LEADING BIT POSITION DETECTOR FOR NORMALIZATION OF FLOATING POINT NUMBERS - A 24-BIT PRIORITY ENCODER

T.T.B. Taiser[1] & A.L. Aita[2]
Universidade Federal de Santa Maria – UFSM
Centro de Tecnologia – CT
Departamento de Eletrônica e Computação – DELC
[1]E-mail: barros@mail.ufsm.br.br
[2]E-mail: aaita@inf.ufsm.br

Abstract - this paper presents the design of a 24-bit priority encoder working as a leading bit position detector, which is necessary in the normalization step in floating point addition operations. The 24-bit encoder is constructed with 8-bit cascadable encoders. This encoder has a 24-bit input and a 5-bit output code, which informs the number of left shifts necessary to normalize the result.

I. Introduction

The floating point addition is described usually as a 4-step procedure [HEN 96]: binary point alignment, addition, normalization and rounding. The normalization step is necessary because the addition/subtraction of two normalized number can result in a non-normalized one. In this case, this result should be normalized.

According to IEEE754/85 standard, the number is normalized if its significand has the following format:

$$1 \bullet xx \ldots x \quad (b_0 \bullet b_1 b_2 \ldots b_{23})$$

After a sum, if a carryout occurs, the result will have a different format, but normalization is easily performed by one right shift. The main problem occurs when two numbers are subtracted. In this case, cancellation is possible and the first significant one (the leading bit) can be anywhere along the significant. So, the number of left shifts is unknown a priori.

A priority encoder [WAK 00] is proposed as a solution, since the code it outputs, informs the position of the leading bit and thus the number of left shifts required to normalize the result.

Since the single precision floating point arithmetic deals with a 24-bit word, a 8-bit cascadable encoder was first designed, serving as a basic building block for the 24-bit leading bit detector.

II. 8-bit Priority position encoder

The 8-bit priority encoder is defined in the tab. I, and its representation is shown in fig. 1. It has 9 inputs, the 8-bit word and the enable-in signal Ein. It has 5 outputs, the 3-bit code Ci, the zero signal Z0 and the enable-out signal Eout. The 3-bit code Ci informs the number of left shifts necessary to normalize the number according to the position of the first significant bit. The enable signals are necessary to allow the cascading of encoders.

Table I
8-bit priority encoder truth table

Inputs									Outputs				
Ein	**b0**	**b1**	**b2**	**b3**	**b4**	**b5**	**b6**	**b7**	**C2**	**C1**	**C0**	**Z0**	**Eout**
1	1	x	x	x	x	x	x	x	0	0	0	0	0
1	0	1	x	x	x	x	x	x	0	0	1	0	0
1	0	0	1	x	x	x	x	x	0	1	0	0	0
1	0	0	0	1	x	x	x	x	0	1	1	0	0
1	0	0	0	0	1	x	x	x	1	0	0	0	0
1	0	0	0	0	0	1	x	x	1	0	1	0	0
1	0	0	0	0	0	0	1	x	1	1	0	0	0
1	0	0	0	0	0	0	0	1	1	1	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	1
0	x	x	x	x	x	x	x	x	0	0	0	0	0

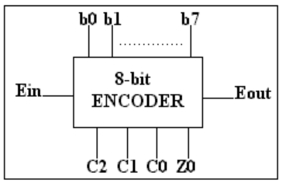


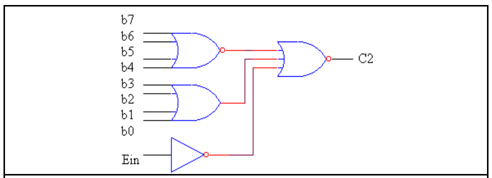
Figure. 1 – 8-bit priority encoder

The Boolean equations of the encoder, obtained after some algebraic manipulation, are shown in the Table. II.

Table II
Encoder outputs

C2 = ((b0 + b1 + b2 + b3) + (b4 + b5 + b6 +b7)’ + Ein’)’
C1 = (b0 + b1 + (b2 + b3 + (b4’b5’b6) + (b4’b5’b7))’ + Ein’)’
C0 = (b0 + (b1 + (b2’b3) + (b2’b4’b5) + (b2’b4’b6’b7))’ + Ein’)’
Z0 = ((b0 + b1 +b2)’ (b3 + b4 + b5)’ (b6’b7’Ein))
Eout = Ein Z0

The logical and electrical designs of the 8-bit encoder are shown in fig. 2 and fig. 3, respectively.



(a)

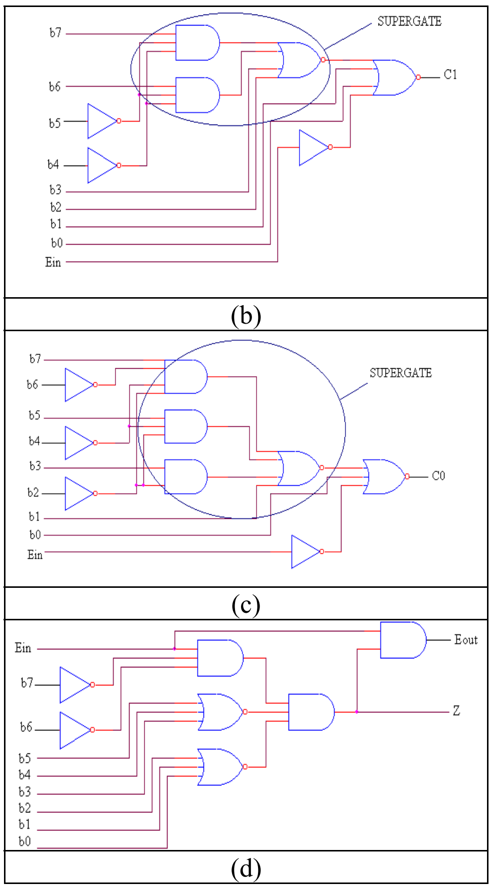
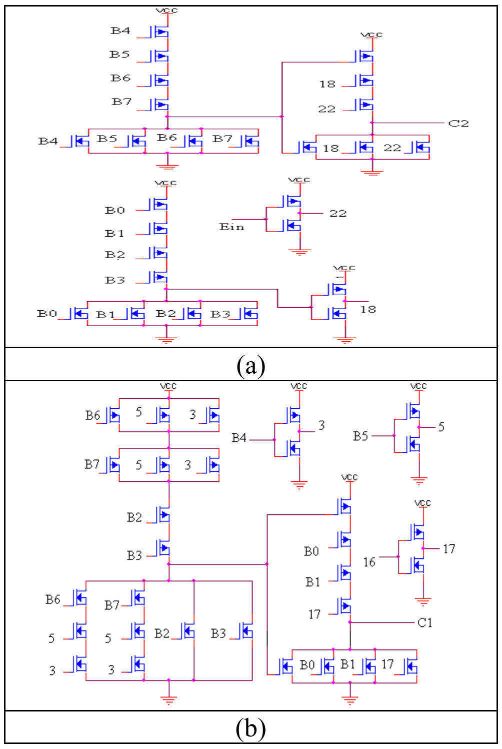


Figure. 2 – 8-bit priority encoder logical schematic, showing (a) C2, (b) C1, (c) C0 and (d) Eout and Z outputs



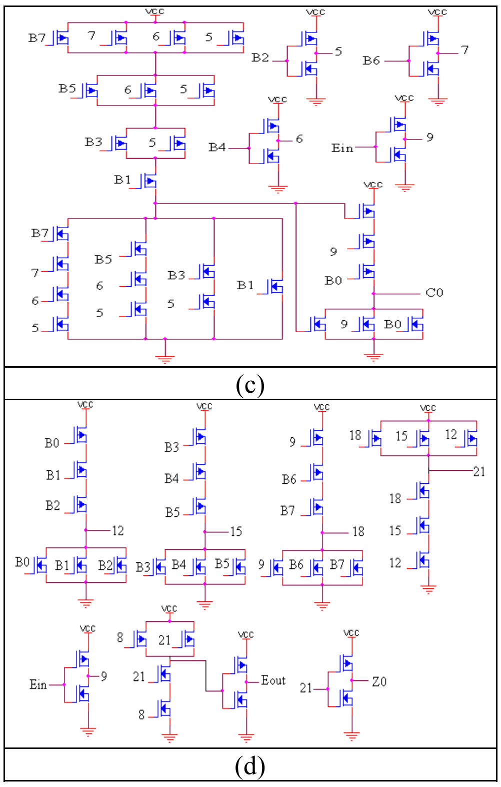


Figure. 3 – 8-bit priority encoder electrical schematic, showing (a) C2, (b) C1, (c) C0 and (d) Eout and Z outputs

III. 24-bit Priority encoder - The leading bit position detector

The 24-bit encoder was implemented cascading three 8-bit encoder just presented. The 8-bit encoder outputs are used as input to the last stage of the 24-bit encoder, that generates a 5-bit coded number, which informs a programmable shifter how many left shifts should be done in the normalization step, fig. 4 shows the block diagram of this encoder while Table. III shows the final leading bit detector truth table.

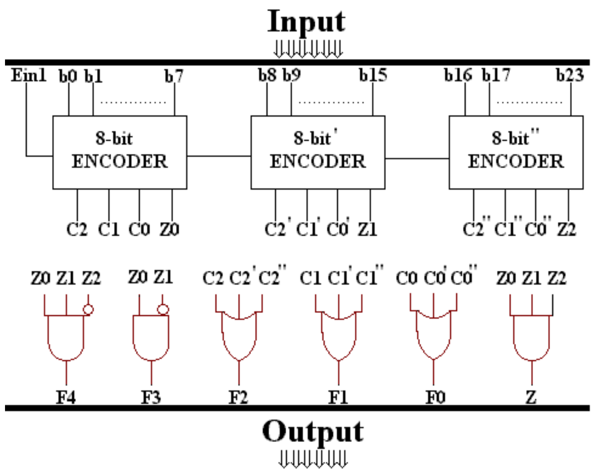


Figure. 4 – 24-bit priority encoder block diagram

Table III
24-bit priority encoder truth table

Inputs			Outputs					
Z0	Z1	Z2	F4	F3	F2	F1	F0	Z
0	x	x	0	0	C2	C1	C0	0
1	0	x	0	1	C2’	C1’	C0’	0
1	1	0	1	0	C2’’	C1’’	C0’’	0
1	1	1	0	0	0	0	0	1

IV. Simulation Results

Logical and electrical simulations were performed. The logical simulation was performed within Xilinx Foundation Environment to verify the logical functioning of the circuit, fig. 5 shows the logical simulation of the final circuit.

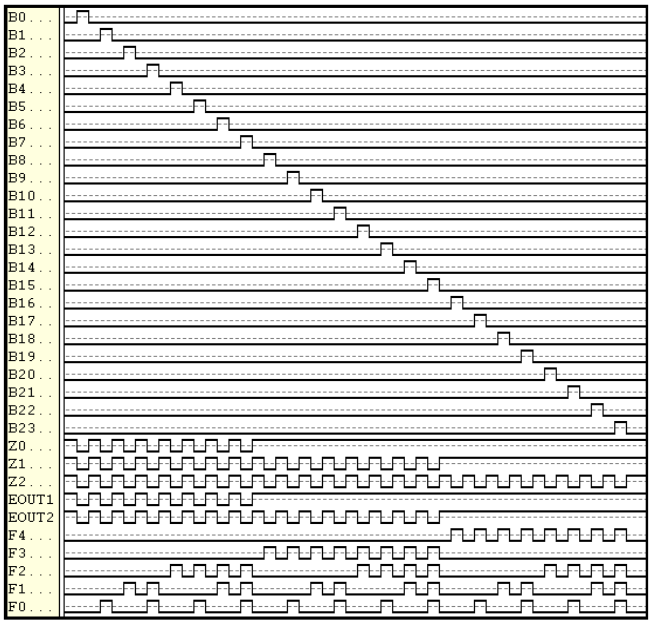
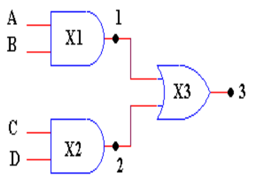


Figure. 5 – 24-bit encoder: leading bit detector logical simulation

V. Layout

The Tropic3 [MOR 00] tool was used to generate the 24-bit encoder layout. The circuit description at gate level is the input of the tool. The output is the circuit layout (.cif file), which can be visualized with the layout editor Edllex [MOR 00]. An example of the Tropic3 input description is shown in Table. IV and the layout generated for the 24-bit encoder is shown in fig. 6.

Table IV
(a) Logic circuit, (b) Tropic3 description

	.include librairie X1 A B 1 Vcc and2 X2 C D 2 Vcc and2 X3 1 2 3 Vcc or2 * interface A in Nt * interface B in Nt * interface C in Nt * interface D in Nt * interface 1 out S * interface 2 out S * interface 3 out S
(a)	(b)

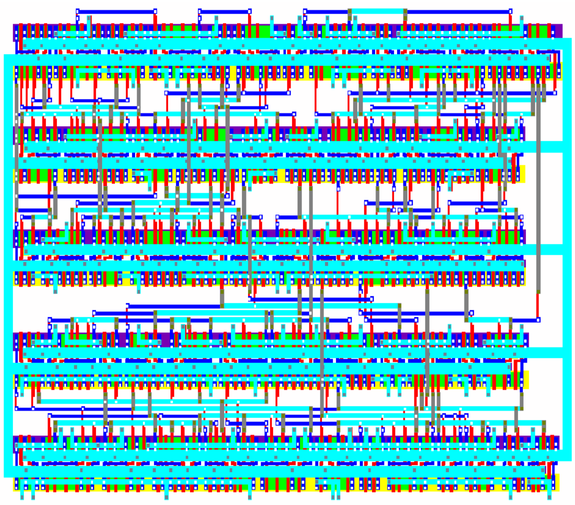


Figure. 6 – 24-bit priority encoder layout

VI. Conclusions

The normalization step in the floating-point subtraction depends on the position of the result-leading bit. To determine its position, a 24-bit priority encoder was proposed. Initially, an 8-bit cascadable encoder was designed. The Boolean equations and logical circuits were determined from its truth table. The electrical circuits were also designed. Then, three 8-bit encoders were cascaded, and the full encoder constructed. Logical and electrical simulation were performed to verify the functioning of the final circuit and to identify worst delays. After simulations, an automatic layout generator tool, Tropic3, generated the leading bit layout.

VII. References

[HEN 96] David A. Patterson e John L. Hennessy. **Computer Architecture: A Quantitative Approach**. 2º. ed. Academic Press. 1996

[IEE 85] **IEEE Standard for Binary Floating-Point Arithmetic**. ANSI/IEEE754-1985, New York, 1985.

[MOR 00] MORAES, Fernando Gehm. **Anatomia de uma Ferramenta de Síntese Automática de Layout**. Technical Report Series, Agosto de 2000.

[WAK 00] Wakerly, John F., **Combinational Logic Design Practices**, Editora Prentice – Hall International Editions, 2000, p. 271-277