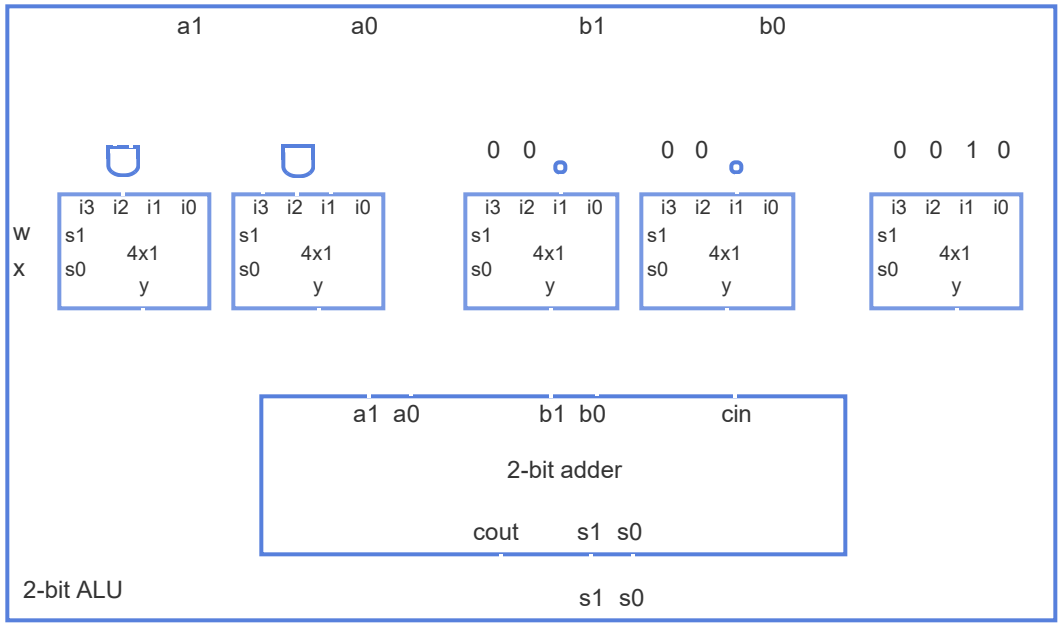
7.16 ALUs

An ***ALU*** is a datapath component capable of performing various arithmetic and logic functions, like add, subtract, and AND. ***arithmetic-logic unit***. Using an ALU rather than multiple components trades off reduced size for increased delay.

An ALU can be designed using muxes in front of an adder's A, B, and cin inputs to pass certain values to the adder. For exar be computed by passing A, B', and 1, respectively.

PARTICIPATION ACTIVITY 7.16.1: 2-bit ALU.

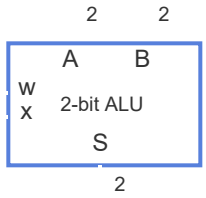
Start 2x speed



Function table

w	x	ALU operataior
0	0	S = A + B
0	1	S = A - B
1	0	S = A AND B
1	1	S = A

Block symbol

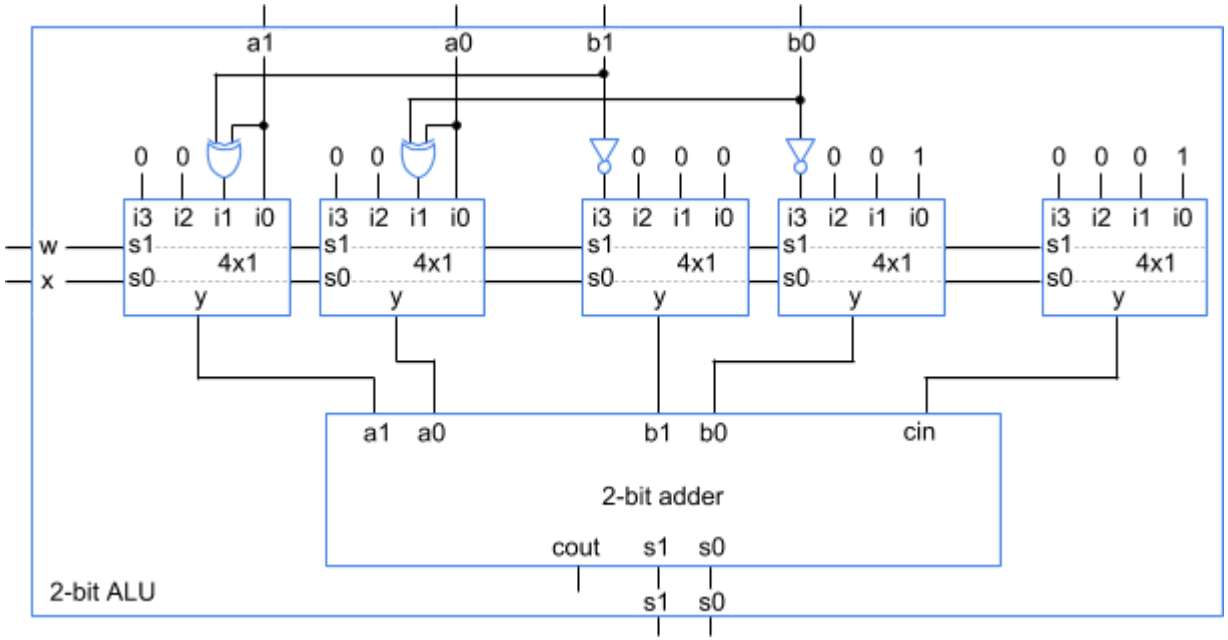


A more-capable ALU is described below. For each desired ALU operation, the items to be passed through the muxes to the cin inputs are also listed.

Table 7.16.1: A more-capable ALU's operation table with the corresponding mux configurations.

Control inputs			ALU operation	Mux configuration		
v	w	x		A	B	cin
0	0	0	S = A + B	A	B	0
0	0	1	S = A - B	A	B'	1
0	1	0	S = A + 1	A	0	1
0	1	1	S = 0	0	0	0
1	0	0	S = A AND B	AB	0	0
1	0	1	S = A OR B	A OR B	0	0
1	1	0	S = A XOR B	A XOR B	0	0
1	1	1	S = NOT A	A'	0	0

PARTICIPATION ACTIVITY 7.16.2: ALU operations.



Refer to the ALU implementation above.
For each of the wx values provided, determine the corresponding ALU operation.

wx = 11 **wx = 10** **wx = 00** **wx = 01**

S = 0

S = B'

S = A + 2

S = A XOR B

Reset

PARTICIPATION ACTIVITY 7.16.3: ALU mux connections.

Assume an 8-bit ALU. Determine the mux configuration needed to implement the given ALU operation.

1) S = E + F
   - ○ A = E, B = F, cin = 0
   - ○ A = E, B = F, cin = 1

2) S = E - F
   - ○ A = E, B = F, cin = 1
   - ○ A = E', B = F, cin = 1
   - ○ A = E, B = F', cin = 1

3) S = E AND F
   - ○ A = E AND F, B = 1, cin = 1
   - ○ A = E AND F, B = 0, cin = 1
   - ○ A = E AND F, B = 0, cin = 0

4) S = 1
   - ○ A = 1, B = 0, cin = 0
   - ○ A = 1, B = 1, cin = 1

5) S = 2E (multiply 2 · E)
   - ○ A = E, B = E, cin = 0
   - ○ A = 2, B = E, cin = 0

Provide feedback on this section