Simple Cost and Performance Model

Simple Model Goals

Choose between design alternatives.

Characterize a particular design approach.

Simple Model Non-Goals

Approximate the post synthesis cost and performance.

The Cost Model

Base Costs

2-input AND gate: 1 unit.

2-input OR gate: 1 unit.

NOT gate: 0 units. (Zero cost.)

Derived Costs

Based on equivalent circuit using gates above.

Cost of n-input OR gate: $n - 1$ units.

Cost of n-input AND gate: $n - 1$ units.

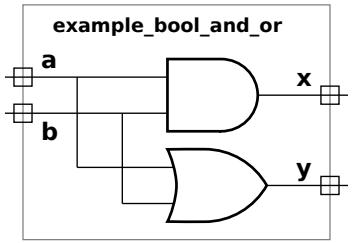
Cost of a 2-input XOR gate is 3 units.

Simple Examples:

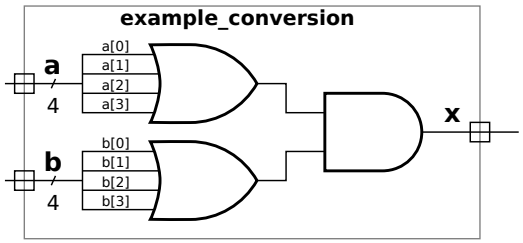
A 2-input AND gate: cost is 1 unit.

A 10-input OR gate: cost is 9 units.

A 1-input OR gate: free.



: cost is $1 + 1 = 2$ units.



: cost is $3 + 3 + 1 = 7$ units.

The Performance Model

Base Delays

2-input AND gate: 1 unit.

2-input OR gate: 1 unit.

NOT gate: 0 units.

Derived Delays

Based on equivalent circuit using gates above.

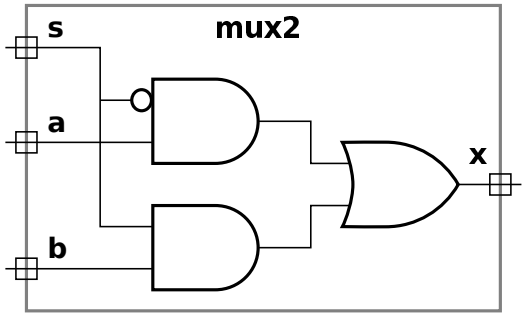
Delay of a 2-input XOR gate is 2 units.

Delay of n-input OR gate: $\lceil \lg n \rceil$ units.

Delay of n-input AND gate: $\lceil \lg n \rceil$ units.

Multiplexors

A 2-input, 1-bit mux:



Cost, 3 units; delay, 2 units.

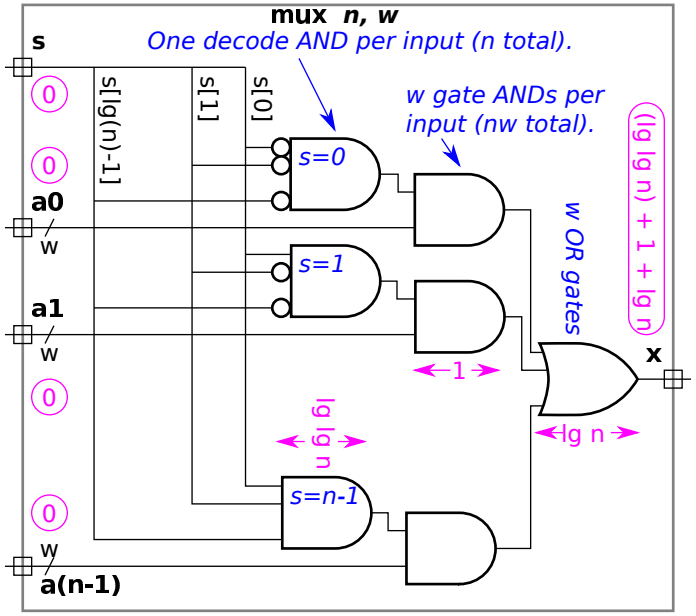
A 2-input, w-bit mux:

This is equivalent to w copies of the mux above.

Cost, $3w$ units; **delay, 2 units**.

Multiplexors

An n-input, w-bit mux,
unoptimized decoder implementation:



Note: The number of select bits is $\lceil \lg n \rceil$.

Cost Computation:

Decoder AND gates (first column): $n(\lceil \lg n \rceil - 1)$.

Selection AND gates (second column): wn.

OR gate: $w(n-1)$.

Total cost: $n(\lceil \lg n \rceil - 1) + 2wn - w$.

cost-6

Delay Computation

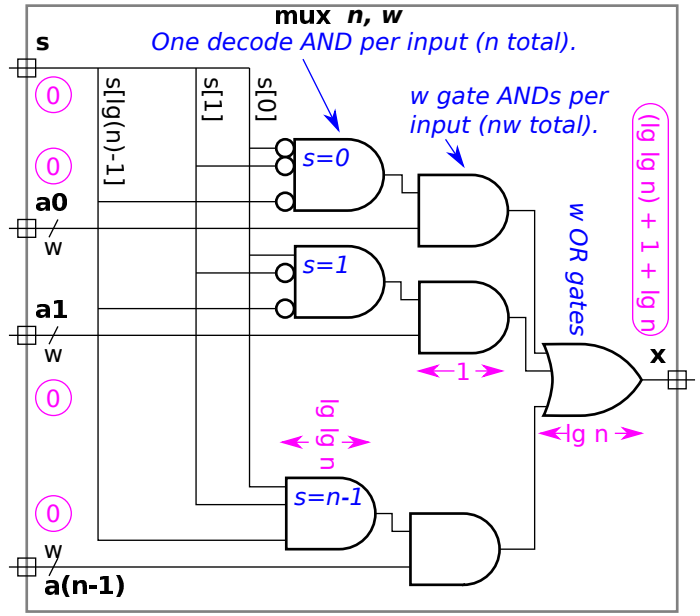
Decoder AND gate (first column): $\lceil \lg(\lceil \lg n \rceil - 1) \rceil$.

Selection AND gates (second column): 1.

OR gate: $\lceil \lg n \rceil$.

Total delay: $\lceil \lg(\lceil \lg n \rceil - 1) \rceil + 1 + \lceil \lg n \rceil$.

Approximate delay: $1 + \lceil \lg n \rceil$.



An n-input, w-bit mux, tree implementation:

Constructed from 2-input multiplexors.

Illustration is for $n = 8$.

The path from the selected input ...
... is through $\lceil \lg n \rceil$ 2-input muxen ...
... through 3 for illustrated size, $n = 8$.

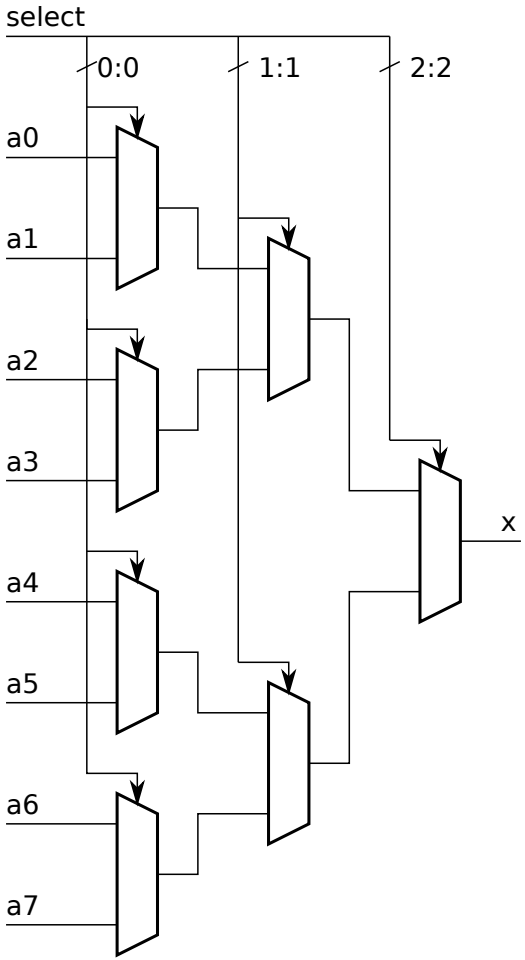
The number of muxen connected to select bit i...
... is $n/2^{i+1}$ for $0 \geq i < \lceil \lg n \rceil$...
... for illustrated size 2 muxen connect to bit 1.

Cost Computation

Total number of 2-input muxen ...
... $\sum_{i=0}^{\lg n-1} n/2^{i+1} = n - 1$ ...
... for illustrated mux, 7 2-input muxen.

Total cost: $3w(n-1)$.

Total Delay: $2\lceil \lg n \rceil$.



Equality Comparison (`a == b`)

Output is `1` iff w-bit inputs are equal.

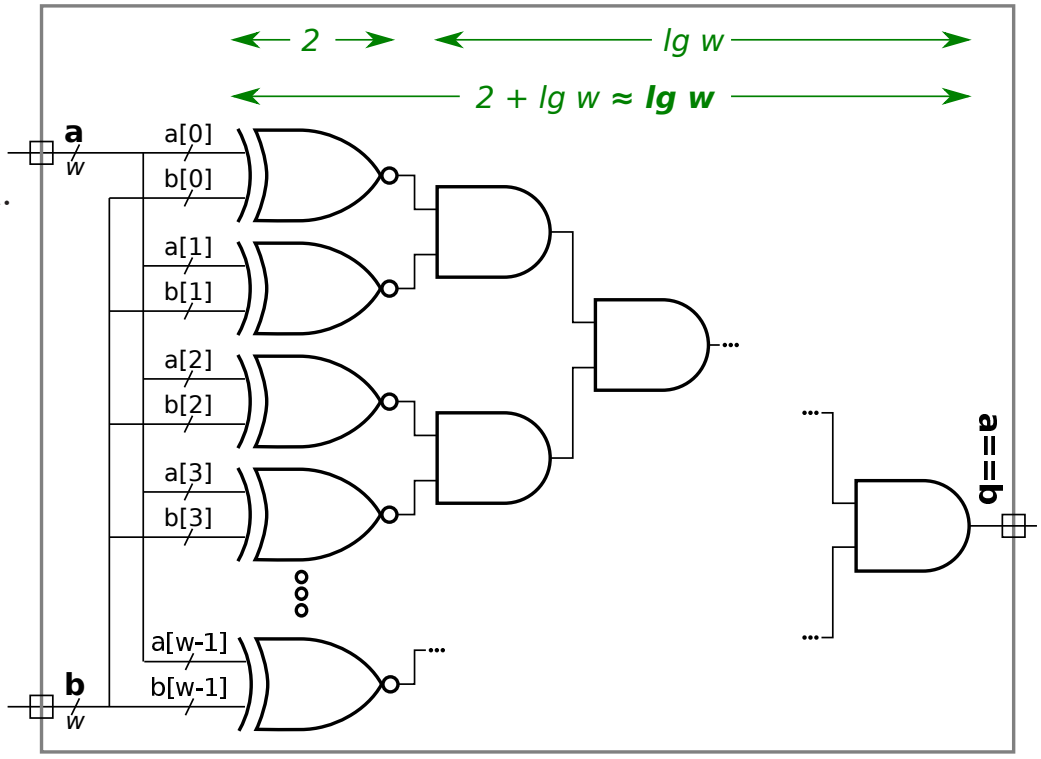
Assume w is a power of 2.

Cost Computation

XOR Gates: $3w$.

Reduction tree of AND gates: $\sum_{i=1}^{\lg w} w/2^i = w - 1$.

Total Cost: $4w - 1 \approx 4w$.



Delay Computation

XOR Gates: 2.

Reduction Tree: $\lg w$.

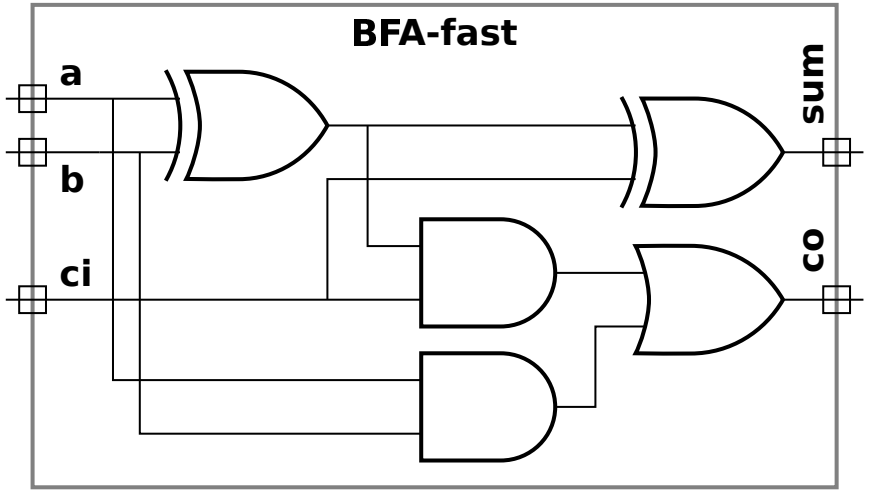
Total Delay: $2 + \lg w \approx \lg w$.

Binary Full Adder Implementations

Fast BFA

Cost Computation

Two XOR, 3 AND: $2 \times 3 + 3 = 9$ units.



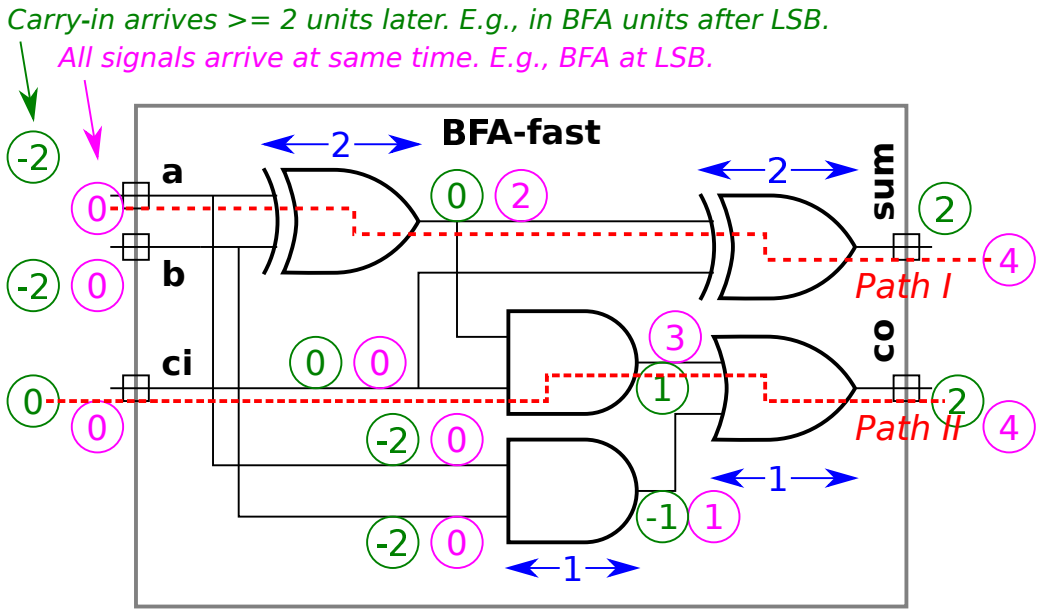
cost-10

Fast BFA

Delay: Any input to any output.

See Path I and...
... purple labels in diagram.

Delay: 4 units.



Delay: `ci` to `co`.

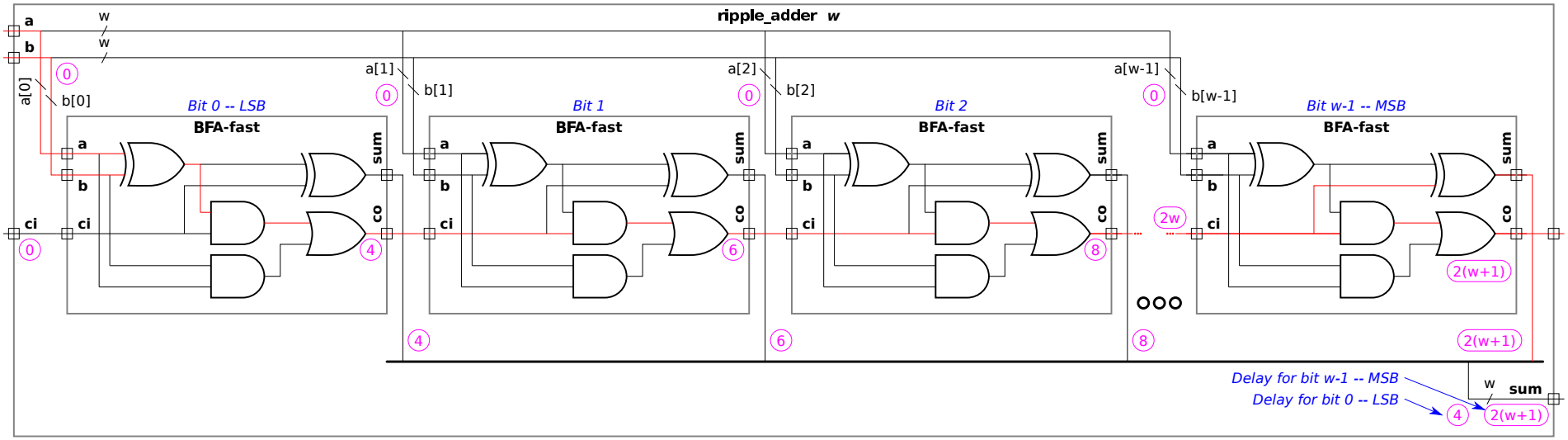
This delay is useful when `a` and `b` arrive earlier than `ci`.

See Path II and green labels in diagram.

Delay: 2 units.

w-Bit Ripple Adder

Implementation using Fast BFA



Cost Computation: Cost of w BFAs: $9w$ units.

Delay Computation

See critical path (in red) in diagram.

Delay: $2(w+1) \approx 2w$.

cost-12

Integer Magnitude Comparison

For comparisons like $a < b$.

Implementation:

Compute $a - b$ and check whether result negative.

If carry out of MSB is `0` then $a - b < 0$ and so $a < b$ is true.

Omit sum hardware in BFA, and replace remaining XOR with an OR.

Cost Computation

Each modified BFA now costs 4 units.

Total cost: $4w$ units.

Delay Computation

Delay is 3 for first bit, 2 for remaining bits.

Total delay: $2w + 1 \approx 2w$.

Cascaded Ripple Units

For computations using ripple units...
... such as $a + b + e$, and $(a + b) < e$, etc.

Cost Computation

Cost is sum of costs of each ripple unit.

For example, $a + b + c$ is two ripple adders, cost is $18w$ ...
... $(a + b) < e$ is a ripple adder plus a magnitude comparison: $9w + 4w = 13w$.

LSU EE 4755 Lecture Transparency. Formatted 10:42, 22 November 2017 from lsli-cost-model.
cost-14

Cascaded Ripple Units

Delay Computation:

Consider $(a + b) + e$.

Naïve analysis: wait for $a + b$ to finish, then start $+e$.

But, LSB of $a + b$ available after only 4 cycles ...
... bit i is available after $4 + 2i$ cycles ...
... so the $+e$ computation can start after 4 cycles.

Delay for two ripple units is $4 + 2(w + 1)$.

Delay for n ripple units is $4(n - 1) + 2(w + 1)$.

Cost and Performance with Constant Inputs

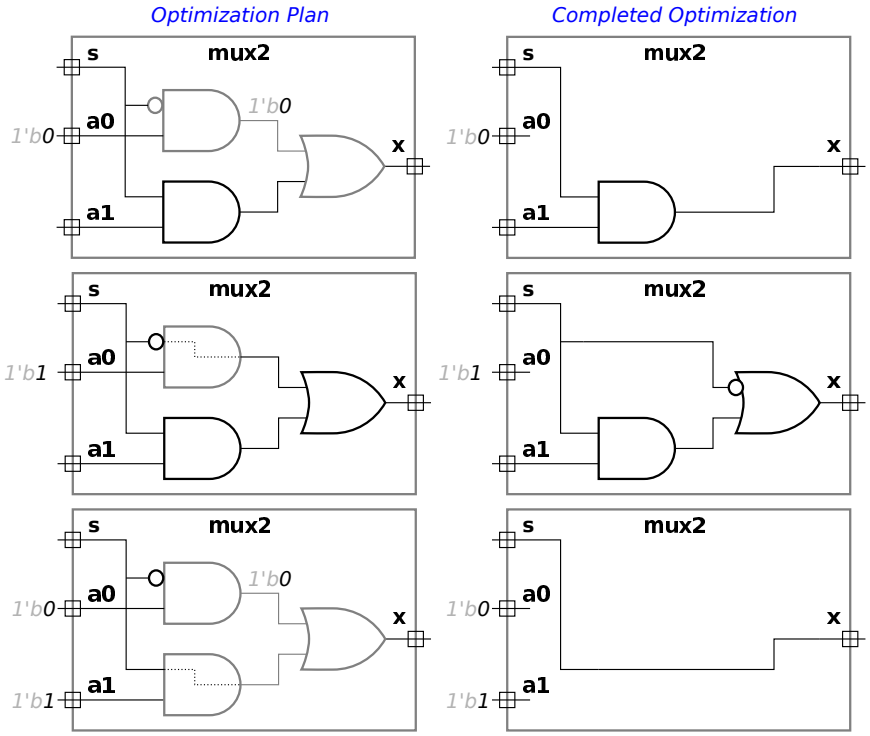
Constant Inputs

Signal values which never change.

Cost and delay are radically different when an input never changes.

In Verilog, this might be an elaboration-time constant . . .
. . . or other expressions that never change.

Multiplexor Constant-Input Optimizations



Sample mux optimizations:

Costs: From top to bottom costs are: 1 unit, two units, zero units.

Delays: From top to bottom delays are: 1 unit, two units, zero units.