Dual Half/Full, Carry-Increment, and Conditional Sum Adders

James E. Stine
Electrical and Computer Engineering Department
Oklahoma State University
Stillwater, OK 74078, USA

1. Dual Half Adders

   (a) A Dual Half Adder (DHA) generates the sum and carry bits for each position.

   (b) The logic equations for a DHA are

$$\begin{aligned} s_k^0 &= a_k \oplus b_k \\ s_k^1 &= \overline{a_k \oplus b_k} = \overline{s_k^0} \\ c_{k+1}^0 &= a_k \cdot b_k \\ c_{k+1}^1 &= a_k + b_k \end{aligned}$$

   (c) The delays for a DHA are

$$\begin{aligned} a_k, b_k \rightarrow s_k^0 &= 3\,\triangle \\ a_k, b_k \rightarrow s_k^1 &= 4\,\triangle \\ a_k, b_k \rightarrow c_{k+1}^0, c_{k+1}^1 &= 1\triangle \end{aligned}$$

2. Dual Ripple Carry Adders (DRCA)

   (a) A DRCA computes one set of sum and carry bits with a carry in of zero, and a second set with a carry in of one.

   (b) A single r-bit DRCA can replace two r-bit RCAs, when constructing a Carry Select Adder

   (c) The first bit of the DRCA uses a Dual Half Adder (DHA).

   (d) A DHA uses 5 gates and has the following delays:

$$\begin{aligned} a_k, b_k \rightarrow s_k^0 &= 3\,\triangle \\ a_k, b_k \rightarrow s_k^1 &= 4\,\triangle \\ a_k, b_k \rightarrow c_{k+1}^0, c_{k+1}^1 &= 1\triangle \end{aligned}$$

   (e) All of the other bits use dual full adders (DFA).

   (f) A DFA uses $2 \cdot 5 + 4 = 14$ gates and has the following gate delays:

$$\begin{aligned} a_k, b_k \rightarrow s_k^0 &= 7\,\triangle \\ a_k, b_k \rightarrow s_k^1 &= 6\,\triangle \\ a_k, b_k \rightarrow c_{k+1}^0 &= 5\,\triangle \\ a_k, b_k \rightarrow c_{k+1}^1 &= 6\,\triangle \\ c_k^0 \rightarrow s_k^0 &= 3\,\triangle \\ c_k^1 \rightarrow s_k^1 &= 3\,\triangle \\ c_k^0 \rightarrow c_{k+1}^0 &= 2\,\triangle \\ c_k^1 \rightarrow c_{k+1}^1 &= 2\triangle \end{aligned}$$

   (g) An n-bit DRCA requires 1 DHA and $(n-1)$ DFAs. That is, it requires $5 + (n-1) \cdot 14$ or $14 \cdot n - 9$ gates and has the following delays.

$$\begin{aligned} a_k, b_k \rightarrow s_{n-1}^0, s_{n-1}^1 &= (2n+3)\,\triangle \\ a_k, b_k \rightarrow c_n^0, c_n^1 &= (2n+2)\triangle \end{aligned}$$

[h!]

3. Carry Select Adders with DRCA

   (a) The DRCA can obviously be integrated with the Carry Select Adder to reduce the overall area content.

4. Carry Increment Adder (CINA)

   (a) The carry increment adder was refined by R. Zimmermann and basically formulated by A. Takagi titled, "A reduced-area scheme for carry-select adders" in 1993.

   (b) The idea is an algorithmic enhancement of carry-select adders using the ideas from the Weinberger-Smith Carry-Lookahead Concept.

   (c) The basic proof of enhanced equation requires adding a redundant term to the equation we formulated with the DHA. That is, it first requires using the following equation, which we will attempt to prove:

$$c_{k+1}^1 = c_{k+1}^0 + c_{k+1}^1$$

To prove this equation, we break part the equations from the DHA definitions.

$$\begin{aligned}
c_{k+1}^1 &= (g_k + p_k \cdot c_k^1) + (g_k + p_k \cdot c_k^0) \\
c_{k+1}^1 &= g_k + g_k + p_k \cdot {c_k}^1 + p_k \cdot c_k^0 \\
c_{k+1}^1 &= g_k + p_k \cdot c_k^1 + p_k \cdot c_k^0 \\
c_{k+1}^1 &= g_k + p_k \cdot (a_{k-1} + b_{k-1} + a_{k-1} \cdot b_{k-1}) \\
c_{k+1}^1 &= g_k + p_k \cdot (a_{k-1} \cdot b_{k-1}) = c_{k+1}^1
\end{aligned}$$

(d) This proof basically uses two elements of DHA, such that:

$$\begin{aligned}
c_{k+1}^0 &= a_k \cdot b_k \\
c_{k+1}^1 &= a_k + b_k
\end{aligned}$$

(e) So, the CINA uses the above formulation in the idea that carry-out or c_{k+1} can be selected between two values inside a multiplexor or mux. That is, the following relationship holds:

$$c_{k+1} = \overline{c_{in}} \cdot c_{k+1}^0 + c_{in} \cdot c_{k+1}^1$$

(f) Using this carry-out equation and the new redundant-enhanced equation forms the following relationship:

$$\begin{aligned}
c_{k+1} &= \overline{c_{in}} \cdot c_{k+1}^0 + c_{in} \cdot (c_{k+1}^0 + c_{k+1}^1) \\
c_{k+1} &= c_{k+1}^0 + c_{in} \cdot c_{k+1}^0 \\
c_{k+1} &= (g_k + p_k \cdot {c_k}^0) + c_{in} \cdot (g_k + p_k \cdot c_k^1) \\
c_{k+1} &= c_{k+1}^0 + c_{in} \cdot p_k \cdot c_k^1 \\
c_{k+1} &= c_{k+1}^0 + c_{in} \cdot p_{k:k-r}
\end{aligned}$$

(g) The previous simplification is achieved due to the absorption theorem in Boolean logic, such that: $g_k + c_{in} \cdot g_k = g_k$.

(h) The last item assumes the basic ideas of the carry-select adder in that a carry-in of 1 (i.e., c_k^1) produces the group-propagate signal. Therefore, the RCA that produces the correct sum is incremented based on the value of c_{k+1}.

(i) Generalized CINA Gate Counts

   i. The first block, similar the carry-select adder, is nothing more than a carry-propagate ader.

   ii. Each subsequent r block uses a RCA to produce the correct sum as well as the bitwise propagate signals from a 9 gate FA.

   iii. It uses $\lceil n/r \rceil - 1$ sets of carry logic blocks, each of which carries 2 gates.

   iv. The second RCA is a RCA that is composed of HA blocks adding s_k^0 and c_{k+1} from the previous block. Since the RCA just needs a c_{in}, the number of gates is $4 \cdot (n - r)$.

   v. Thus, the total number of gates used by a n-bit CINA with r-bit blocks is:

$$\begin{aligned}
&= 9 \cdot n + 2 \cdot \left(\lceil \frac{n}{r} \rceil - 1 \right) + 4 \cdot (n - r) \\
&= 9 \cdot n + 2 \cdot \lceil \frac{n}{r} \rceil - 2 + 4 \cdot n - 4 \cdot r \\
&= 13 \cdot n + 2 \cdot \lceil \frac{n}{r} \rceil - 2 - 4 \cdot r
\end{aligned}$$

(j) Generalized CINA Delay

   i. The first block is a RCA and has a delay of $2 \cdot r + 3$

   ii. The next $(\lceil n/r \rceil - 1)$ blocks have a delay of $2\triangle$ for the carry to skip.

   iii. The last block must go through the RCA incrementor, which is composed of HAs aligned like a RCA. The delay of this structure is

$$(r - 1) \cdot 1 + 3 = (r + 2)\ \triangle$$

   iv. Thus, the total delay for s_{n-1} is:

$$\begin{aligned}
&= (2 \cdot r + 3) + 2 \cdot (\lceil \frac{n}{r} \rceil - 2) + (r + 2) \\
&= 3 \cdot r + 2 \cdot \lceil \frac{n}{r} \rceil + 1
\end{aligned}$$

5. Conditional Sum Concept.

   (a) The conditional sum adder (CSUA) generates a pair of sum and carry bits is generated at each bit position. One pair assumes carry in of one and the other assumes a carry in of zero.

   (b) The correct sums and carries are then selected using a tree of multiplexors.

6. Example of Conditional Sum Addition

7. An 8-bit Conditional Sum Adder

   (a) An 8-bit CSUA requires one row of DHAs and three levels of multiplexors.

   (b) The first row has a delay of $5\triangle$ to produce the carry out of the full adder, and the next three rows each have a delay of $4\triangle$ for the muxes. Thus, the total delay for an 8-bit CSUA is $5 + 3 \cdot 4 = 17\triangle$

stage	k	7	6	5	4	3	2	1	0
0	a_k	1	0	1	1	0	1	1	0
	b_k	0	0	1	0	1	1	0	1
1	s_k^0	1	0	0	1	1	0	1	1
	c_{k+1}^0	0	0	1	0	0	1	0	0
	s_k^1	0	1	1	0	0	1	0	
	c_{k+1}^1	1	0	1	1	1	1	1	
2	s_k^0	1	0	0	1	0	0	1	1
	c_{k+1}^0	0		1		1		0	
	s_k^1	1	1	1	0	0	1		
	c_{k+1}^1	0		1		1			
3	s_k^0	1	1	0	1	0	0	1	1
	c_{k+1}^0	0				1			
	s_k^1	1	1	1	0				
	c_{k+1}^1	0							
4	s_k	1	1	1	0	0	0	1	1
	c_{k+1}	0							

(c) The first row requires 7 DHAs (5 gates each) and 1 FA (9 gates). The second row requires 7 mux21x2 (8 gates each). The third row requires 3 mux21x3 (12 gates). The fourth row requires 1 mux21x5 (20 gates). Thus, the total number of gates is

$$7 \cdot 5 + 9 + 7 \cdot 8 + 3 \cdot 12 + 20 = 156$$

8. Generalized CSUA Gate Count (assume n is a power of 2)

   (a) An n-bit CSUA uses $n-1$ DHAs and 1 FA in the first level.

   (b) After this, there are $\lceil \log_2(n) \rceil$ rows of muxes. If the mux rows are labeled with indices $i = 0, 1, \ldots, \lceil log_2(n) \rceil - 1$, then in row i number of muxes is $(\frac{n}{2^i} - 1)$, the number of bits per mux is $2^i + 1$, and the number of gates per mux is $4(2^i + 1) = 4 \cdot 2^i + 4$.

   (c) Thus the toal number of gates for an n-bit CSUA is

$$\begin{aligned} & 5 \cdot (n-1) + 9 + \\ & \sum_{i=0}^{\lceil log_2(n) \rceil - 1} (\frac{n}{2^i} - 1)(4 \cdot 2^i + 4) \\ \approx \; & 4 \cdot n \cdot \log_2(n) + 9 \cdot n - 4 \cdot log_2(n) \end{aligned}$$

9. Generalized CSUA Delay

   (a) The first row has a delay of 5 $\triangle$ to produce the carry out of the full adder.

   (b) Each mux row has a delay of 3 $\triangle$ and there are $\lceil \log_2(n) \rceil$ rows of muxes.

   (c) Thus, the total delay is $(5 + 3 \cdot \lceil \log_2(n) \rceil)\ \triangle$.

10. Characteristics of CSUAs

   (a) CSUAs require the largest number of gates. The have $O(n \cdot log_2(n))$ area, whereas, the other adders (beside the CLA) had $O(n)$ area.

   (b) Like the CLA, CSUAs have logarithmic delay. However, there delay is always base two, whereas, the delay for the CLA is base r, where r is the maximum number of inputs.

   (c) For $r = 2$ CSUAs become faster than CLAs, as n gets large.

   (d) CSUAs are highly irregular and difficult to layout. Consequently, they are seldom implemented in practice.

11. Hybrid Carry Select Adders (HCSEA)

   (a) With a HCSEA, the first adder is a carry lookahead adder, and a carry lookahead generator is used to produce the carries.

   (b) 16-bit HCSEA with 4 bit blocks has a worst case delay of 16$\triangle$ and uses 310 gates.

   (c) If DRCAs are used, the worst case delay is reduced to 15$\triangle$ and only 254 gates are required.

12. Power Dissipation in Adders

   (a) The main source of power dissipation in well designed CMOS circuits is due to low-to-high logic transitions in digital circuits.

   (b) This power dissipation, sometimes called dynamic power dissipation, can be expressed as

$$P = V_{DD}^2 \cdot f_{clk} \cdot C_{eff} \cdot p_t$$

   where V_{DD} is the source voltage, f_{clk} is the clock frequency, C_{eff} is the effective capacitance and p_t is the probability of a low-to-high logic transition.

   (c) Dynamic power dissipation can be reduced by reducing any of the above 4 factors more or less.

   (d) If all other factors are constant, then the probability of low-to-high logic transitions is a good measure of the total power.

   (e) The actual power consumption ranks and the output transitions ranks are fairly close. The main exceptions to this is the Carry

Lookahead Adder and the Majerski Ripple Adder. Both of these adders have gates with higher fan-in, which increases the capacitance.

(f) There is still some significant research that needs to be done in this area.

References

[1] T. K. Callaway and E. E. Swartzlander, “Estimating the power consumption of cmos adders,” in *Proceedings of IEEE 11th Symposium on Computer Arithmetic*, pp. 210–216, 1993.

Table 1: 16-Bit Adder Area [1].

Adder Type	Area (mm^2)	rank	Gate Count	rank
Ripple Carry	0.26	2	144	2
Majerski Ripple Carry	0.23	1	122	1
Constant Carry Skip	0.33	3	156	3
Variable Carry Skip	0.49	4	170	4
Carry Lookahead	0.53	5	200	5
Brent and Kung	0.53	6	203	6
Hybrid Carry Select	0.88	7	284	7
Conditional Sum	1.14	8	368	8

Table 2: 16-Bit Adder Delay [1].

Adder Type	Delay (nsec)	rank	Gate Delay	rank
Ripple Carry	51.4	8	36	8
Majerski Ripple Carry	34.3	7	19	6
Constant Carry Skip	28.6	6	23	7
Variable Carry Skip	22.8	5	17	4
Carry Lookahead	22.5	4	10	1
Brent and Kung	22.1	3	18	5
Hybrid Carry Select	18.6	1	14	3
Conditional Sum	21.2	2	12	2

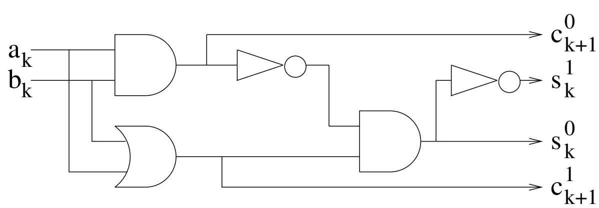

Figure 1: Dual Half Adder.

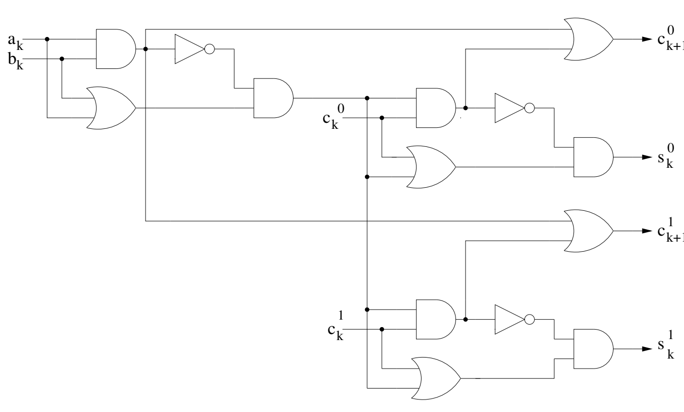

Figure 2: Dual Full Adder.

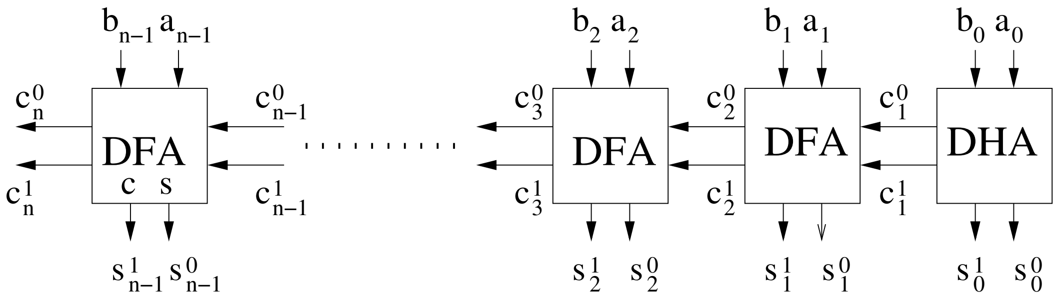

Figure 3: Dual Ripple Carry Adder.

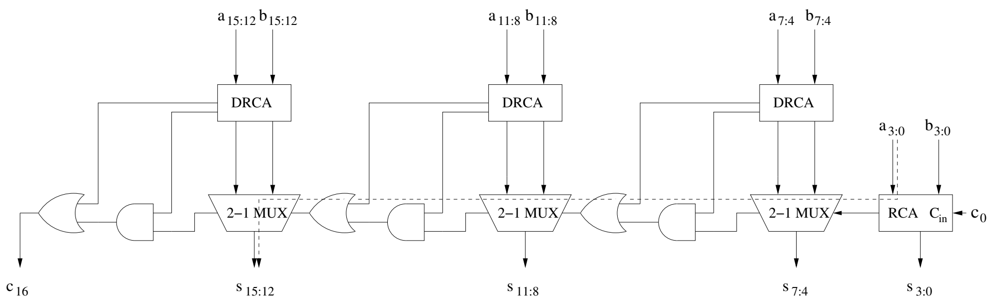

Figure 4: 16-bit Carry Select Adder using DRCA.

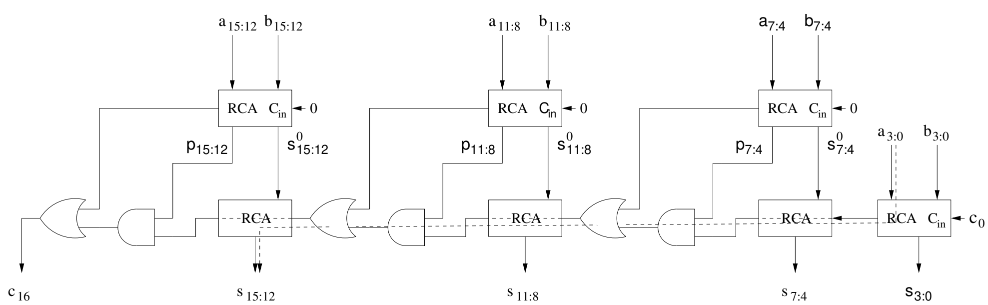

Figure 5: 16-bit Carry Increment Adder with $r = 4$.

Table 3: 16-Bit Adder Power [1].

Adder Type	Power (mWatt)	rank	Gate Output Transitions	rank
Ripple Carry	1.7	1	90	1
Majerski Ripple Carry	2.7	4	91	2
Constant Carry Skip	1.8	2	99	3
Variable Carry Skip	2.2	3	108	5
Carry Lookahead	2.7	5	100	4
Brent and Kung	3.1	6	112	6
Hybrid Carry Select	3.8	7	150	7
Conditional Sum	5.4	8	231	8

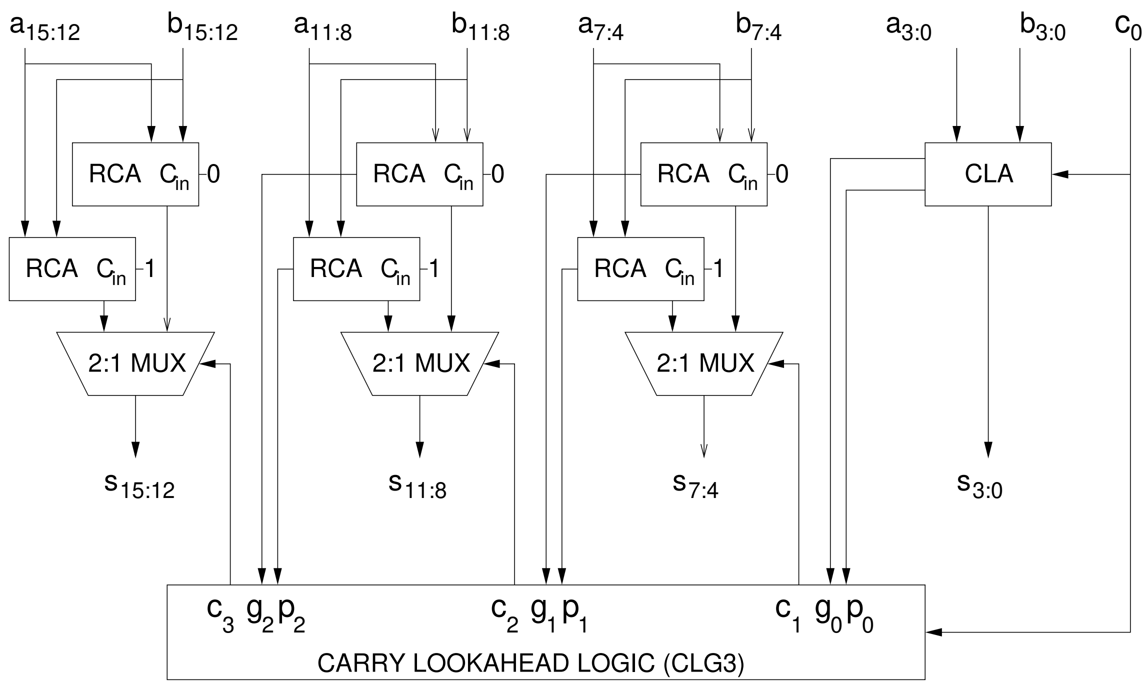

Figure 6: 16-bit Hybrid Carry Select Adder.

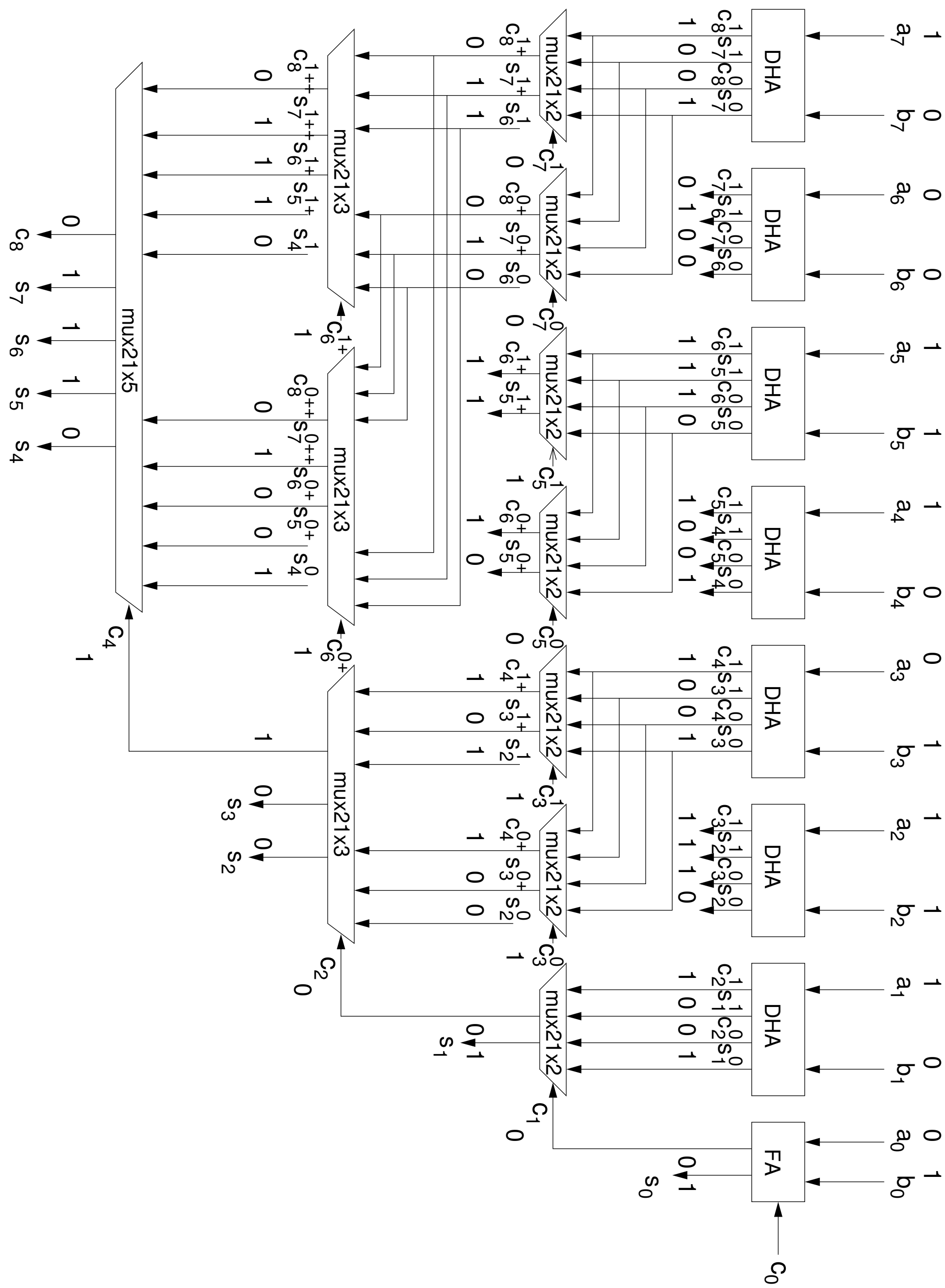

Figure 7: 8-bit Conditional Sum Adder.