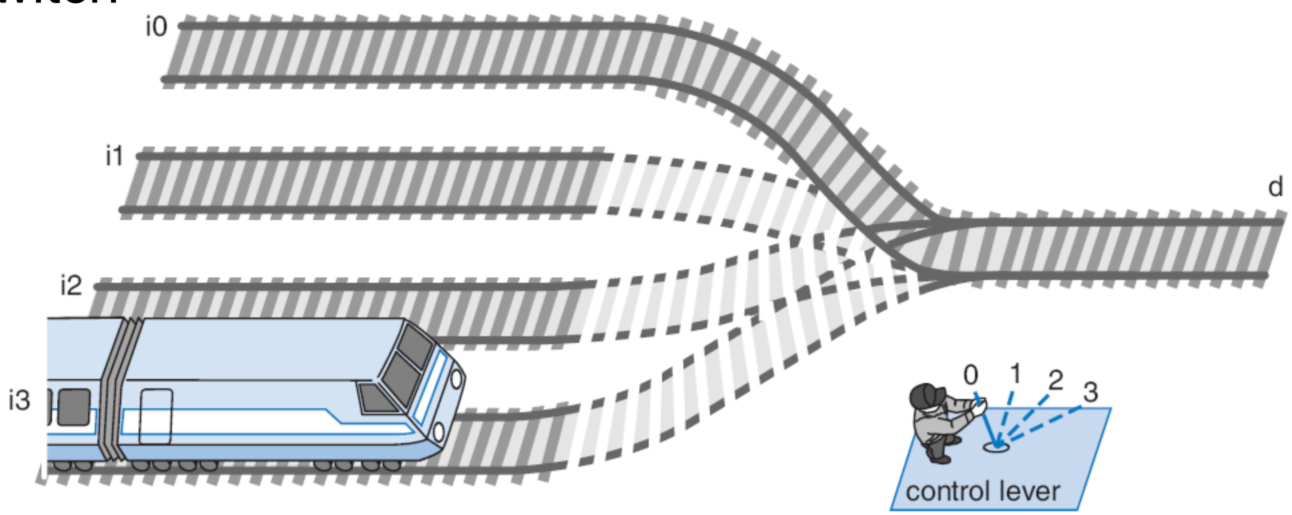
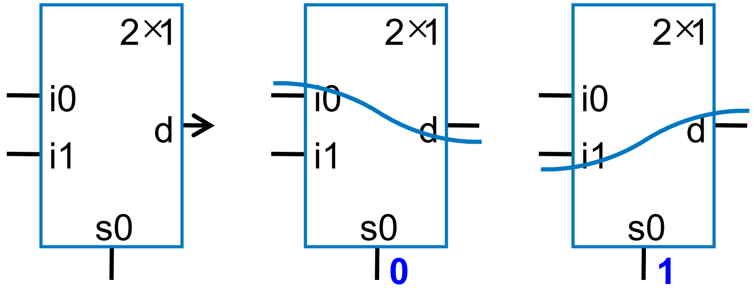
Multiplexor (Mux)

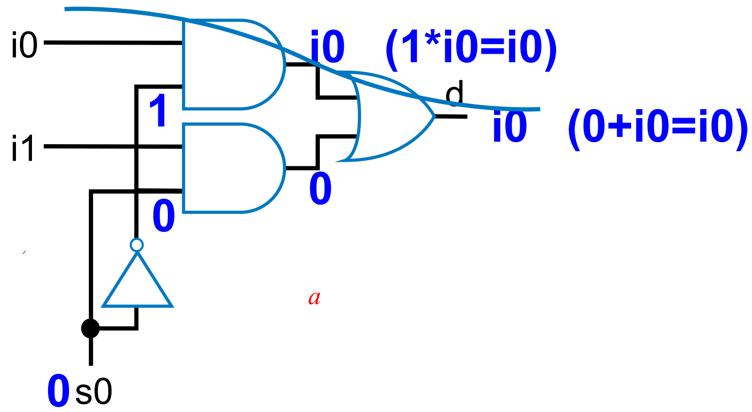
- Mux: Another popular combinational building block
  - Routes one of its N data inputs to its one output, based on binary value of select inputs
    - 4 input mux → needs 2 select inputs to indicate which input to route through
    - 8 input mux → 3 select inputs
    - N inputs → $\log_2(N)$ selects
  - Like a rail yard switch

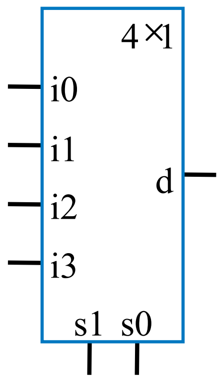


Mux Internal Design

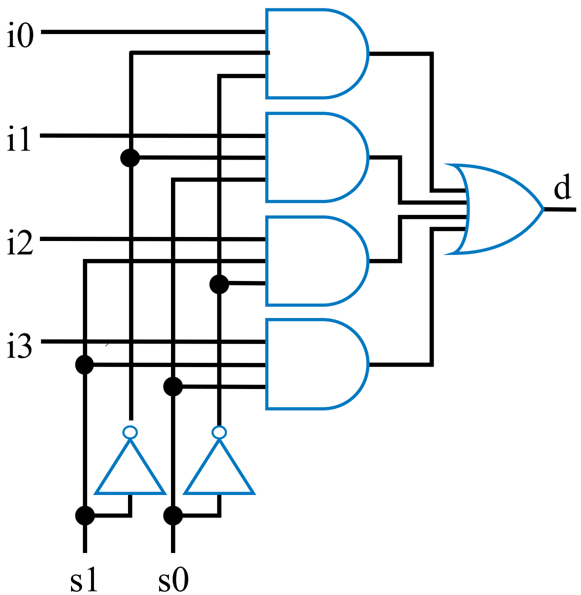


2x1 mux



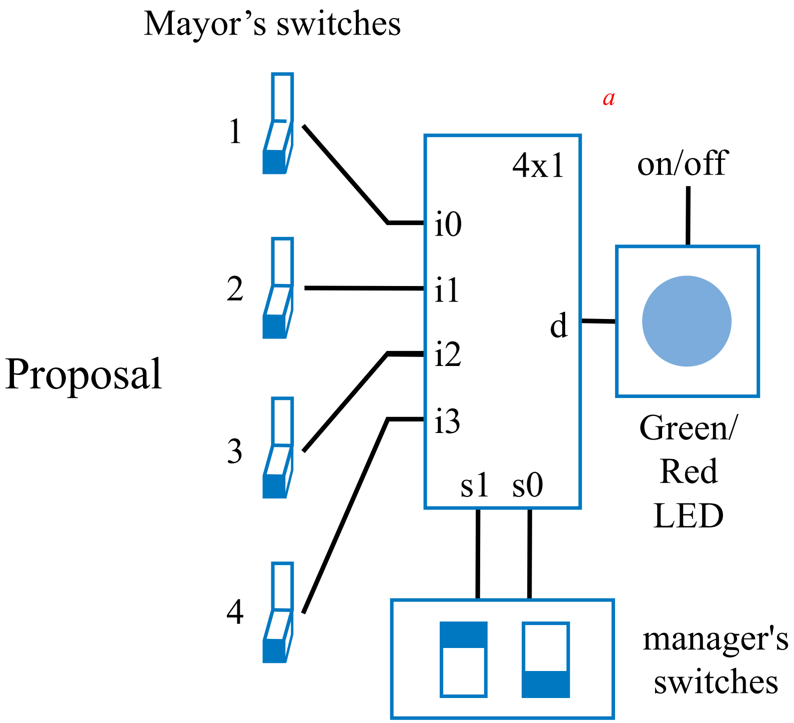


4x1 mux

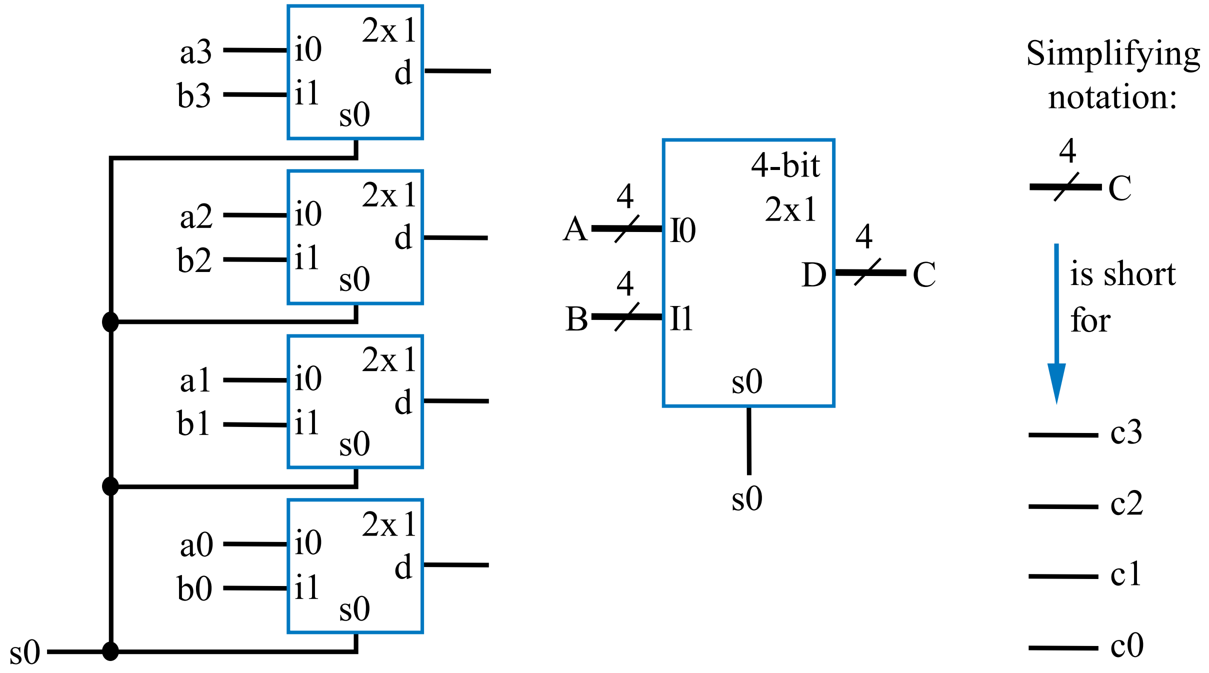


Mux Example

- City mayor can set four switches up or down, representing his/her vote on each of four proposals, numbered 0, 1, 2, 3
- City manager can display any such vote on large green/red LED (light) by setting two switches to represent binary 0, 1, 2, or 3
- Use 4x1 mux

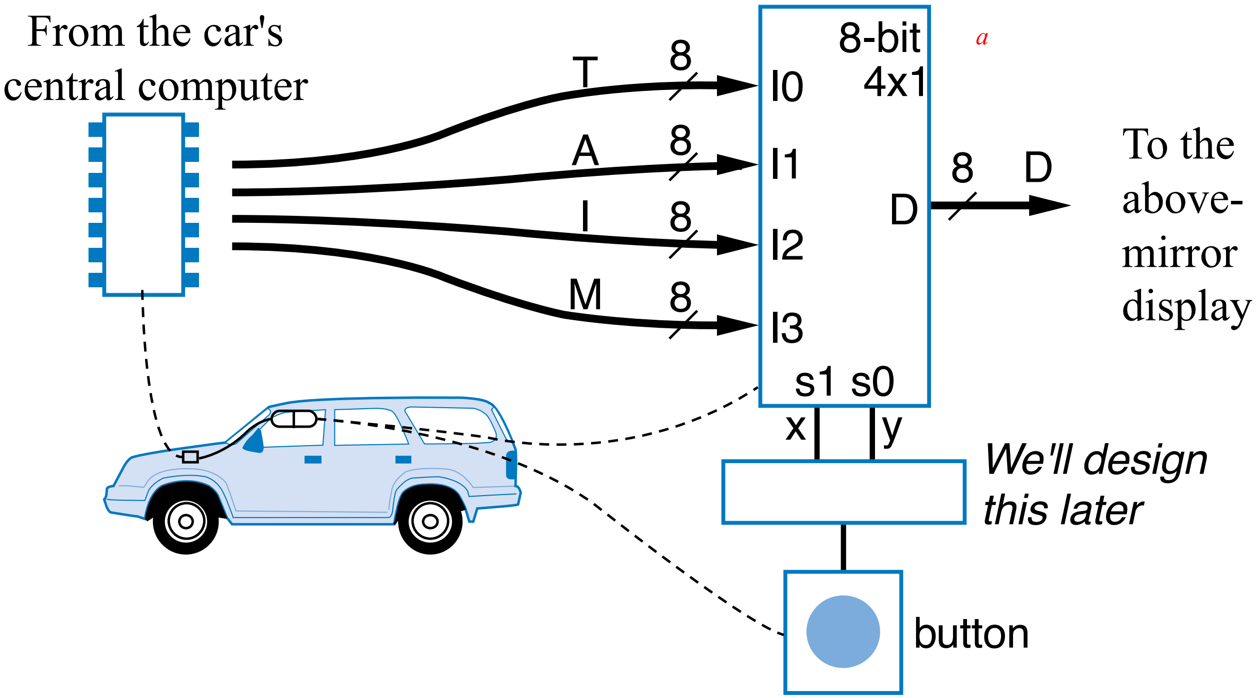


Muxes Commonly Together – N-bit Mux



- Ex: Two 4-bit inputs, A (a3 a2 a1 a0), and B (b3 b2 b1 b0)
  - 4-bit 2x1 mux (just four 2x1 muxes sharing a select line) can select between A or B

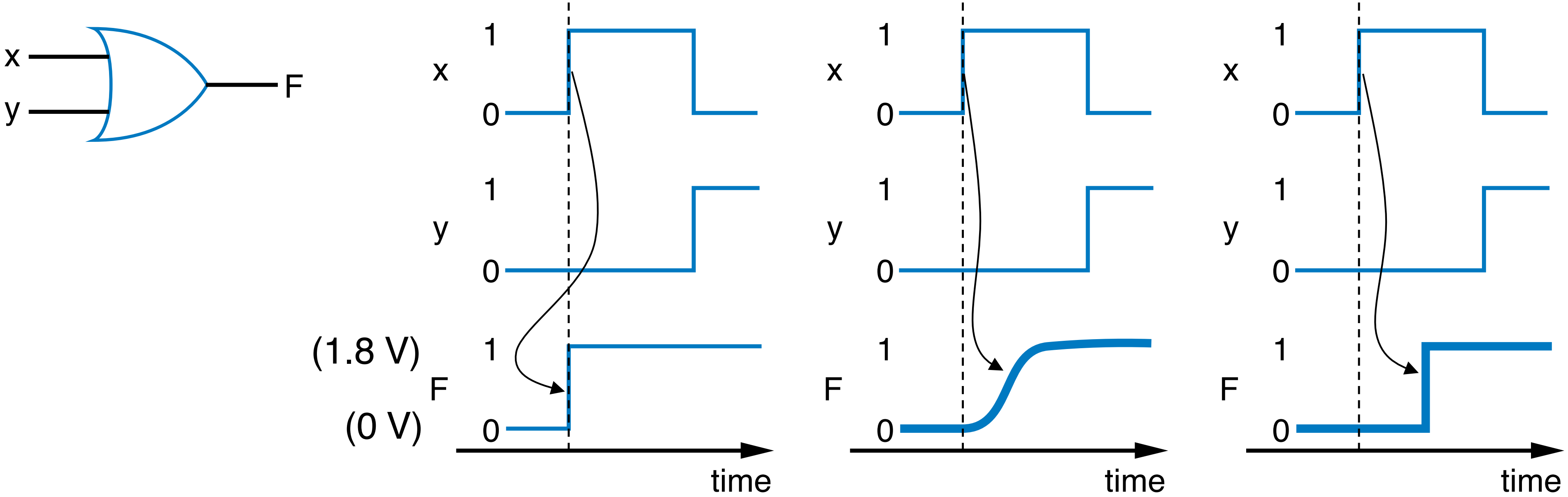
N-bit Mux Example



- Four possible display items
  - Temperature (T), Average miles-per-gallon (A), Instantaneous mpg (I), and Miles remaining (M) – each is 8-bits wide
  - Choose which to display on D using two inputs x and y
    - Pushing button sequences to the next item
  - Use 8-bit 4x1 mux

Additional Considerations

Non-Ideal Gate Behavior -- Delay



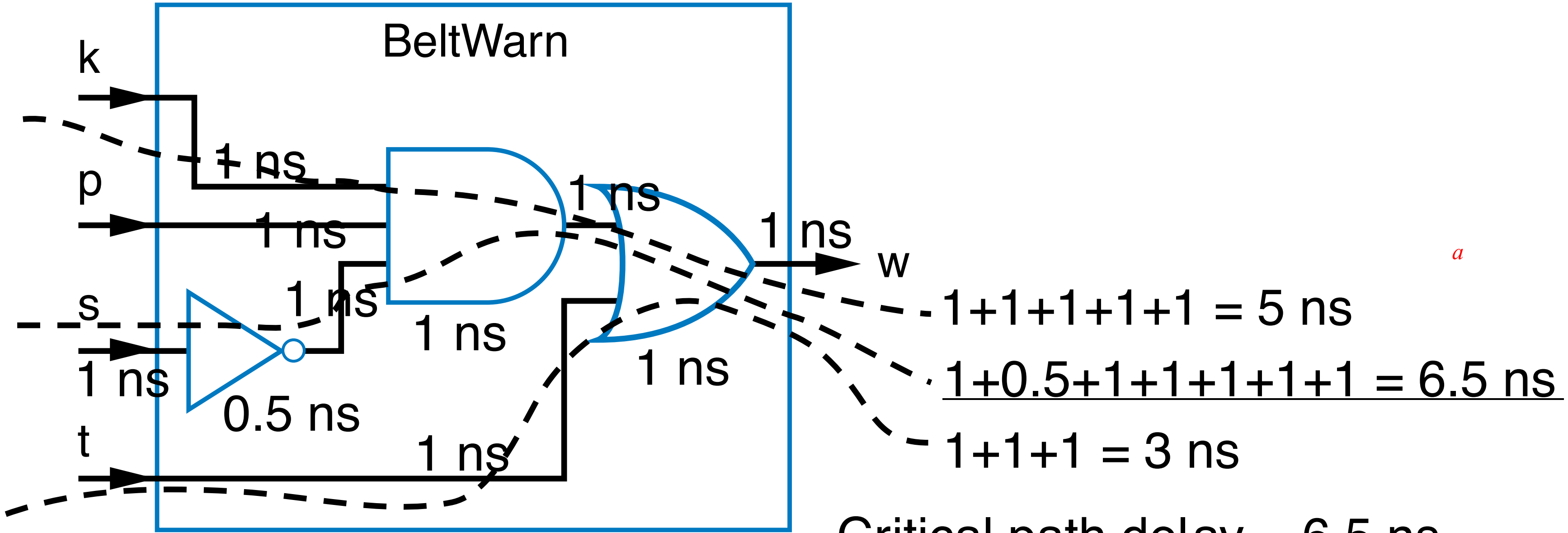
(**a**)
ideal [a]

(**b**)
more realistic [a]

(**c**)
with delay but otherwise ideal [a]

- Real gates have some delay
  - Outputs don't change immediately after inputs change

Circuit Delay and Critical Path

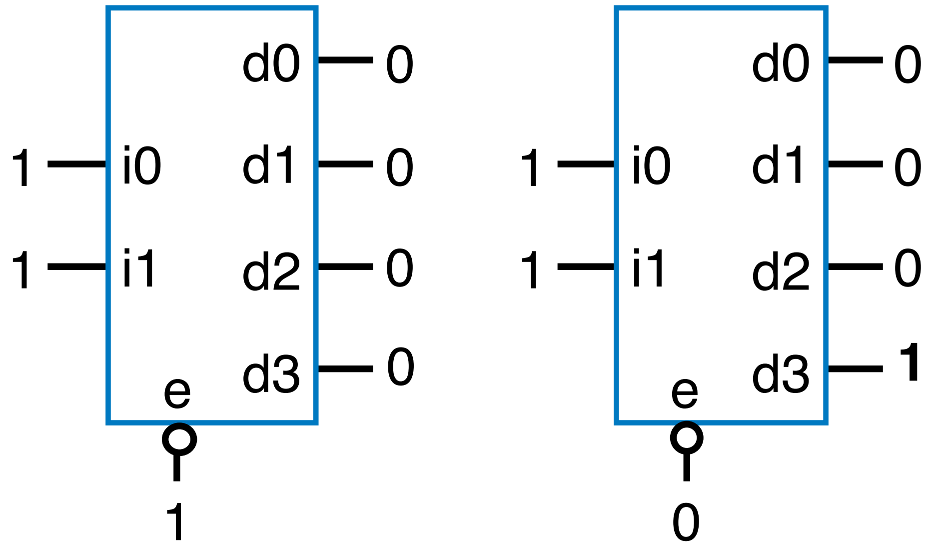


Critical path delay = 6.5 ns
Hence, circuit’s delay is 6.5 ns

- Wires also have delay
- Assume gates and wires have delays as shown
- Path delay – time for input to affect output
- Critical path – path with longest path delay
- Circuit delay – delay of critical path

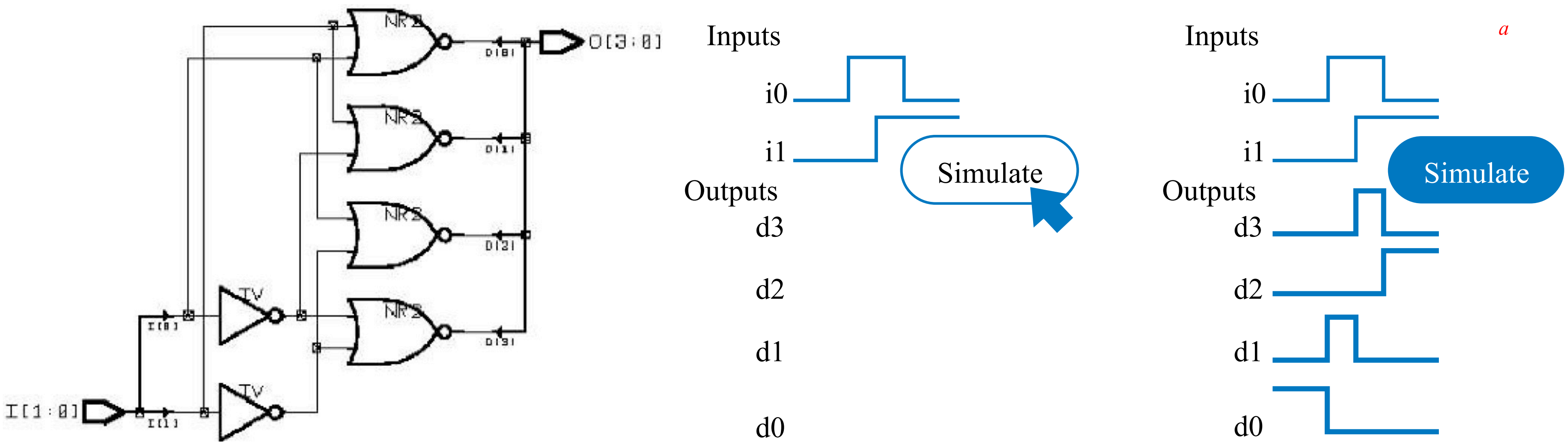
Active Low Inputs

- Data inputs: flow through component (e.g., mux data input)
- Control input: influence component behavior
  - Normally active high – 1 causes input to carry out its purpose
  - Active low – Instead, 0 causes input to carry out its purpose
  - Example: 2x4 decoder with active low enable
    - 1 disables decoder, 0 enables
  - Drawn using inversion bubble



(**a**) (**b**)

Schematic Capture and Simulation



- **Schematic capture**
  - Computer tool for user to capture logic circuit graphically
- **Simulator**
  - Computer tool to show what circuit outputs would be for given inputs
    - Outputs commonly displayed as **waveform**

Appendix: Boolean properties and examples

Boolean Algebra Properties

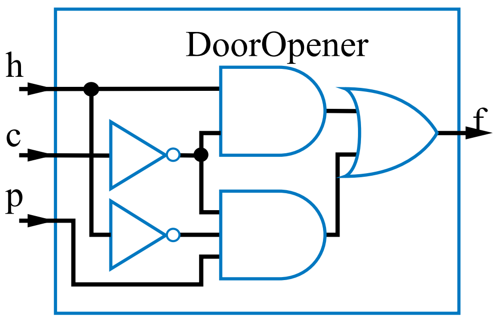
- Commutative
  - $a + b = b + a$
  - $a * b = b * a$
- Distributive
  - $a * (b + c) = a * b + a * c$
    - Can write as: $a(b+c) = ab + ac$
  - $a + (b * c) = (a + b) * (a + c)$
    - (This second one is tricky!)
    - Can write as: $a+(bc) = (ab)(ac)$
- Associative
  - $(a + b) + c = a + (b + c)$
  - $(a * b) * c = a * (b * c)$
- Identity
  - $0 + a = a + 0 = a$
  - $1 * a = a * 1 = a$
- Complement
  - $a + a' = 1$
  - $a * a' = 0$
- To prove, just evaluate all possibilities

Example uses of the properties

- Show abc' equivalent to $c'ba$.
  - Use commutative property:
    - $a*b*c' = a*c'*b = c'*a*b = c'*b*a$
- Show $abc + abc' = ab$.
  - Use first distributive property a
    - $abc + abc' = ab(c+c')$.
  - Complement property
    - Replace $c+c'$ by 1: $ab(c+c') = ab(1)$.
  - Identity property
    - $ab(1) = ab*1 = ab$.
- Show $x + x'z$ equivalent to $x + z$.
  - Second distributive property
    - Replace $x+x'z$ by $(x+x')*(x+z)$.
  - Complement property
    - Replace $(x+x')$ by 1,
  - Identity property
    - replace $1*(x+z)$ by $x+z$.

Example that Applies Boolean Algebra Properties

- Want automatic door opener circuit (e.g., for grocery store)
  - Output: f=1 opens door
  - Inputs:
    - p=1: person detected
    - h=1: switch forcing hold open
    - c=1: key forcing closed
  - Want open door when
    - h=1 and c=0, or
    - h=0 and p=1 and c=0
  - Equation: f = hc' + h'pc'



- Can the circuit be simplified?

$f = hc' + h'pc'$

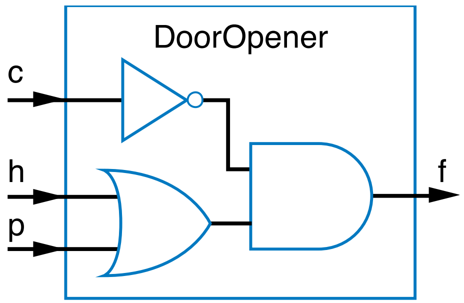
$f = c'h + c'h'p$ (by the commutative property)

$f = c'(h + h'p)$ (by the first distrib. property)

$f = c'((h+h')*(h+p))$ (2nd distrib. prop.; tricky one)

$f = c'((1)*(h + p))$ (by the complement property)

$f = c'(h+p)$ (by the identity property)

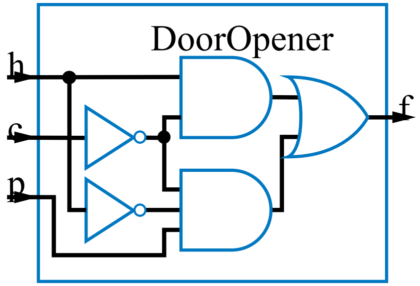


Simplified circuit

Simplification of circuits is covered in Sec. 2.11 / Sec 6.2.

Example that Applies Boolean Algebra Properties

- Found inexpensive chip that computes:
  - f = c'hp + c'hp' + c'h'p
  - Can we use it for the door opener?
    - Is it the same as f = hc' + h'pc'?
- Apply Boolean algebra:



- Commutative
  - a + b = b + a
  - a * b = b * a
- Distributive
  - a * (b + c) = a * b + a * c
  - a + (b * c) = (a + b) * (a + c)
- Associative
  - (a + b) + c = a + (b + c)
  - (a * b) * c = a * (b * c)
- Identity
  - 0 + a = a + 0 = a
  - 1 * a = a * 1 = a
- Complement
  - a + a' = 1
  - a * a' = 0

f = c'hp + c'hp' + c'h'p

f = c'h(p + p') + c'h'p (by the distributive property)

f = c'h(1) + c'h'p (by the complement property)

f = c'h + c'h'p (by the identity property)

f = hc' + h'pc' (by the commutative property)

Same! Yes, we can use it.

Boolean Algebra: Additional Properties

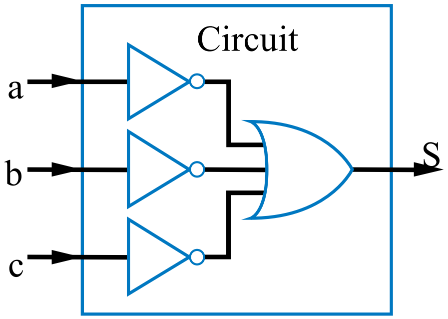
- Null elements
  - a + 1 = 1
  - a * 0 = 0
- Idempotent Law
  - a + a = a
  - a * a = a
- Involution Law
  - (a')' = a
- DeMorgan's Law
  - (a + b)' = a'b'
  - (ab)' = a' + b'
  - *Very useful!*
- To prove, just evaluate all possibilities

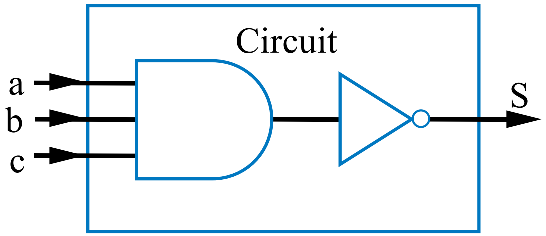
Example Applying DeMorgan's Law

$(a + b)' = a'b'$
$(ab)' = a' + b'$

Aircraft lavatory sign example

- Behavior
  - Three lavatories, each with sensor (a, b, c), equals 1 if door locked
  - Light "Available" sign (S) if any lavatory available
- Equation and circuit
  - $S = a' + b' + c'$
- Transform
  - $(abc)' = a'+b'+c'$ (by DeMorgan's Law)
  - $S = (abc)'$
- New circuit





- Alternative: Instead of lighting "Available," light "Occupied"
  - Opposite of "Available" function

    $S = a' + b' + c'$
  - So $S' = (a' + b' + c')'$
    - $S' = (a')' * (b')' * (c')'$ (by DeMorgan's Law)
    - $S' = a * b * c$ (by Involution Law)
  - Makes intuitive sense
    - Occupied if all doors are locked

Example Applying Properties

- Commutative
  - a + b = b + a
  - a * b = b * a
- Distributive
  - a * (b + c) = a * b + a * c
  - a + (b * c) = (a + b) * (a + c)
- Associative
  - (a + b) + c = a + (b + c)
  - (a * b) * c = a * (b * c)
- Identity
  - 0 + a = a + 0 = a
  - 1 * a = a * 1 = a
- Complement
  - a + a' = 1
  - a * a' = 0
- Null elements
  - a + 1 = 1
  - a * 0 = 0
- Idempotent Law
  - a + a = a
  - a * a = a
- Involution Law
  - (a')' = a
- DeMorgan's Law
  - (a + b)' = a'b'
  - (ab)' = a' + b'

- For door opener f = c'(h+p) , *prove* door stays closed (f=0) when c=1
  - f = c'(h+p)
  - *Let c = 1* (door forced closed)
  - f = 1'(h+p)
  - f = 0(h+p)
  - f = 0h + 0p (by the distributive property)
  - f = 0 + 0 (by the null elements property)
  - f = 0

Complement of a Function

- Commonly want to find complement (inverse) of function F
  - 0 when F is 1;   1 when F is 0
- Use DeMorgan's Law repeatedly
  - Note: DeMorgan's Law defined for more than two variables, e.g.:
    - $(a + b + c)' = (abc)'$
    - $(abc)' = (a' + b' + c')$
- Complement of $f = w'xy + wx'y'z'$
  - $f\,' = (w'xy + wx'y'z')'$
  - $f\,' = (w'xy)'(wx'y'z')'$ (by DeMorgan's Law)
  - $f\,' = (w+x'+y')(w'+x+y+z)$ (by DeMorgan's Law)
- Can then expand into sum-of-products form