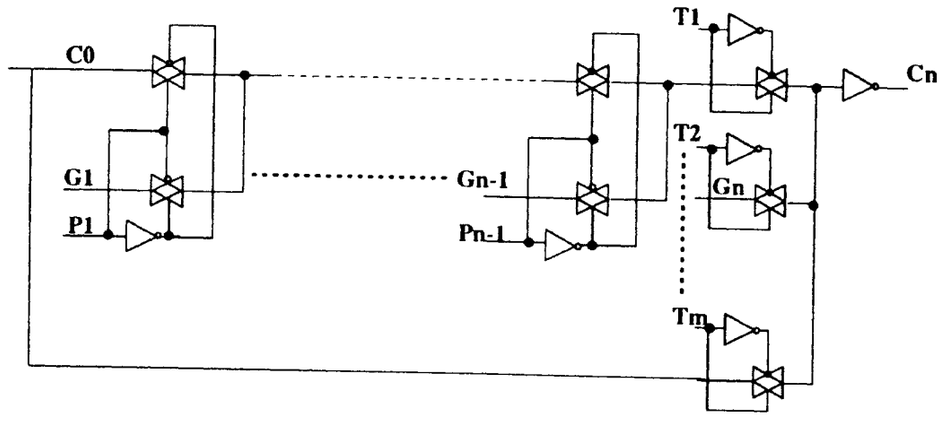


Fig. 3 Conflict - free circuit

The best method to solve this problem is the conflict - free bypass circuit shown in Fig. 3. A selector to generate the C_n signal is used to instead of a wired-OR , a NAND gate or a NOR gate. The transmission gates are controlled by some signals which are combined by propagate signals of the bypassed gates.

In the Fig. 3, ***n*** is the number of gates bypassed by the circuit, in other words, ***n*** represents how many signals are used to control the bypass circuit, and ***m*** is the number of gates used to implement the bypass function. Although the number ***m*** are normally less than or equal to the number ***n***, sometimes the number ***m*** could be one more than the number ***n***. In fact, ***m*** depends upon how many gates need to be bypassed, or is dependent upon ***n***. When ***n*** is getting bigger, ***m*** should become larger. However, considering the power consumption, ***m*** normally takes the number 2, 3 or 4 .

When ***n*** = 2, ***m*** = 3 :

$$T_1 = \overline{P_1} P_2$$
$$T_2 = \overline{P_2}$$
$$T_3 = P_1 P_2$$

In this case, there are two gates bypassed by the circuit. i.e. there are two signals are used to control the bypass circuit. And there are three gates used to implement the bypass function. When P_1 is equal ***zero*** and P_2 is equal to ***one***, then the gate T_1 is activated and T_2 and T_3 are opened. When P_2 is ***zero***, no matter P_1 equals to ***zero*** or ***one***, the gate T_2 turns to be activated and the gate T_1 and T_3 are open. When both P_1 and P_2 are ***one***s, the gate T_3 turned to be activated and T_1 remains open and T_2 switches to open. So, there is only one of three gates is activated to form the shortest path.

When $n = 3$, $m = 3$:

$$T_1 = \overline{(P_1 P_2)} P_3$$
$$T_2 = \overline{P_3}$$
$$T_2 = P_1 P_2 P_3$$

In this case, there are three gates bypassed by the circuit. i.e. there are three signals are used to control the bypass circuit. Also there are three gates used to implement the bypass function. When P_1 and P_2 are not equal to each other, and P_3 is ***one***, the gate T_1 is activated while T_2 and T_3 are opened. When P_3 is ***zero***, no matter P_1 and P_2 are equal to ***zero*** or ***one***, the T_2 is activated while T_1 and T_3 are opened. When all of the three gates are ***one***s, the gate T_3 is activated and T_1 remains open and T_3 turn to open. So, there is only one of three gates is activated to form the shortest path.

When $m = 4$, $n = 5$:

$$T_1 = \overline{(P_3 P_4)} P_5$$
$$T_2 = \overline{(P_1 P_2)} P_3 P_4 P_5$$
$$T_3 = \overline{P_5}$$
$$T_4 = P_1 P_2 P_3 P_4 P_5$$

In this case, there are five gates bypassed by the circuit. i.e. there are five signals are used to control the bypass circuit. And there are four gates used to implement the bypass function. When P_3 and P_4 are not equal to each other, and P_5 is ***one***, the gate T_1 is activated while T_2 , T_3 and T_4 are opened. When P_5 is ***zero***, no matter P_1, P_2, P_3 and P_4 are equal to ***zero*** or ***one***, the T_3 is activated while T_1, T_2 and T_4 are opened. When all of the five gates are ***one***s, the gate T_4 is activated and T_1 and T_2 remains open and T_3 turn to open. So, there is only one of five gates is activated to form the shortest path.

It is easy for the reader to find the rule of this bypass control signal. For example, if $m = 4$, and $n = 10$, then:

$$T_1 = \overline{(P_6P_7P_8P_9)}P_{10}$$
$$T_2 = \overline{(P_1P_2P_3P_4P_5)}P_5P_6P_7P_8P_9P_{10}$$
$$T_3 = \overline{P_{10}}$$
$$T_4 = P_1P_2P_3P_4P_5P_6P_7P_8P_9P_{10}$$

TABLE 2 shows how the bypass signals to control the circuit. Where we put some possible combinations of signal on the rows and the very right column of the table shows which gate is activated. As we have explained before, there is exactly one of those gate activates at one time to form the shortest path.

TABLE 2 The control signals and the activated gate

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Activated gate
1	0	0	0	0	0	0	0	0	0	T3
0	1	0	0	0	0	0	0	0	0	T3
0	0	1	0	0	0	0	0	0	0	T3
0	0	0	1	0	0	0	0	0	0	T3
0	0	0	0	0	1	0	0	0	0	T3
0	0	0	0	0	0	1	0	0	0	T3
0	0	0	0	0	0	0	1	0	0	T3
0	0	0	0	0	0	0	0	1	0	T3
0	0	0	0	0	0	0	0	0	1	T1
1	1	0	0	0	0	0	0	0	0	T3
0	1	1	0	0	0	0	0	0	0	T3
0	0	1	1	0	0	0	0	0	0	T3
0	0	0	1	1	0	0	0	0	0	T3
0	0	0	0	1	1	0	0	0	0	T3
0	0	0	0	0	1	1	0	0	0	T3
0	0	0	0	0	0	1	1	0	0	T3
0	0	0	0	0	0	0	1	1	0	T3
0	0	0	0	0	0	0	0	1	1	T1
1	1	1	0	0	0	0	0	0	0	T3

0	1	1	1	0	0	0	0	0	0	T3
0	0	1	1	1	0	0	0	0	0	T3
0	0	0	1	1	1	0	0	0	0	T3
0	0	0	0	1	1	1	0	0	0	T3
0	0	0	0	0	1	1	1	0	0	T3
0	0	0	0	0	0	1	1	1	0	T3
0	0	0	0	0	0	0	1	1	1	T1
1	1	1	1	0	0	0	0	0	0	T3
0	1	1	1	1	0	0	0	0	0	T3
0	0	1	1	1	1	0	0	0	0	T3
0	0	0	1	1	1	1	0	0	0	T3
0	0	0	0	1	1	1	1	0	0	T3
0	0	0	0	0	1	1	1	1	0	T3
0	0	0	0	0	0	1	1	1	1	T1
1	1	1	1	1	0	0	0	0	0	T3
0	1	1	1	1	1	0	0	0	0	T3
0	0	1	1	1	1	1	0	0	0	T3
0	0	0	1	1	1	1	1	0	0	T3
0	0	0	0	1	1	1	1	1	0	T3
0	0	0	0	0	1	1	1	1	1	T2
1	1	1	1	1	1	0	0	0	0	T3
0	1	1	1	1	1	1	0	0	0	T3
0	0	1	1	1	1	1	1	0	0	T3
0	0	0	1	1	1	1	1	1	0	T3
0	0	0	0	1	1	1	1	1	1	T2
1	1	1	1	1	1	1	0	0	0	T3
0	1	1	1	1	1	1	1	0	0	T3
0	0	1	1	1	1	1	1	1	0	T3
0	0	0	1	1	1	1	1	1	1	T2
1	1	1	1	1	1	1	1	0	0	T3
0	1	1	1	1	1	1	1	1	0	T3
0	0	1	1	1	1	1	1	1	1	T2
1	1	1	1	1	1	1	1	1	0	T3
0	1	1	1	1	1	1	1	1	1	T2
1	1	1	1	1	1	1	1	1	1	T4

This proposed bypass scheme can be extended for the network. In the control scheme, a transmission gate is opened when all the inner gates are activated and the outer gate is not activated. Exactly one of these signals activates to form the shortest

path. All of the other wired - OR nodes are controlled in a same way, so there is no signal conflict in this bypass network.

2.3 Design of the 16 - b adder

Since the ***112 - b*** adder is divided into seven ***16 - b*** adder blocks, we start here to explain the circuit function and bypass network for the ***16 - b*** adder.

The ***16 - b*** adder is basically formed by 16 ***1-b*** adder shown in Fig. 1, with the block - carry generate signals (BG_i) and block - carry propagate signals (BP_i) connected in the way shown in Fig. 4. This bypass network has four bypass units. The first one is for gate numbers 3, 4 and 5 , the second one is for the gate number 6, 7, 8, 9 and 10, the third one is for gate numbers 13 and 14, and the last one is for gate numbers 6 to 15. The bypass control signal for every unit is made according to the rule of the conflict-free bypass circuit which we mentioned before. For every bit in the carry - chain, there is a Ai signal as one of the input of the multiplexer. Whenever Pi equals ***zero***, the carry signal at that bit equals the Ai. In that way, we can save one gate used to implement the bypass function. In other words, if we still take ***n*** to represents how many signals are used to control the bypass circuit, and ***m*** to represents how many gates to implement the bypass function, then the number ***m*** should be less than or equal to the number ***n***.

The critical paths of this bypass network shown in Fig. 4 are chains of inverters and transmission gates through which the BG_i(Fig. 4 (a)) and BP_i (Fig. 4 (b)) signals are propagated. This multi - bypass network shortens the critical paths. In this network, from wherever the BG_i(or BP_i) signal starts to propagate, the path consists of four or less transmission gates.

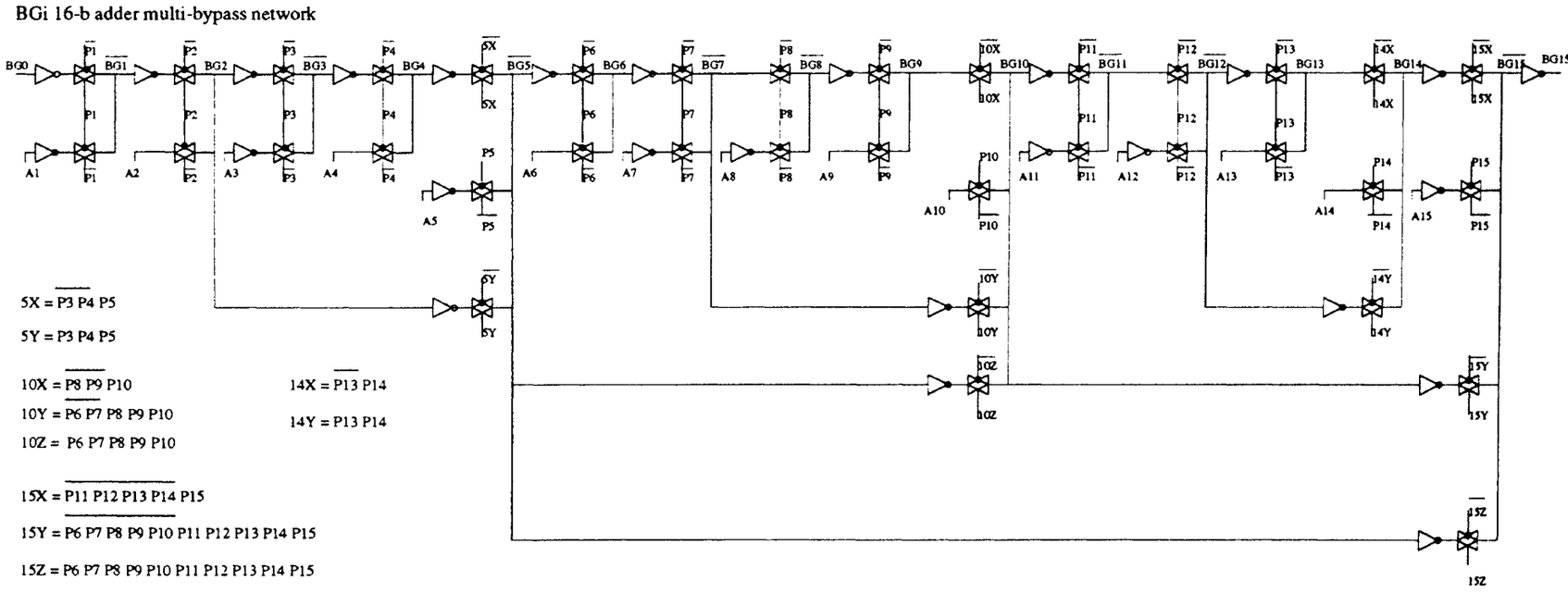


Fig. 4 (a) 16 - b BGi bypass network

BPn 16-b adder multi-bypass network

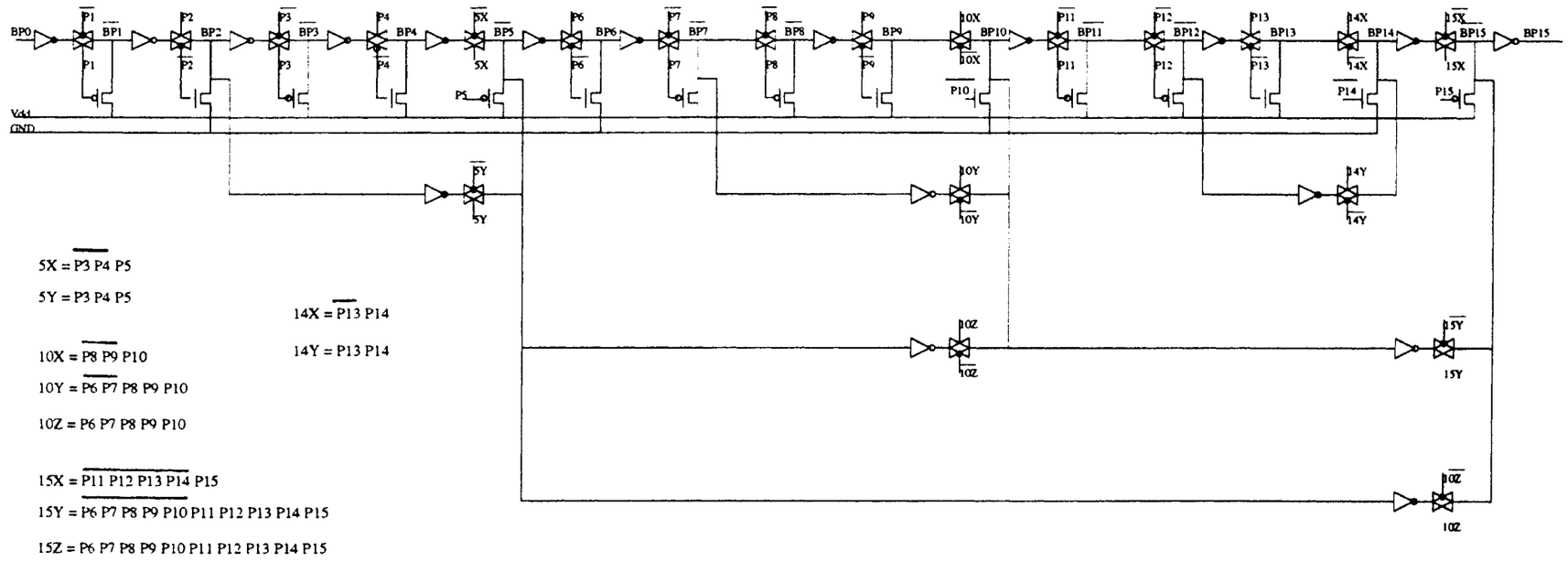


Fig. 4 (b) 16 - b BPi bypass network

From Fig. 4(a), we can see that control signals of the first bypass unit for the carry generate signal are:

$$5X = \overline{(P_3P_4)}P_5$$

$$5Y = P_3P_4P_5$$

That means there are two gates used to implement bypass function to bypass three gates in the circuits. Since this bypass unit is on the bit number 5, BG_5 equals A_5 if the P_5 is ***zero***. When P_5 equals to ***one***, BG_5 depends upon P_3 and P_4. Only when both P_3 and P_4 are ***one***s, is the gate 5Y activated with 5Y opened, otherwise the gate 5X is activated.

The control signals of the second bypass unit for the carry generate signal are:

$$10X = \overline{(P_8P_9)}P_{10}$$

$$10Y = \overline{(P_6P_7)}P_8P_9P_{10}$$

$$10Z = P_6P_7P_8P_9P_{10}$$

In this unit, there are three gates used to implement bypass function to bypass five gates in the circuits. Since this bypass unit is on the bit number 10, BG_{10} equals A_{10} if the P_{10} is ***zero***. When P_{10} equals to ***one***, BG_{10} depends upon P_6, P_7, P_8 and P_9. When P_8 and P_9 are not both ***one***s, the gate 10X is activated with the gates 10Y and 10Z opened. When P_6 and P_7 are not both ***one***s and all P_6, P_7, P_8 are ***one***s, the gate 10Y is activated with the gates 10X and 10Z opened. Only when all P_6, P_7, P_8 and P_9 are ***one***s, is the gate 10Z activated with 10X and 10Y opened.

The control signals of the third bypass unit for the carry generate signal are:

$$14X = \overline{P_{13}}P_{14}$$

$$14Y = P_{13}P_{14}$$

In this unit, there are two gates used to implement bypass function to bypass two gates in the circuits. Since this bypass unit is on the bit number 14, BG_{14} equals A_{14} if the P_{14} is ***zero***. When P_{14} equals to ***one***, BG_{14} depends upon P_{13}. When P_{13} equals ***zero***, the gate 14X is activated with the gate 10Y is opened. When P_{13} ***one***, the gate 14Y is activated with the gate 14X opened.

The control signals of the forth bypass unit for the carry generate signal are:

$$15X = \overline{(P_{11}P_{12}P_{13}P_{14})}P_{15}$$

$$15Y = \overline{(P_6P_7P_8P_9P_{10})}P_{11}P_{12}P_{13}P_{14}P_{15}$$

$$15Z = P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$$

In this unit, there are three gates used to implement bypass function to bypass ten gates in the circuits. Since this bypass unit is on the bit number 15, BG_{15} equals A_{15} if the P_{15} is ***zero***. When P_{15} equals to ***one***, BG_{15} depends upon $P_6 \sim P_{14}$. When $P_{11} \sim P_{14}$ are not all ***one***s, the gate 15X is activated with the gates 15Y and 15Z opened. When $P_6 \sim P_{10}$ are not all ***one***s and all $P_{11} \sim P_8$ are ***one***s, the gate 15Y is activated with the gates 15X and 15Z opened. Only when all $P_6 \sim P_{14}$ are ***one***s, is the gate 15Z activated with 15X and 15Y opened.

There are only 15 - b (1-15) of bypass network shown in Fig. 4. Actually there is one bit before the first bit with A_0 and B_0 as the input and BG_{-1}, BP_{-1} and BC_m as the carry in signals. For every ***16 - b*** adder block, BG_{-1} is always set to ***zero*** and BP_{-1} is always set to ***one***. However, the block carry signals BC_m depend upon the carry out

signals of the advanced adder block.. Inside the ***16 - b*** adder block all the BC_m's are connected. We have seven ***16 - b*** adder blocks in our ***112 - b*** adder, so the ***m*** here equals 15, 31, 47, 63, 79, 95 and 111 respectively for those seven blocks.

If the block carry signal BC_m equals ***zero***, that means there is no carry in signal generated by the advanced block, then $SUM_i = P_i \oplus BG_{i-1}$.

If BC_m equals ***one***, then SUM_i will depend upon the block propagate signal BP_i. When BP_i equals ***zero***, SUM_i will still remain the same as above, i.e. same as when the BC_m equals ***zero***. However, when BP_i equals ***one***, SUM_i $\overline{P_i}$.

Although the block carry signal BC_m remains the same within an adder block, the propagate signals BP_i are variant. BP_i depends upon P_i. When P_i equals ***zero***, BP_i equals ***zero***; when P_i equals ***one***, BP_i equals BP_{i-1}. Therefore, once BP_i changes from ***one*** to ***zero***, it will keep the ***zero*** until the end bit of the whole block.

2.4 Design of the 112 - b adder

Fig. 5 shows a block diagram of the adder, where seven ***16-b*** adder made that ***112-b*** adder. The block carry signal ($BC'm$) is generated by a block carry generator. Whenever the block carry generate signal of the last bit of 16 - b adder (BGm) equals ***one***, $BC'm$ equals ***one***. If BGm equals ***zero*** but both BPm and BC'_{m-16} are ***one***s, then the $BC'm$ also equals ***one***. So,

$$BC'_m = BG_m + BP_m \bullet BC'_{m-16}$$

(m = 15, 31, 47,63,79,95 and 111)

Table 3 uses some examples to show the relationship of A_i, B_i, P_i, BG_i, BP_i, C_i (carry), BC'_m and SUM_i. For convenience, here take 4 bits as a block. Where the A_i and B_i are two inputs of the adder(s). There are 4 pairs of inputs signals for every one of the 4 bit adder block. The least significant bit on the left. Propagate signals $P_i = A_i \oplus B_i$. The

Table 3 ***The Block Adder Truth Table***

	Block #1	Block #2	Block #3	Block #4	Block #5	Block #6	Block #7
A i	1 0 1 0	0 1 0 1	0 1 1 1	1 1 0 1	1 1 1 0	1 0 0 1	1 0 1 0
B i	0 0 1 1	1 0 1 0	1 0 1 1	1 1 0 1	1 0 0 0	0 1 1 0	0 1 1 1
P i	1 0 0 1	1 1 1 1	1 1 0 0	0 0 0 0	0 1 1 0	1 1 1 1	1 1 0 1
SUM(block)	1 0 0 0	1 1 1 1	1 1 0 1 *1*	0 1 1 0 *1*	0 0 0 1	1 1 1 1	1 1 0 0 *1*
BG i	0 0 0 1 1	0 0 0 0 0	0 0 0 1 1	0 1 1 0 1	0 1 1 1 0	0 0 0 0 0	0 0 0 1 1
BP i	1 1 0 0 0	1 1 1 1 1	1 1 1 0 0	1 0 0 0 0	1 0 0 0 0	1 1 1 1 1	1 1 1 0 0
C i	0 0 0 1 1	1 1 1 1	1 1 1 1	1 1 0 1	1 1 1 0	0 0 0 0	0 0 1 1
SUM i	1 0 0 0	0 0 0 0	0 0 1 1	1 1 1 0	1 0 0 1	1 1 1 1	1 1 0 0 *1*

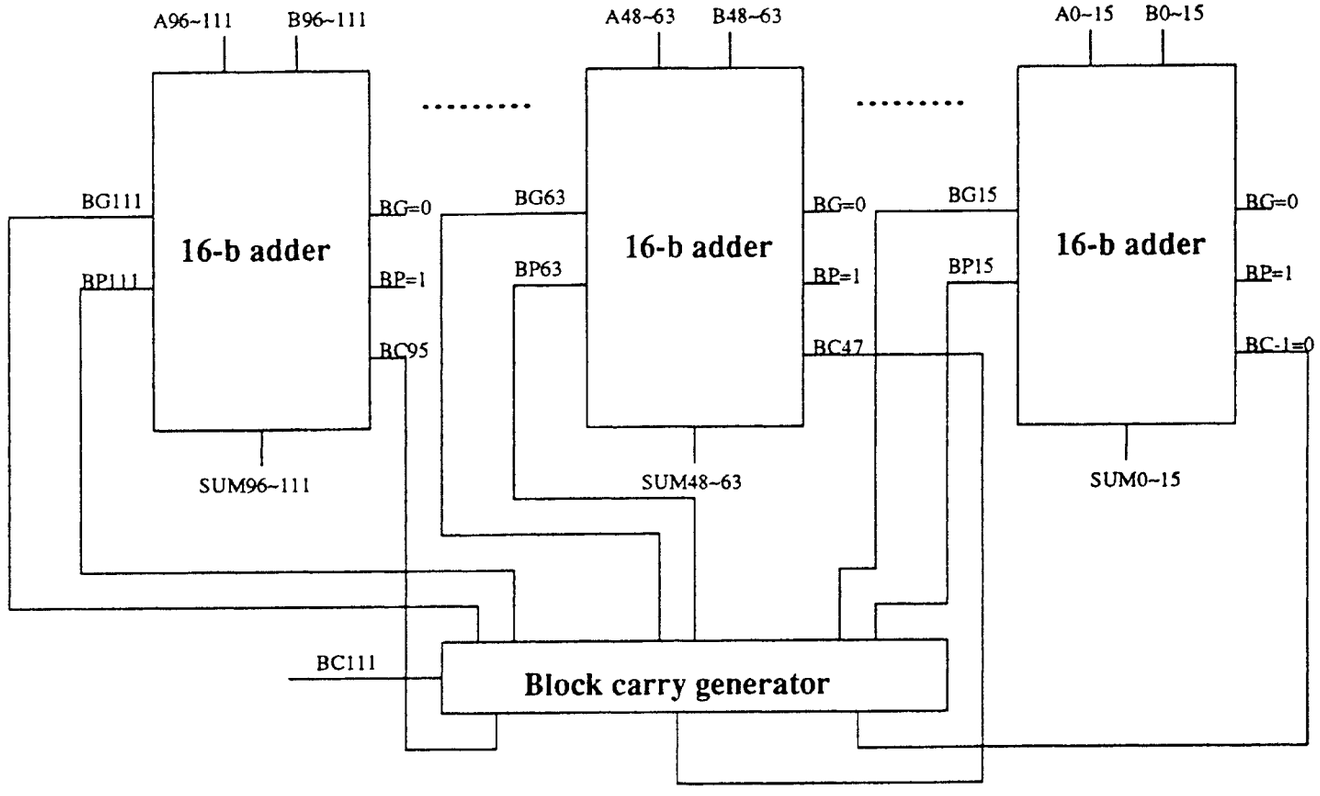


Fig. 5 Block Diagram

$SUM_{(block)}$s are the summations inside the blocks. On the right side of the block #3, #4 and #5, there are one more bit which mean at those blocks there is a carry out signal on that most significant bit of that adder block. At the beginning of every block, the block carry generate signals BG_i were set to ***zero*** and the block carry propagate signals BP_i were set to ***one***. Therefore, there are 5 bits numbers for the carry general signal BG_i and the carry propagate signal BP_i. Both BG_i and BP_i depend on the propagate signal P_i. Whenever P_i equals ***one***, $BG_i = BG_{i-1}$ and $BP_i = BP_{i-1}$. While P_i equals ***zero***, BG_i equals A_i and BP_i equals ***zero***. The C_i's show if there have carry out signals on that bit or not. The SUM_i 's show the summations of the whole adder which combined the block adders together. Comparing $SUM_{(block)}$ and SUM_i, we can find whenever $SUM_{(block)}$ signal and SUM_i are not same, that should be at the situation that both C_m and BP_{i-1} are ***one***, so SUM_i P_i.

Fig. 6 shows the block carry generator and the bypass network. The bypass control scheme is the same as in the BG_i (or BP_i) network, so there is no signal conflict in this bypass network.

There are seven ***16 - b*** adder blocks in the ***112 - b*** adder with three bypass units. The three bypass units are on the bit number 63 (the forth block), the bit number 79 (the fifth block) and the bit number 95 (the sixth block).

The control signals of the first bypass unit for the ***112 - b*** adder are:

$$63X = \overline{BP_{47}} BP_{63}$$
$$63Y = BP_{47} BP_{63}$$

BCm Block carry generator and the multi-bypass network

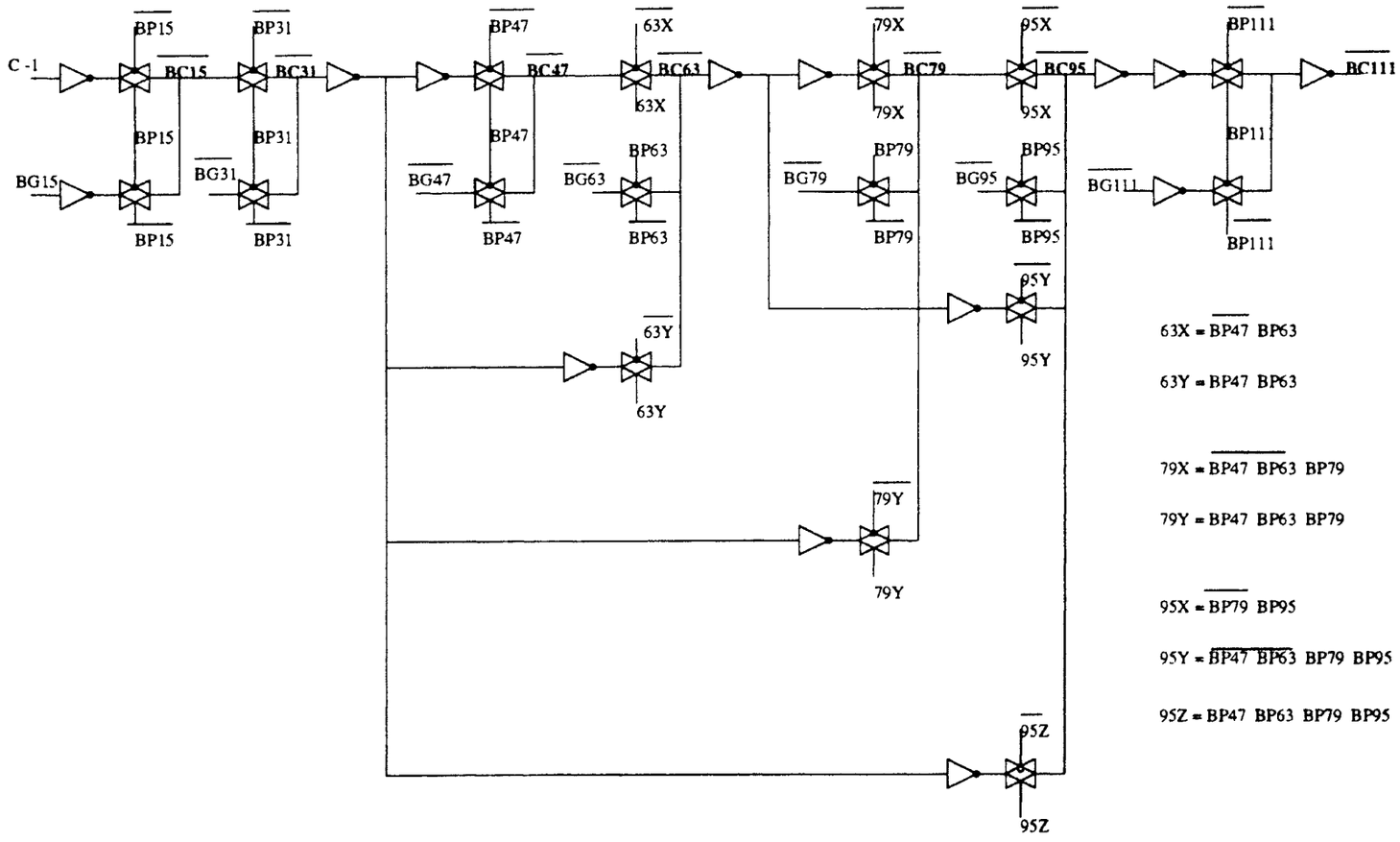


Fig. 6 Block carry generator and bypass network for the 112 - b adder

In this unit, there are two gates used to implement bypass function to bypass two blocks in the circuits. Since this bypass unit is on the bit number 63, the block carry signal BC_{63} equals BG_{63} if the BP_{63} is ***zero***. When the BP_{63} is ***one***, BC_{63} depends upon BP_{47}. When BP_{47} is equal to ***zero***, the gate 63X is activated. When BP_{47} is equal to ***one***, the gate 63Y is activated.

The control signals of the second bypass unit for the ***112 - b*** adder are:

$$79X = \overline{BP_{47}BP_{63}}BP_{79}$$
$$79Y = BP_{47}BP_{63}BP_{79}$$

In this unit, there are two gates used to implement bypass function to bypass three blocks in the circuits. Since this bypass unit is on the bit number 79, the block carry signal BC_{79} equals BG_{79} if the BP_{79} is ***zero***. When the BP_{79} is ***one***, BC_{79} depends upon BP_{47} and BP_{63}. When both BP_{47} and BP_{63} are equal to ***one***, the gate 79Y is activated. Otherwise the gate 79X is activated.

The control signals of the third bypass unit for the ***112 - b*** adder are:

$$95X = \overline{BP_{79}}BP_{95}$$
$$95Y = \overline{BP_{47}BP_{63}}BP_{79}BP_{95}$$
$$95Z = BP_{47}BP_{63}BP_{79}BP_{95}$$

In this unit, there are three gates used to implement bypass function to bypass four blocks in the circuits. Since this bypass unit is on the bit number 95, the block carry signal BC_{95} equals BG_{95} if the BP_{95} is ***zero***. When the BP_{95} is ***one***, BC_{79} depends upon BP_{47}, BP_{63} and BP_{79}. When BP_{79} is ***zero*** , no matter BP_{47} and BP_{63} are ***zero***s or ***one***s, the

gate 95X is activated. When not both BP_{47} and BP_{63} are equal to ***one***, BP_{79} is ***one***, the gate 95Y is activated. When BP_{47}, BP_{63} and BP_{79} are ***one***s, the gate 95Z is activated. As we explained before, there is only one of the three gates is activated to form the shortest path. There are no signal conflicts in the bypass network.

CHAPTER 3

LAYOUT DESIGN

The layout was designed by a fully manual layout method to decrease the propagation delay time, and is based on a 2-μm CMOS design rules for a double metal layout process.

The major effort of this research is fighting for the optimal layout design so that the adder could be as fast as possible. The distance between the n diffusion and p diffusion regions of all transistors are the minimum size. The connections between the transistors are also as short as possible.

Fig. 7 shown the layout design of the ***1 - b*** adder. The area of the layout design is 150x180 μm^2.

Fig. 8 shows the layout design of the ***16-b*** adder block with the ***bypass circuit*** under the 16 ***1-b*** adders. The area of this ***16 - b*** adder layout is 400x1700 μm^2. The left side of the graph is the least significant bit.

Fig. 9 shows the layout design of the ***112-b*** adder with seven ***16-b*** adder blocks overlapped and the ***bypass circuit*** on the left side. The area of this 112 - b adder layout is 2750x1750 μm^2. The bottom right corner is the least significant bit and the top left corner is the most significant bit.

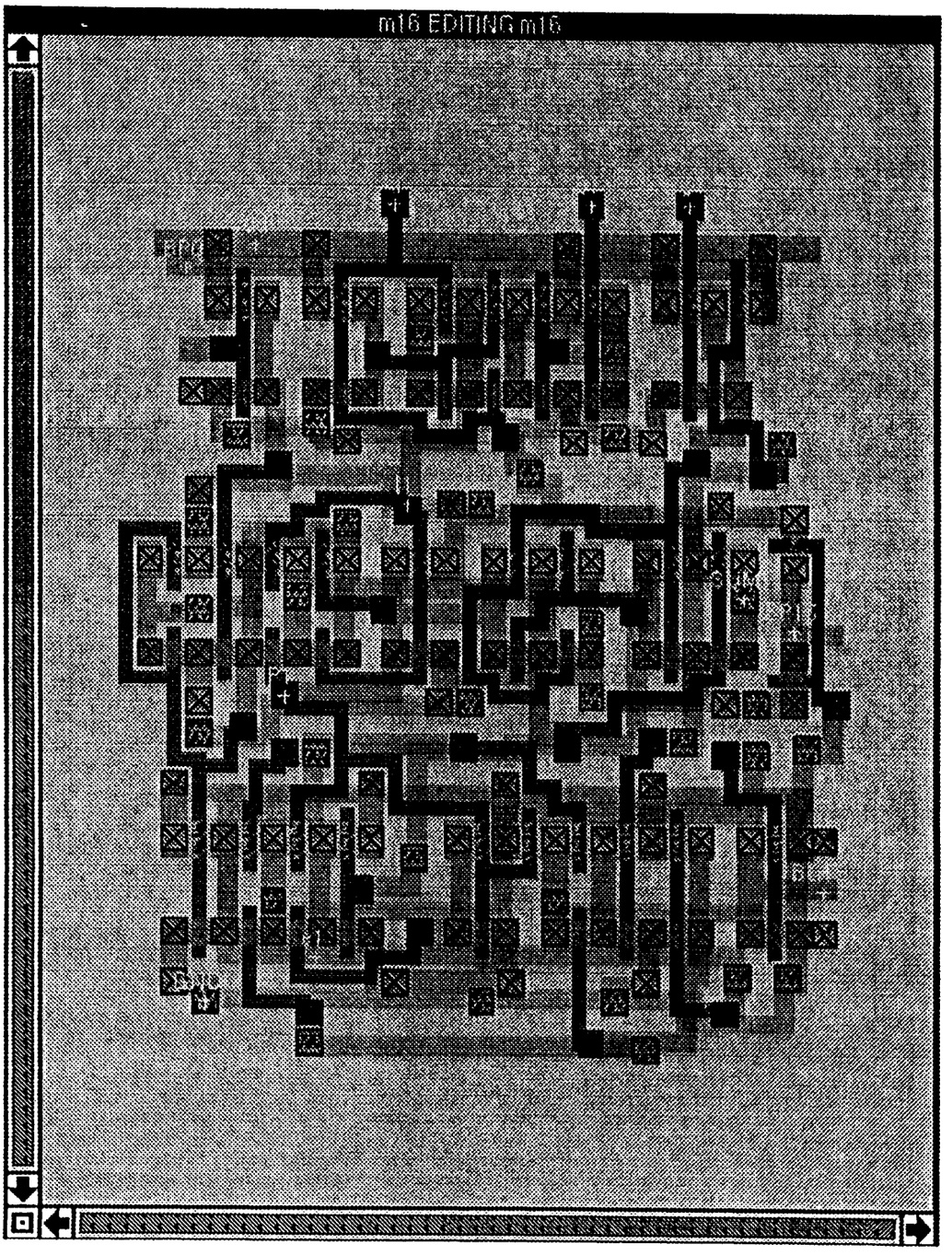


Fig . 7 Layout Design of the 1 - b Adder

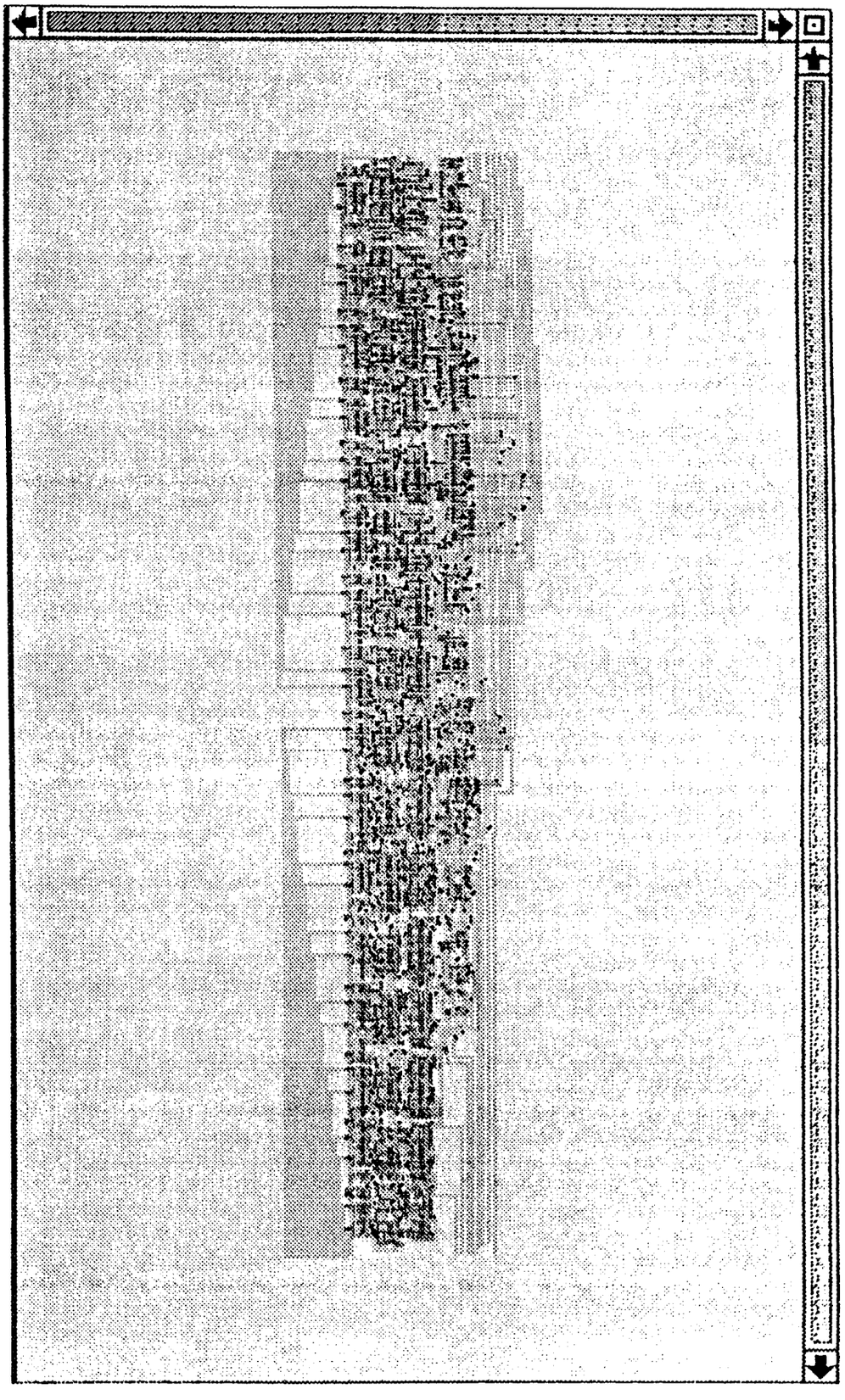


Fig. 8 The layout design of the 16 - b adder

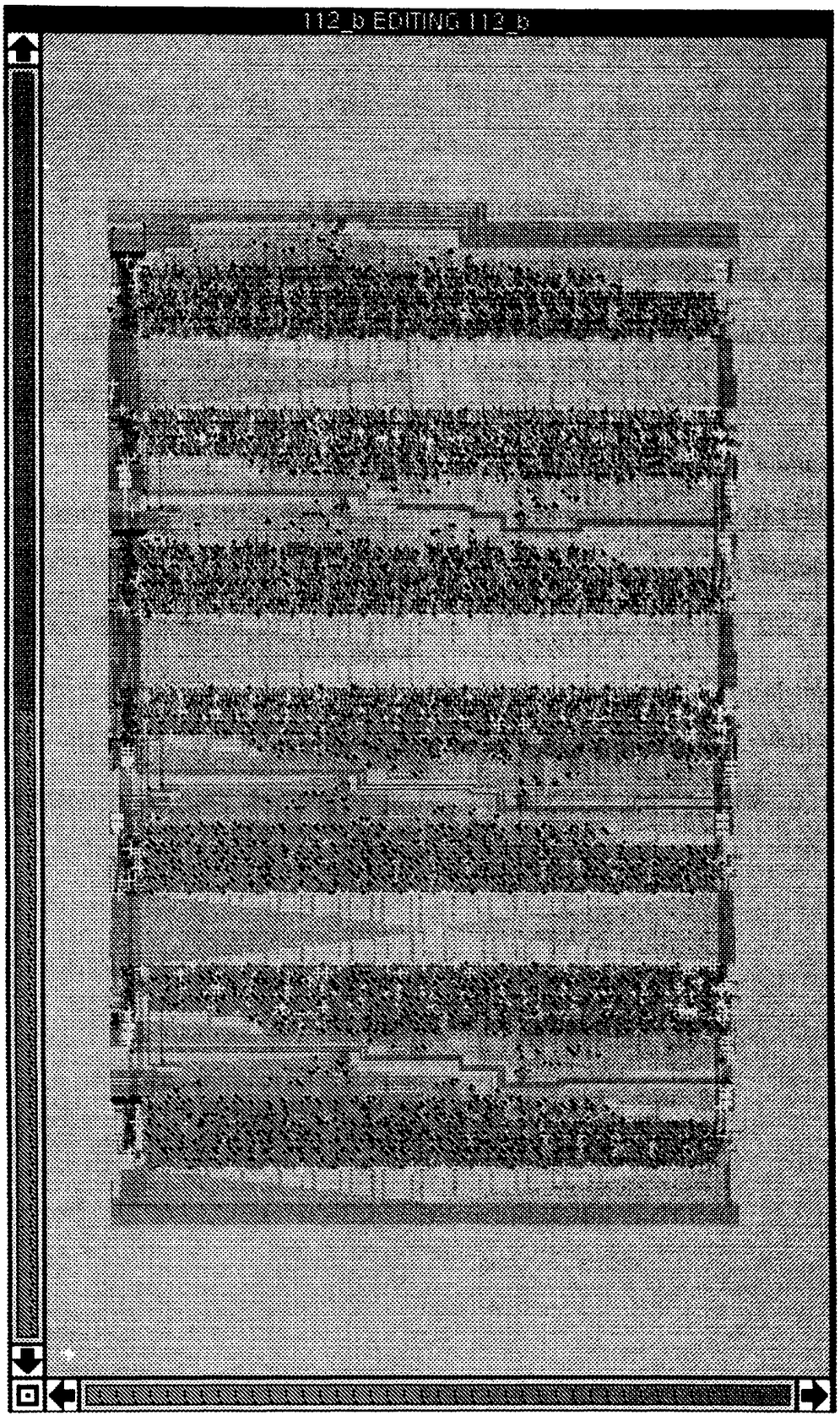


Fig. 9 The layout design fo the 112 - b adder

3.1 The area of the transistors

Once the basic layout has been determined, some optimization of transistor sizing may take place. This is only necessary if after simulation the adder is found to be lacking in speed.

According to the rule for the 2 - μm CMOS technology, the minimum area of metal contact is 4x4 μm^2. The minimum area of the transistor gate is 2x4 μm^2. In order to figure out the optimum area of the transistors which results in the fastest circuit, we design several different transistor gate area. We found the minimum area for both P - channel and n - channel transistors does not give the fastest result. If we change the gate area of p - channel transistors to 2x6 μm^2 and remain the gate area of n - channel transistors to 2x4 μm^2, the delay of the ***16 - b*** adder can be reduced 20%.

3.2 Discussion about the inverters

As the carry out signal is used in the generation of summation, the summation signals will be delayed with respect to carry out signal. In the circuit of the ***112 - b*** adder, every block generate signal and the block propagate signal of each bit has an inverter at input and an inverter at output. To optimize the carry delay, the inverters of every other bit should be omitted. But in the practical layout design, simply taking off the inverter does not always help to speed up the circuit. If one inverter drives more than 3 transistors, it will slow down the circuit speed. Comparing the speed with or without inverters, we conclude here with: taking the inverters out gave more delay than leaving them in.

3.3 About compound gates control signals

In the bypass control signals of 112 - b transmission gate adder, there are several control signals which are generated from logic explanations with many inputs, such as the control signals for the gate 10Z, 15Y, and 15Z. If we use signal compound gates designed exactly as the equation, that should slow down the circuit. If we divide the logic gates into several parts, and for every part there is less than 4 input signals, the speed of the circuit increases significantly. For example, instead of

$$15Z = P_6P_7P_8P_9P_{10}P_{11}P_{12}P_{13}P_{14}P_{15}$$

we use :

$$15Z = \overline{\overline{P_6P_7P_8} + \overline{P_9P_{10}P_{11}P_{12}} + \overline{P_{13}P_{14}P_{15}}}$$

In this way, we can increase the circuit speed by 11%.

CHAPTER 4

SIMULATION RESULTS

The irsim program running on the Sun work stations has been used to simulate the performance of our transmission gate adder layout since it is a good simulating method for the large circuit.

4.1 simulation results of the 16 - b adder

The ***16-b*** adder has four bypass units on the fifth, the tenth, the fourteenth and the fifteenth bits. Those bypass units only work when not every propagate signal P_i equals ***zero***. In other words, if all P_i's are ***zero***s, then the bypass network is not needed at all, since whenever P_i is ***zero***, BG_i equals A_i. Therefore, there are no carry signals to propagate through the circuits. Fig. 10 shows the simulation results for the ***16-b*** adder where all P_i's equal ***zero***s. There are two steps on the Fig. 10, all A_i and B_i are ***one***s on the first step and all A_i and B_i are ***zero***s on the second step. So, all the propagate signals P_i are ***zero***s on both steps. The delay for the ***16 - b*** adder is 9.8 ns. Note, the number in the middle of the top line represents the delay of the circuit, which is shown by a dotted line in the figure. In fact, no matter how many bits in the adder, this delay will remain the same as long as the P_i's are ***zero***s.

Fig. 11 shows the simulation results of ***16 - b*** adder for some certain input data. For this input data, the P_4, P_5, P_8, P_{10}, P_{14} and P_{15} are ***one***s, the bypass gates: 5X

$((\overline{P_3P_4})P_5)$, 10X $((\overline{P_8P_9})P_{10})$, 14X $(\overline{P_{13}}P_{14})$ and 15X$(\overline{(P_{11}P_{12}P_{13}P_{14})}P_{15})$ are activated. The delay for the ***16 - b*** adder is 13 ns. Note, the number in the meddle of the top line represents the delay of the circuit, which is shown by a dotted line in the figure. The simulation results of Fig. 11 are clearly presented in Table 4.

Table 4 ***Truth table for Fig. 11***

Bit number (i)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Ai	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bi	1	1	1	1	0	0	1	1	0	1	0	1	1	1	1	0
Pi	0	0	0	0	1	1	0	0	1	0	1	0	0	0	0	1
SUMi	0	1	1	1	0	0	1	1	0	1	0	1	1	1	0	0

In the table 4, the least significant number bit is A_0 on the left of the table. Pi = Ai ⊕ Bi, and SUMi's are the summations of the input A_i and B_i.

Fig. 12 shows the simulation results of the ***16 - b*** adder for another group of input data. In this case, P_0~P_6 are ***zero***'s, P_7~P_{15} are ***one***'s. So, the bypass gates: 10Y $((\overline{P_6P_7})P_8P_9P_{10})$ and 15Y$(\overline{(P_6P_7P_8P_9P_{10})}P_{11}P_{12}P_{13}P_{14}P_{15})$ are activated. Since A_0 ~ A_6 and B_0 ~ B_6 are ***one***'s and A_7 ~ A_{15} and B_7 ~ B_{15} are ***zero***'s, SUM_1 ~ SUM_6 are ***one***'s while other SUM_i's bits are ***zero***'s delay of the ***16 - b*** adder is 18.9 ns. . Note, the number in the middle of the top line represents the delay of the circuit, which is shown by a dotted line in the figure.

When all propagate signals P_i's are ***one***s, the worst delay results. Since at this case, the carry generate signal will propagate through all of the circuit. By using the bypass gates: 5Y$(P_3P_4P_5)$, 10Z$(P_6P_7P_8P_9P_{10})$, 14Y$(P_{13}P_{14})$ and 15Z$(P_6P_7P_8P_9P_{10}P_{11}P_{12}$

$P_{13}P_{14}P_{15}$), the worst delay for the 16 - b adder is 19.7 ns. The simulation results are shown on Fig. 13. Note, the number in the middle of the top line represents the delay of the circuit, which is shown by a dotted line in the figure.

4.2 simulation results of the 112 - b adder

The ***112 - b*** adder is combined from seven ***16 - b*** adder blocks with block carry generator. Fig. 14 shows the simulation results for the case when all 112 P_i's are ***zero***s. Just as we explained before, no bypass gates are needed here and the delay is still about 10 ns all the way to the end bit in this case, since there are no carry signals to propagate through the circuits. Note, the number in the middle of the top line represents the delay of the circuit, which is shown by a dot line on the figure.

Now we are ready to test the delay time for ***112 - b*** adder which used the bypass gates. Although the initial block-carry generate signals BG_{-1} and the initial block-carry propagate signals BP_{-1} are same in each of the seven ***16 - b*** adder blocks, the block carry signals BC_m's are different. Since the BC_m's depend upon the output block-carry generate and propagate signals (BG_{15} and BP_{15}) of previous block, there should be some undefined stage at the node of BC_m while the signals propagate from BG_{-1} (or BP_{-1}) to BG_{15} (or BP_{15}). From Fig. 1, we can find there some transmission gates between BG_i (or BP_i) and BC_m. Note the ***irsim*** could not predict the real propagate direction of the signals. So, when we use the ***irsim*** to simulate the circuit, the undefined signals could send some undefined feedback signals to BG_i (or BP_i), that will give the incorrect results although this case will not happen in the real circuit . In order to avoid this situation, we use two steps to test the circuits: first make all P_i equal ***zero***s, then send the input signals on the second step to test the delay time.

There are three block bypass units in the ***112 - b*** adder. If the input data shown on Fig. 11 were used on the third ***16 - b*** adder block, then the block propagate signal BP_{47} is ***zero***. In this way, the bypass gates 63X($\overline{BP'_{47}BP'_{63}}$). 79X($\overline{(BP'_{47}BP'_{63})}BP'_{79}$) and 95Y($\overline{(BP'_{47}BP'_{63})}$) are activated. Fig. 15 shows the simulation results. The delay for the 112 - b adder in 26.8 ns. Note the delay of the circuit equals the number in the middle of the top line (shown by a dotted line in the figure) minus 100 ns, since the input signals are changed at 100 ns.

The worst delay of the ***112 - b*** adder is in the case where all the propagate signals (P_i) are ***ones*** and the carry-in signal for all circuit C-1 equals ***one***. That means all the inputs A_i's are different from inputs B_i's. Since at this case, the carry generate signal will propagate through all circuit. We set all A_i to ***one*** and all inputs B_i's to ***zero***, the simulation results for this case are shown on Fig. 16. The delay for the ***112 - b*** adder is 27.5 ns. Note the delay of the circuit equals the number in the middle of the top line (shown by a dotted line in the figure) minus 100 ns, since the input signals are changed at 100ns.

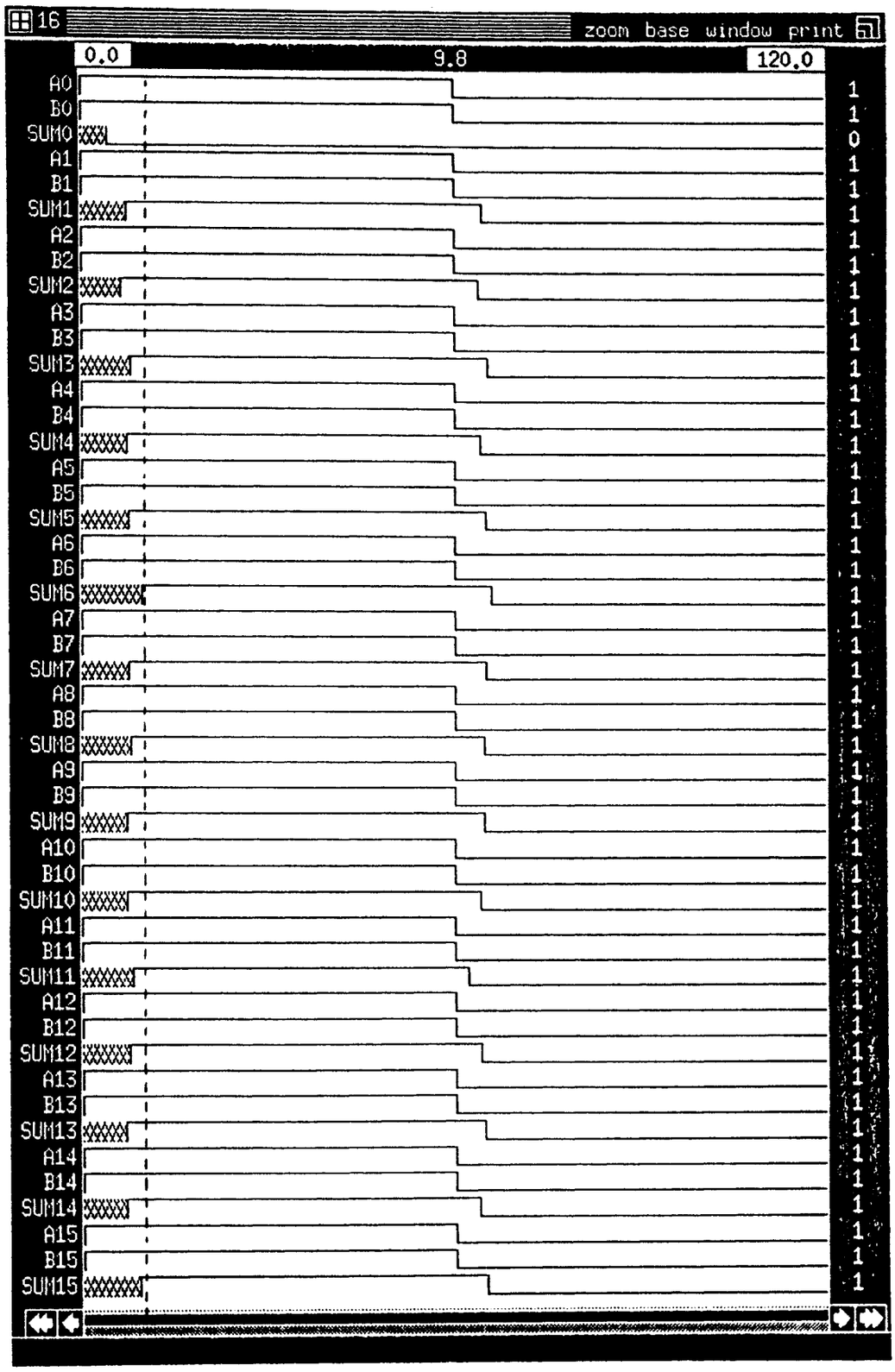


Fig. 10 The simulation results of the 16 - b adder (a)

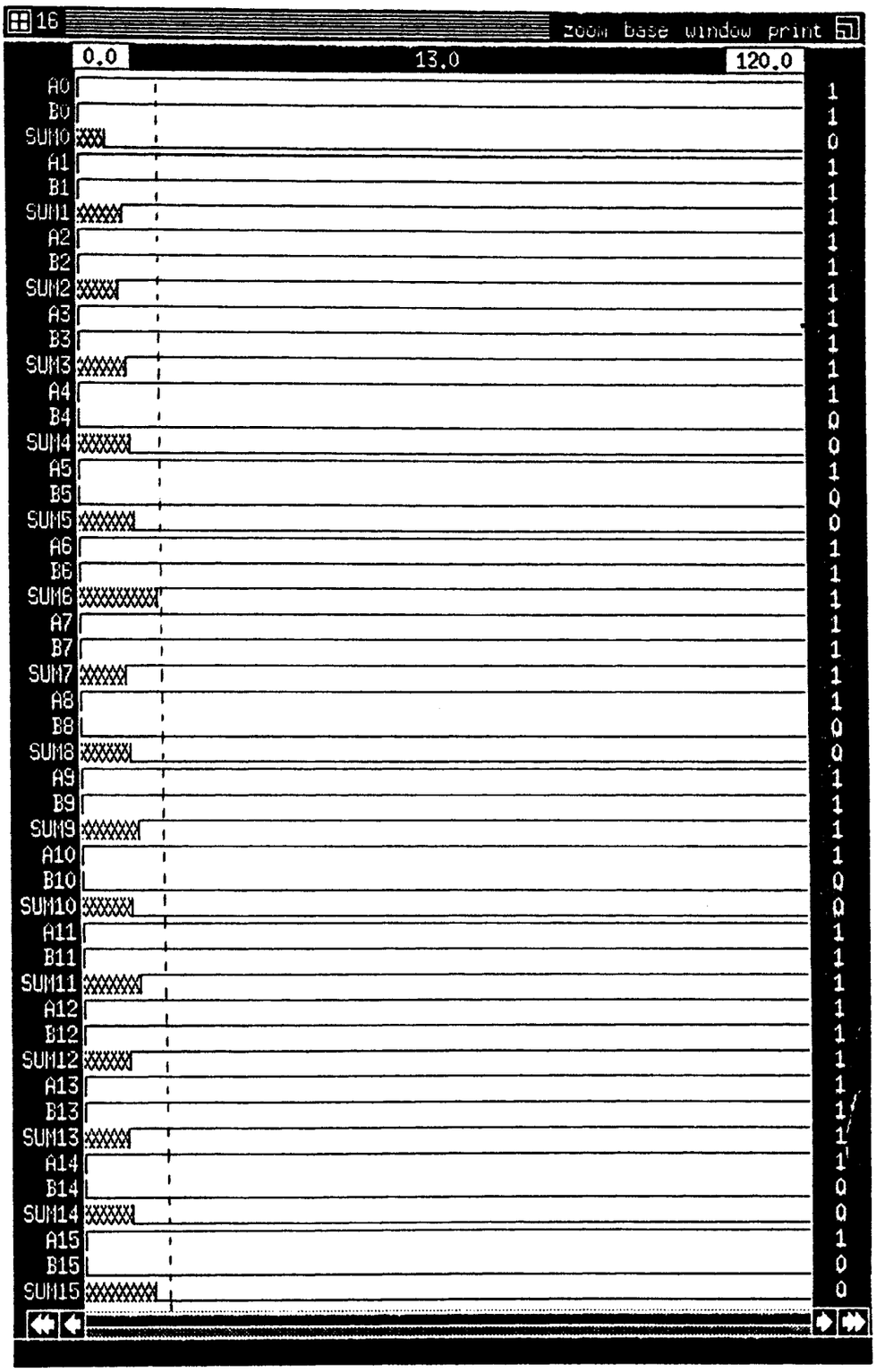


Fig. 11 The simulation results of the 16 - b adder (b)

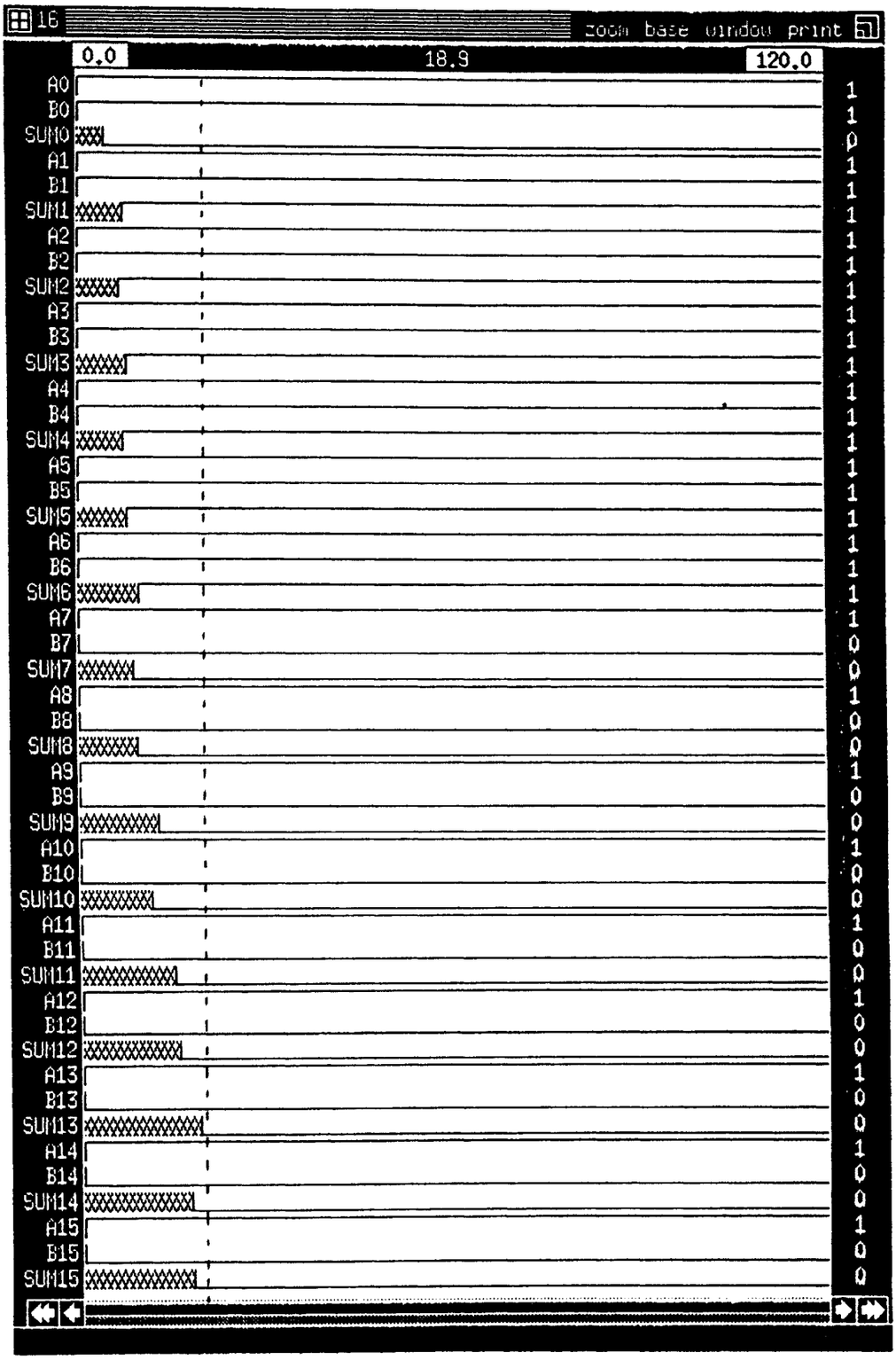


Fig. 12 The simulation results of the 16 - b adder (c)

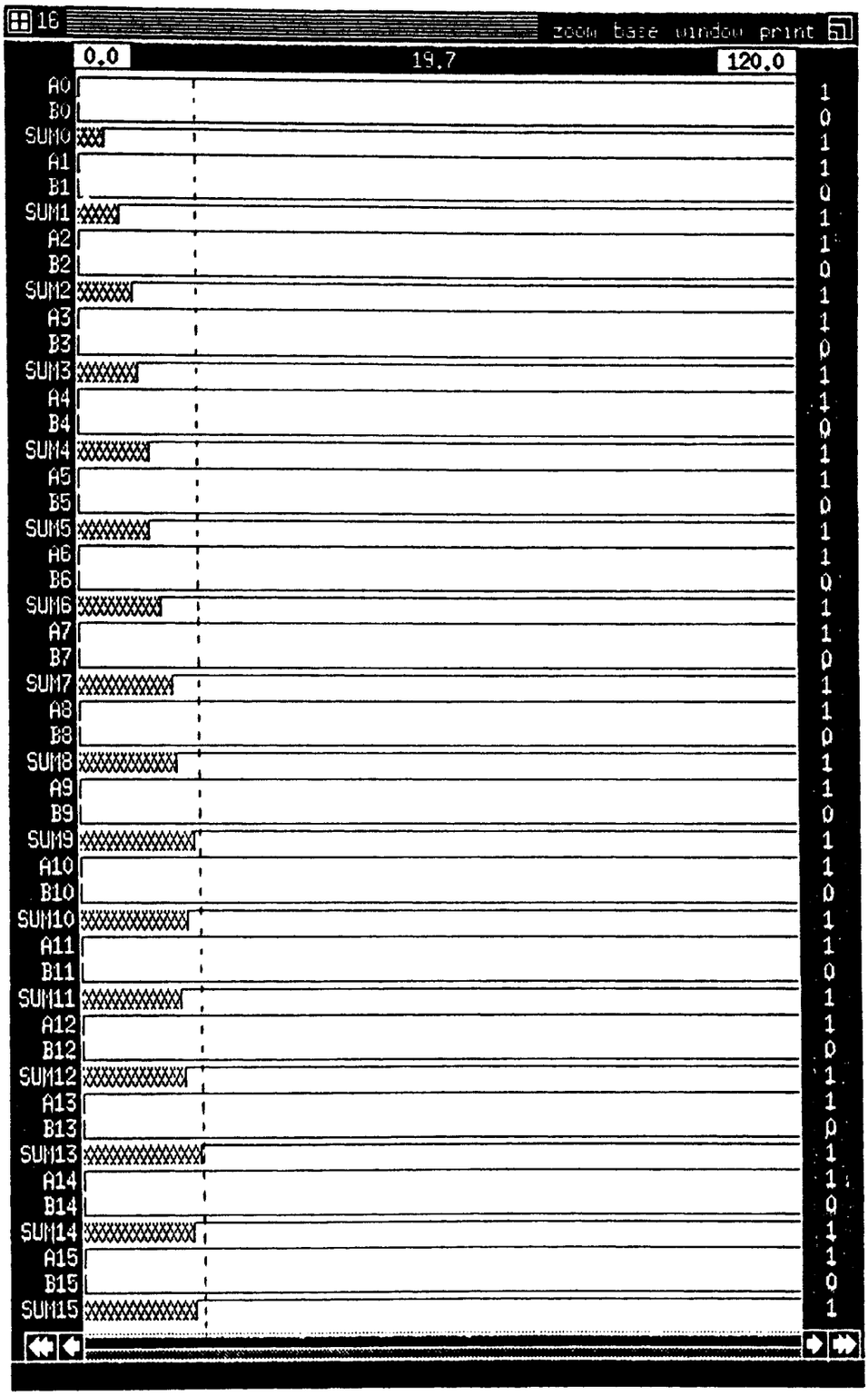


Fig. 13 The simulation results of the 16 - b adder (d)

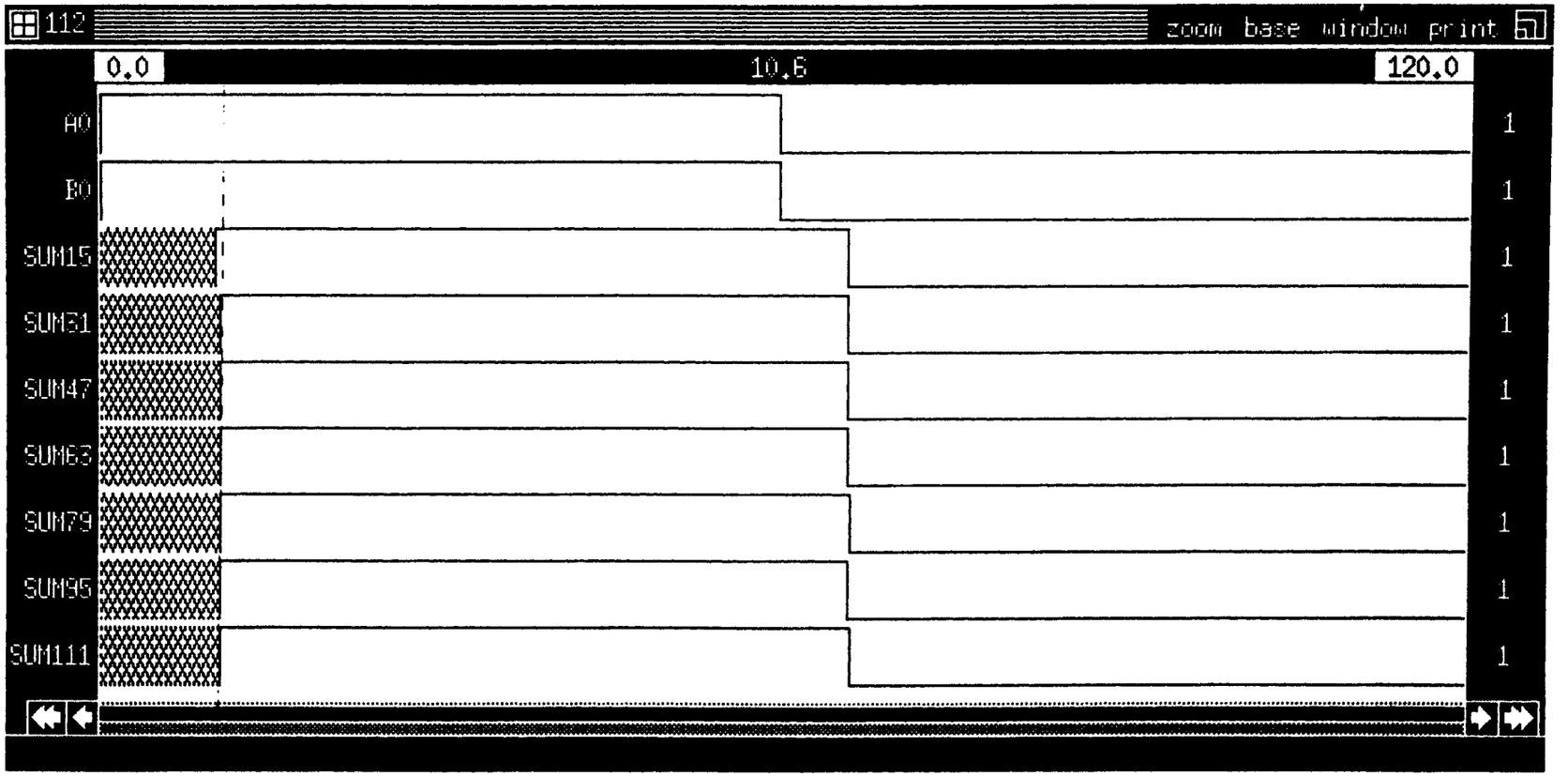


Fig.14 The simulation results of the 112 - b adder (a)

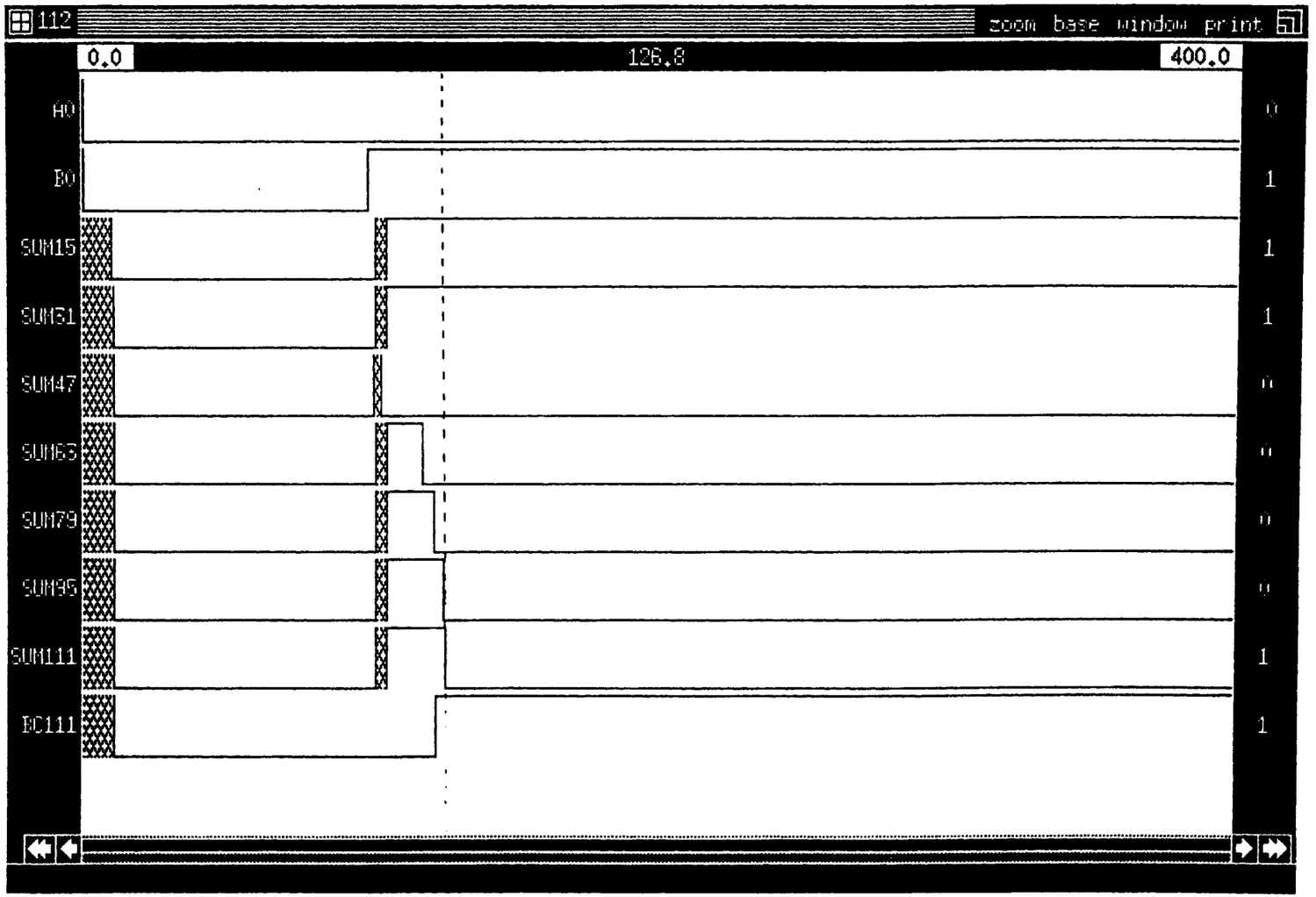


Fig.15 The simulation results of the 112 - b adder (b)

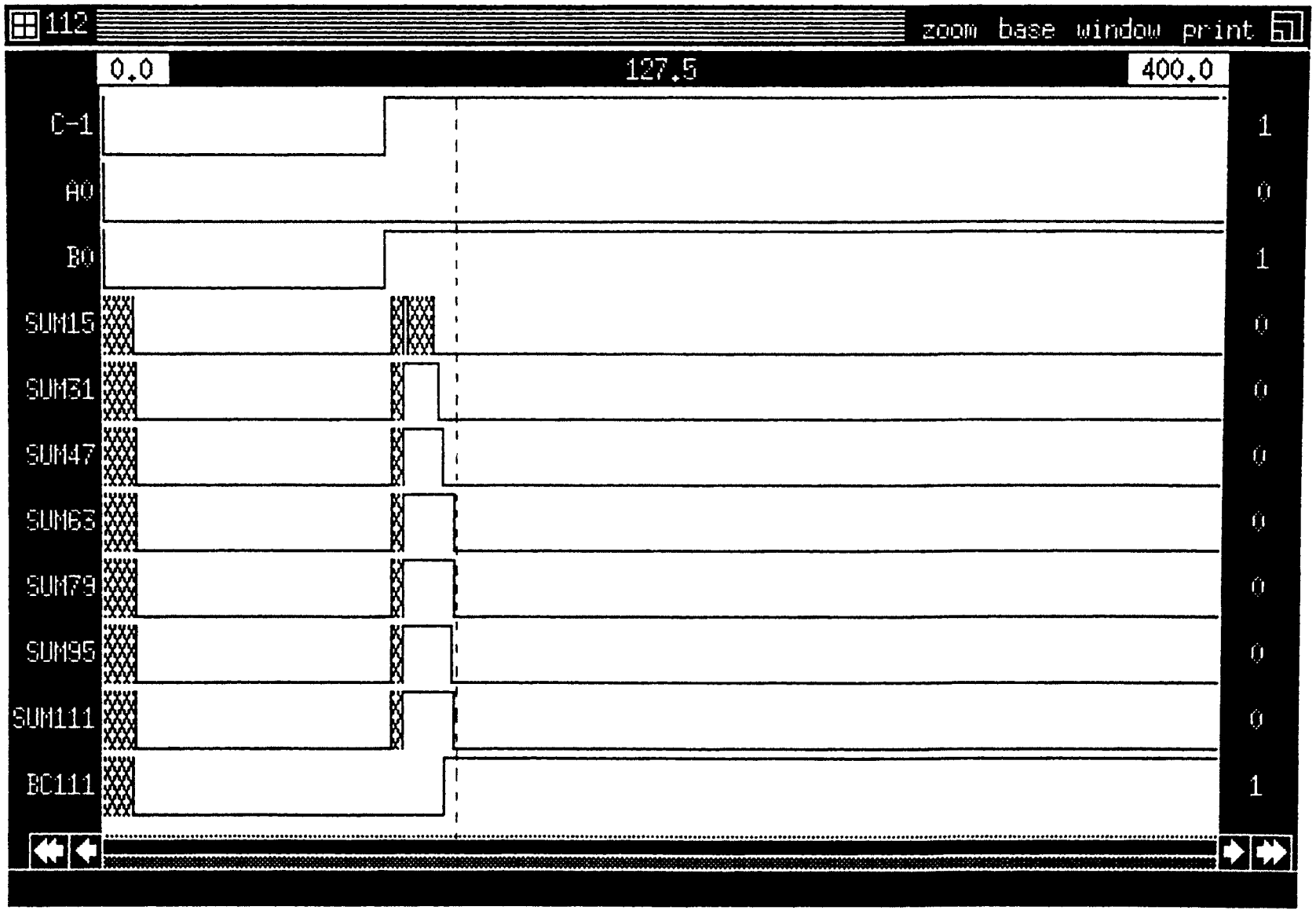


Fig.16 The simulation results of the 112 - b adder (c)

CHAPTER 5

PREPARATION FOR FABRICATION

There are 112 pairs of inputs and 112 outputs for our ***112 - b*** adder. In other words, the total signals for inputs and outputs are 336. The maximum pad number we can use to do fabrication is 64. So we need some buffers to reduce the numbers of pins or pads before the design circuit to be fabricated.

Fig. 17(a) shows the diagram of the I/O control circuit, where two pairs of inputs and two outputs share one pad. In the Fig. 17(a), circuit unit 3 uses D flip-flops[11]. This design uses only one clock phase and gated RS flip-flop; it is clock race immune. Circuit unit 2 represents adder bits and circuit unit 1 is a transmission gate with Rd_H and Rd_L as the control signals. Fig. 17(b) shows the waveforms of the clock control signals. There are 56 pins for input and output data. and 6 pins for I/O clock control signals. Plus Vdd and GND, the total number of pins is 64.

Fig. 18 shows the ***112 - b*** adder layout design with buffers on both sides. The area of the layout design is 2900x2800 μm^2.

Fig. 19 shows the layout design of whole circuit together with 64 pads. The total area of the layout together with pads is 6300x4500 μm^2.

In case some one want to fabricate this 112 - b transmission gate, we put all the detail arrangement of pads, inputs, outputs and clock signals on TABLE 5. It tells the pad number and the corresponding inputs and outputs bits.

In the table 5, the first group of rows represents the pad's number. One can find the number D1 ~D56 from the pads of the designed layout. The second group of rows labeled CL1 gives the all the corresponding input bits to which the input signals were sent from the corresponding pads during the clock circle 1. Based on the same principle, the third, the forth and the fifth row groups give the input bits to which the input signal were sent from the corresponding pads during the clock circle 2, 3 and 4. The last two row groups are the summation signals which were sent back to the corresponding pads through the clock signals RD-H and RD-L, respectively. Note that the portion of Table 5 on the following page should be on the right of the portion below.

Table 5 Arrangement of pads, inputs, outputs and clock signals

PAD	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28
CL1	A8	A10	A12	A14	A16	A18	A20	A22	A40	A42	A44	A46	A48	A50	A52	A54	A72	A74	A76	A78	A80	A82	A84	A86	A104	A106	A108	A110
CL2	B8	B10	B12	B14	B16	B18	B20	B22	B40	B42	B44	B46	B48	B50	B52	B54	B72	B74	B76	B78	B80	B82	B84	B86	B104	B106	B108	B110
CL3	A9	A11	A13	A15	A17	A19	A21	A23	A41	A43	A45	A47	A49	A51	A53	A55	A73	A75	A77	A79	A81	A83	A85	A87	A105	A107	A109	A111
CL4	B9	B11	B13	B15	B17	B19	B21	B23	B41	B43	B45	B47	B49	B51	B53	B55	B73	B75	B77	B79	B81	B83	B85	B87	B105	B107	B109	B111
RdH	S8	S10	S12	S14	S16	S18	S20	S22	S40	S42	S44	S46	S48	S50	S52	S54	S72	S74	S76	S78	S80	S82	S84	S86	S104	S106	S108	S110
RdL	S9	S11	S13	S15	S17	S19	S21	S23	S41	S43	S45	S47	S49	S51	S53	S55	S73	S75	S77	S79	S81	S83	S85	S87	S105	S107	S109	S111

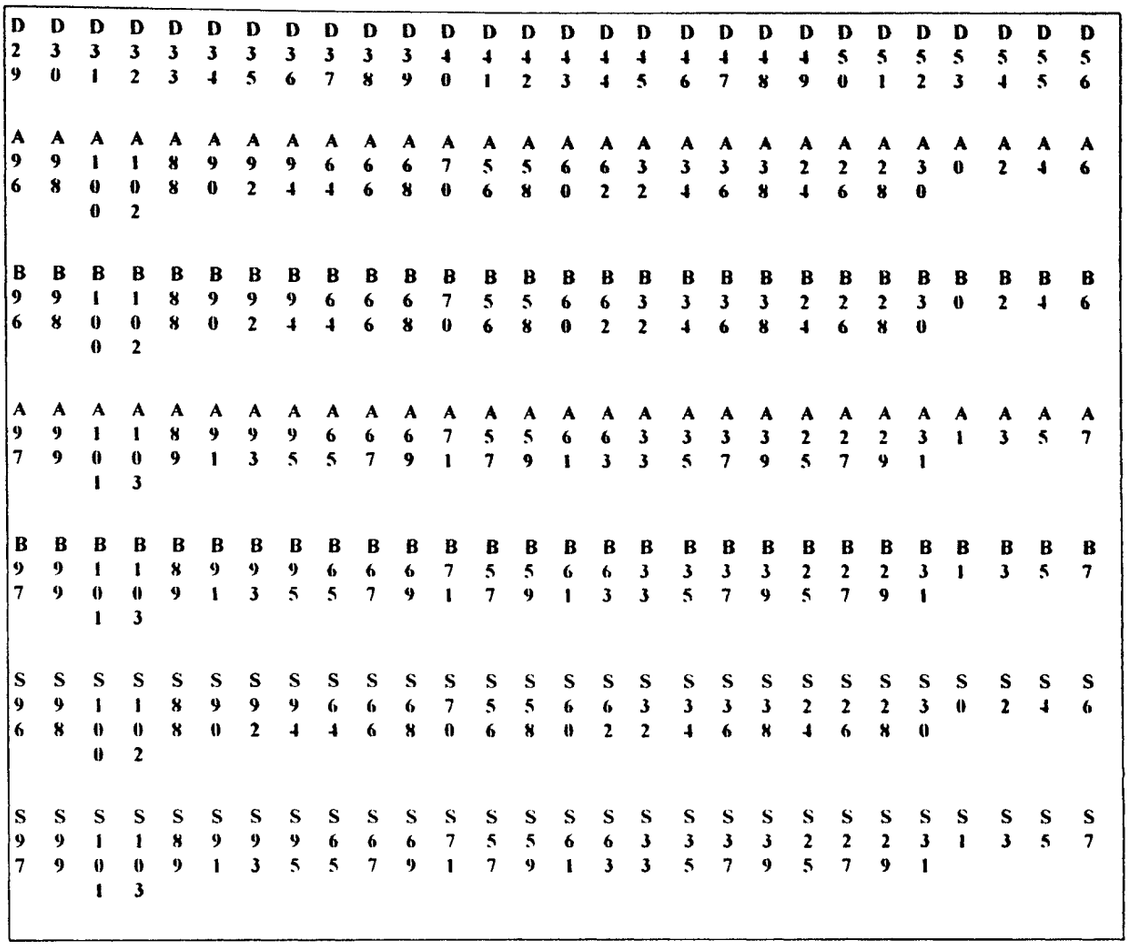


Fig. 20 shows the simulation results from the pads. By using the clock signals clock1, clock2, clock3 and clock4, the input data D25 were separated into two pairs of adder bit input signals: A104, B104 and A105, B105. The summation of two adder bit SUM104 and SUM105 were sent back to the pads D25 through the clock signals RD-H and RD-L.

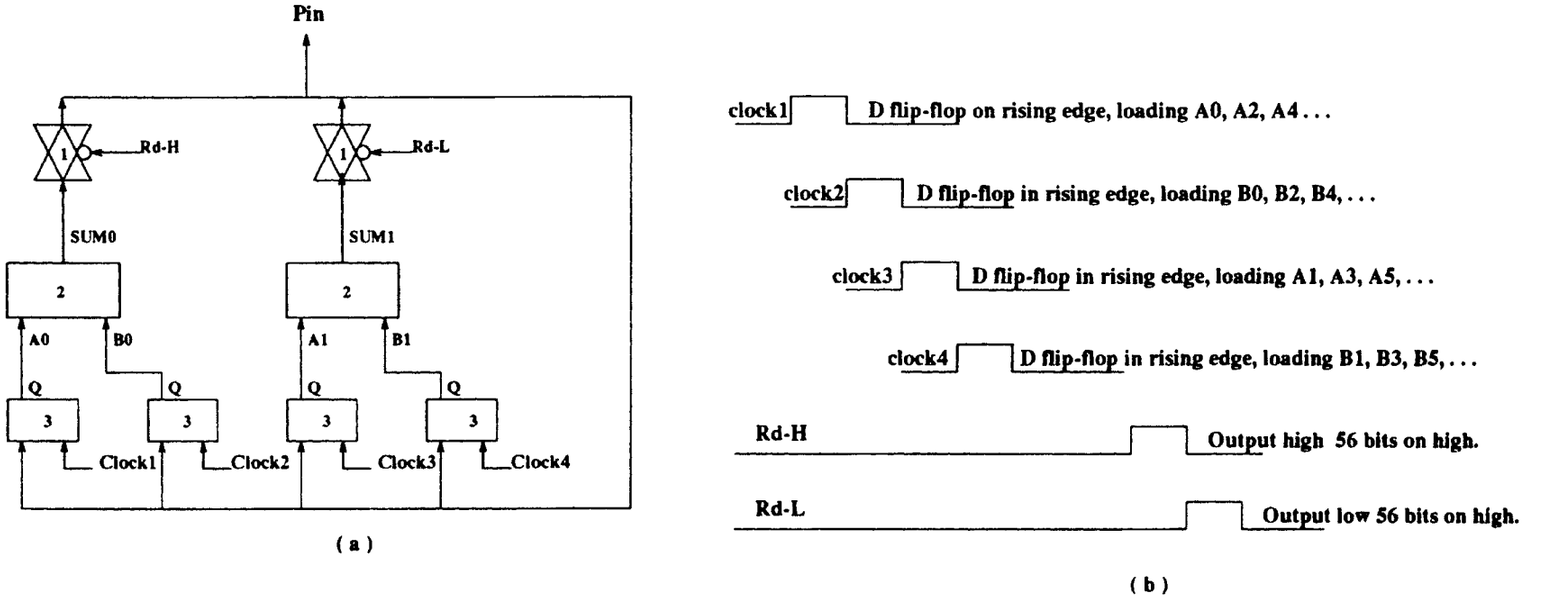


Fig. 17 Diagram of the I/O control cutcuit and waveforms

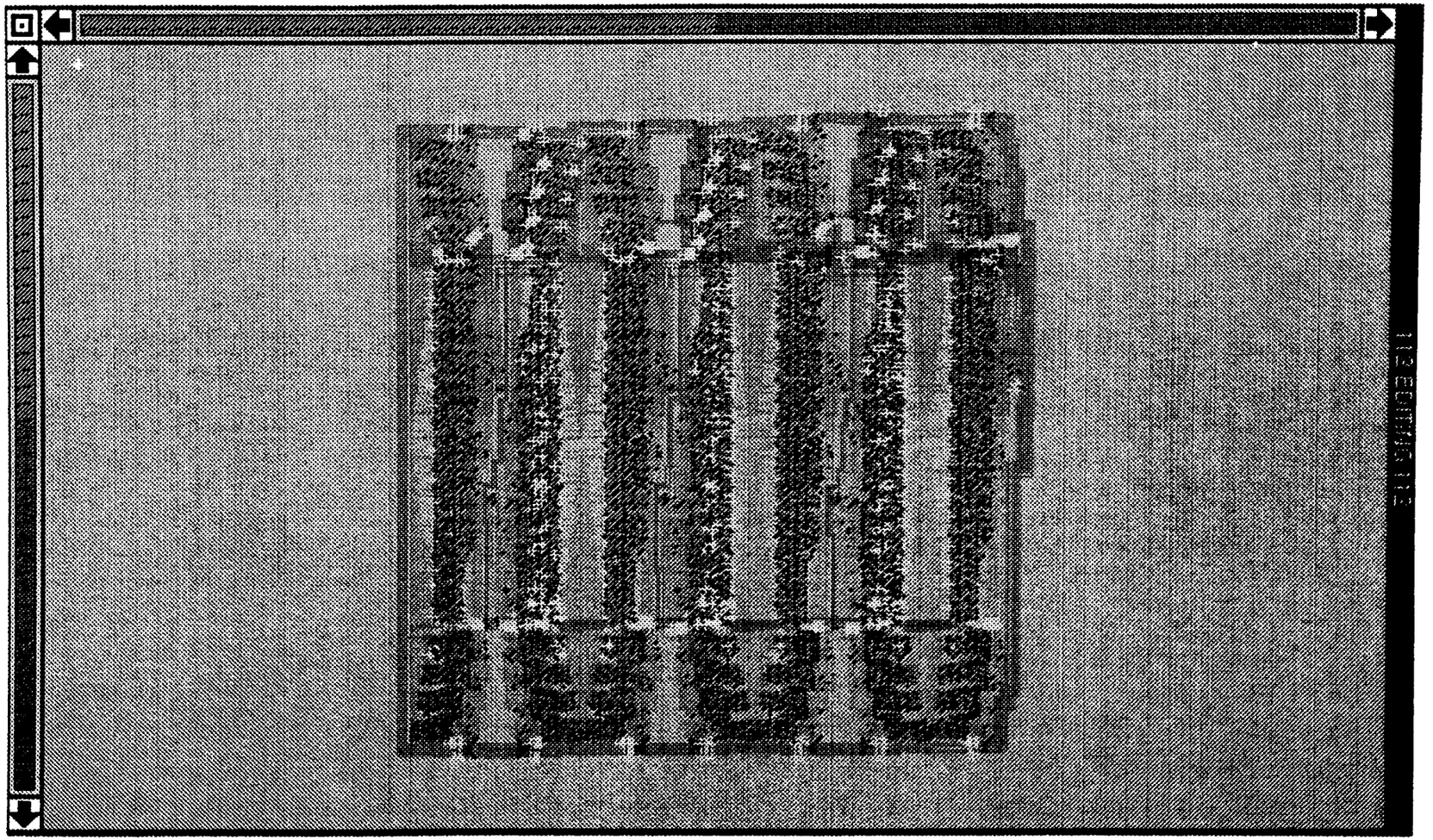


Fig. 18 The layout design of the 112 - b adder with buffers

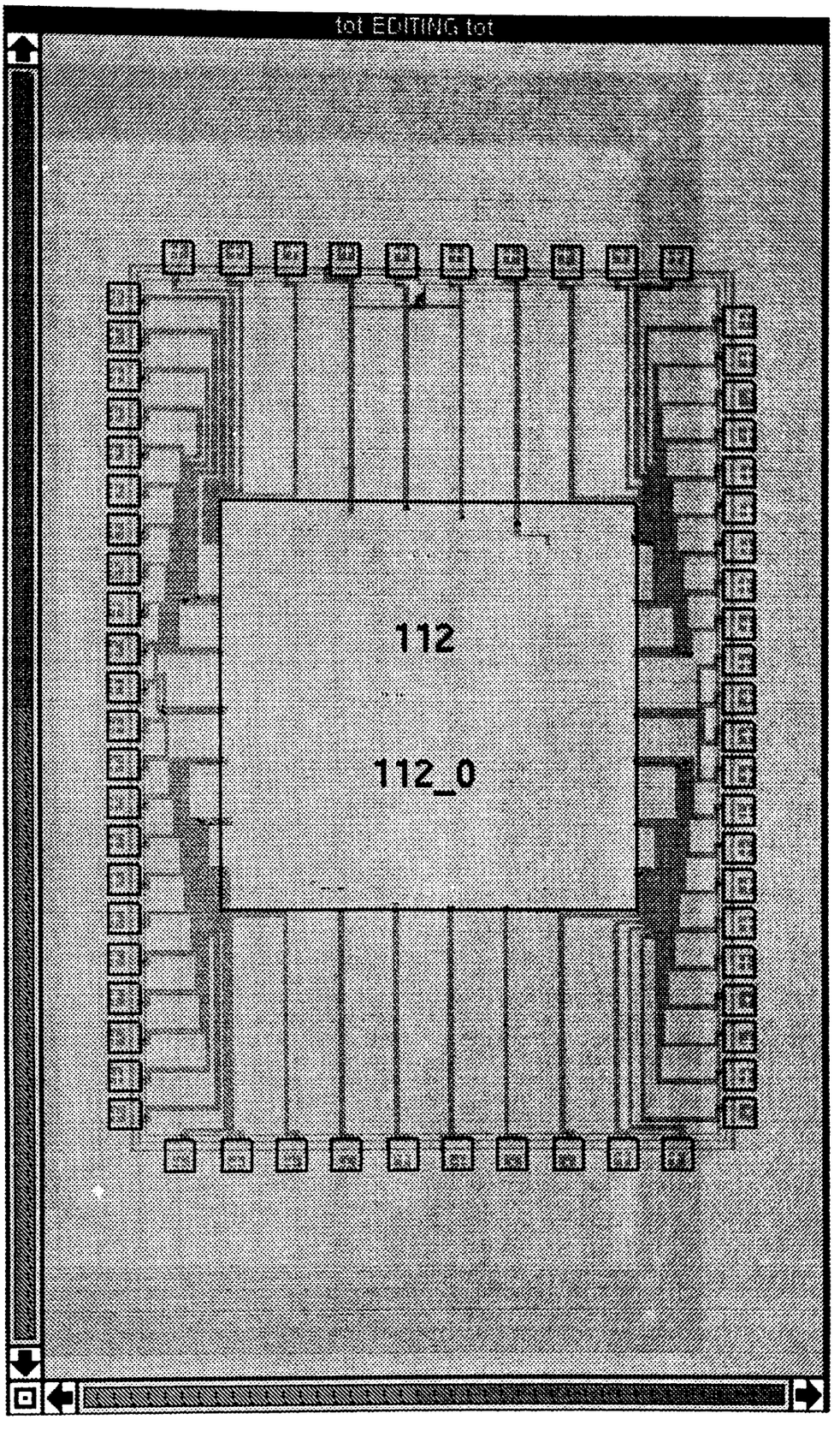


Fig.19 The layout design of the whole circuit

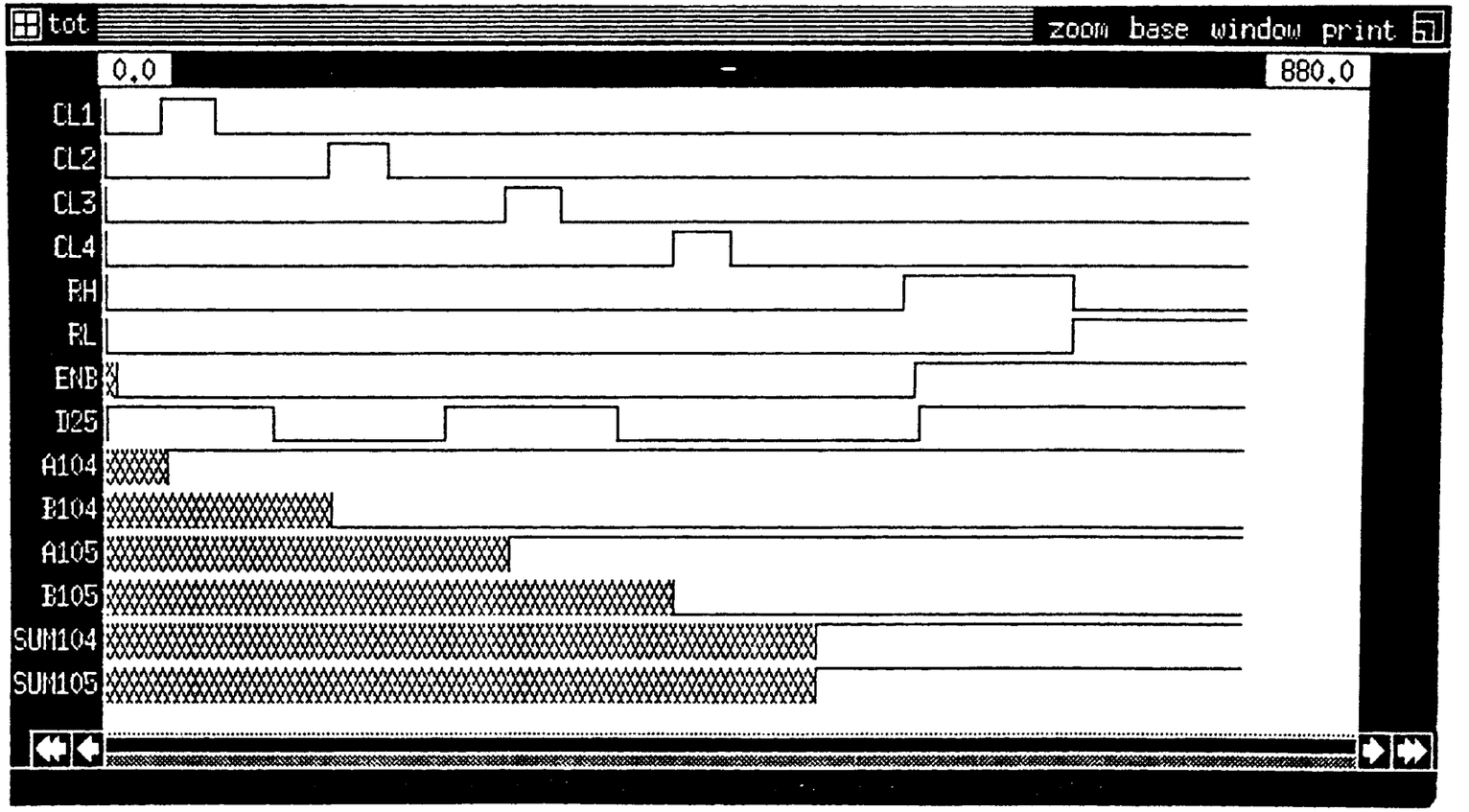


Fig. 20 The simulation results from the pads

CHAPTER 6

CONCLUSION AND COMPARISON

In this paper, we explained the basic circuit of the improved carry - skip adder, the summation of which is selected by logically multiplied block carry and block carry propagate signals based on the logical equation. We specified a general way how to build the conflict - free bypass signal. We designed the conflict - free bypass networks for the ***16 - b*** transmission gates adder and for the ***112 - b*** transmission gates adder based on the general law of the conflict - free bypass signal. The major effort of this work was the optimal layout design of ***112 - b*** adder by using the ***magic*** program, not only implemented the layout design for the adder but put the adder and 56 flip-flops together inside the 64 - pads package for preparation of the fabrication. Also, we used the ***Irsim*** program to simulate the final results.

In order to clarify the advantages of our adder, we compare it with a previously designed ***100 - b*** carry select adder[3], which uses multi - input NAND and NOR gates. This carry select adder is fabricated in a 1.0 μm, triple-metal, full-CMOS process. The delay for this adder is 10.7 ns and consists of 15,096 transistors. The other previously designed ***112 - b*** adder[2] uses the same bypass network as we use here. Though the addition time is 8.5 ns, they use the 0.8 μm full CMOS triple-metal process technology. We use 2 μm double-metal CMOS process to build ***112 - b*** transmission gate adder, the worst delay is 27.5 ns and about 11,000 transistors were used including the buffers. If we

If we re-estimated our addition time on the same process basis as the one used to fabricate their adder, our results are by no means poor. The most important point is that we clearly explained how the conflict-free bypass network can be fabricated via simulation showed the fabrication works.

REFERENCES

1. N. Weste and K. Eshraghian, *principles of CMOS VLSI Design.* Reading, MA: Addison-Wesley, 1985.

2. T. Sato, M. Sadate, T. Sukemura, and G, Goto, " An 8.5-ns 112-b transmission gate adder with a conflice-free bypass circuit." *IEEE Journal of solid-state Circuits,* vol. 27, no 4. April, 1992.

3. A. Guyot, B. Hocher, and J. M. Muller, " A way to build efficient carry - skip adders." *IEEE Trans. Comput.*, vol. C - 36. no. 10, pp. 1144-1151. Oct. 1987.

4. A. katsumo et al., " A 64-bit floating-point processing unit with a horizontal instruction code for parallel operations," *in proc. IEEE ICCD* 1990, pp. 347-350.

5. P. K. Chan et al., " Analysis and design of CMOS Manchester adders with variable carry - skip." *IEEE Trans., Comput.*, vol. 39, no. 8, pp. 983-992, Aug. 1990.

6. T. Kudo, T. Tokumaru, "Design of a 32 bit high-speed adder," in *IEICE 70th Anniv. Nat. Conv. Rec.*, no. 284, 1987, p. 2-88.

VITA

Xiaobo Liu

Candidate for the Degree of

Master of Science

Thesis: **EXPLANATION AND IMPLEMENTATION OF A 112 - b TRANSMISSION GATE ADDER**

Major Field: Electrical Engineering

Biographical:

Education: Received Bachelor of Science degree in physics from Nankai University, Tianjin, China in Oct. 1978; received Master of Science degree in Electrical Engineering from Beijing Polytechnic University, Beijing, China in Dec. 1985; completed the requirements for the Master of Science at Oklahoma State University in December 1994.

Experience: Engineer at Beijing Semiconductor Device Institute. Substantial Examiner at third Examination Dept., Chinese Patent Office for examine patent application in the semiconductor device and IC process.