Digital Logic Design: a rigorous approach ©

Chapter 14: Selectors

Guy Even Moti Medina

School of Electrical Engineering Tel-Aviv Univ.

April 10, 2012

Book Homepage:
http://www.eng.tau.ac.il/~guy/Even-Medina

Preliminary questions:

1. Which types of shifts are you familiar with in your favorite programming language? What is the differences between these shifts? Why do we need different types of shifts?
2. How are these shifts executed in a microprocessor?
3. Should shifters be considered to be combinational circuits? After all, they simply "move bits around" and do not compute "new bits".

Multiplexer (MUX)

Definition

A MUX-gate is a combinational gate that has three inputs $D[0], D[1], S$ and one output Y. The functionality is defined by

$$Y = \begin{cases} D[0] & \text{if } S = 0 \\ D[1] & \text{if } S = 1. \end{cases}$$

Note that we could have used the shorter expression $Y = D[S]$ to define the functionality of a MUX-gate.

n-bit selector

Definition

An (n:1)-MUX is a combinational circuit defined as follows:

Input: data input $D[n-1:0]$ and select input $S[k-1:0]$ where $k = \lceil \log_2 n \rceil$.

Output: $Y \in \{0,1\}$.

Functionality:

$$Y = D[\langle \vec{S} \rangle].$$

To simplify the discussion, we will assume in this chapter that n is a power of 2, namely, $n = 2^k$.

Example

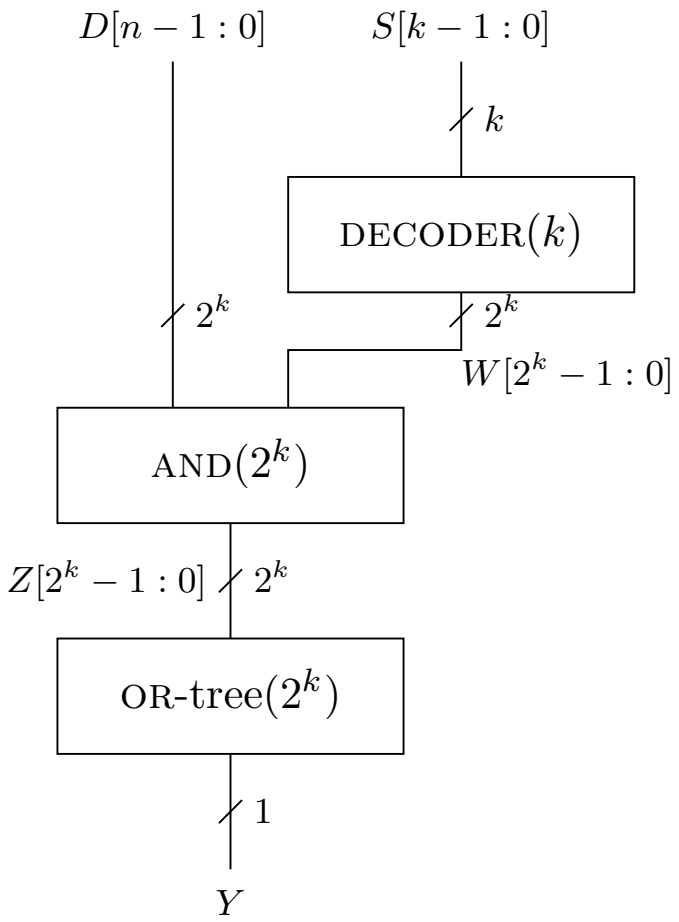
Let $n = 4$ and $D[3:0] = 0101$. If $S[1:0] = 00$, then $Y = D[0] = 1$. If $S[1:0] = 01$, then $Y = D[1] = 0$.

Implementation

We describe two implementations of (n:1)-MUX.

- translate the number $\langle \vec{S} \rangle$ to 1-out-of-n representation (using a decoder).
- tree based.

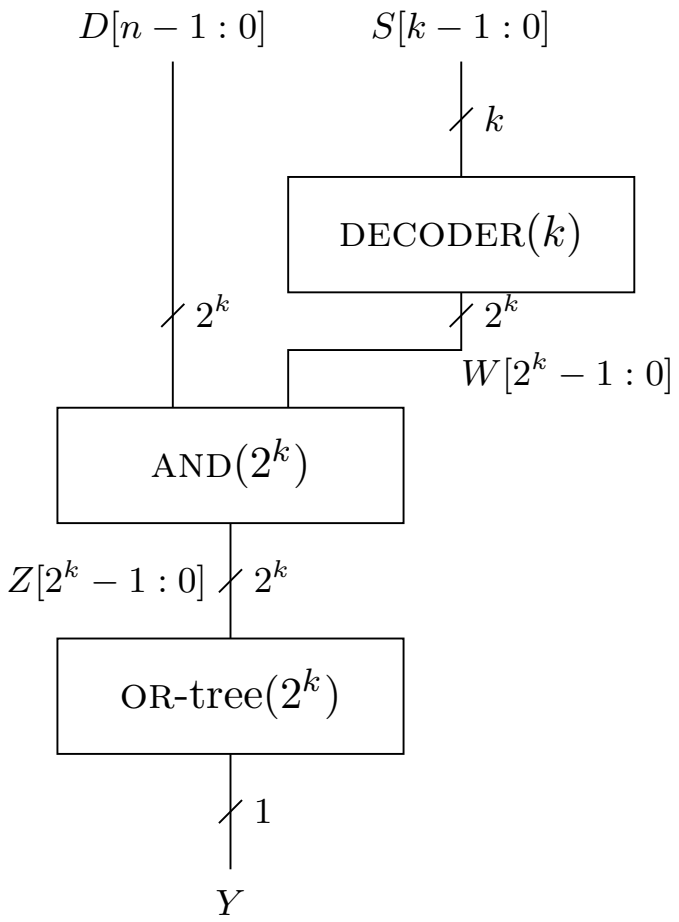
decoder based (n:1)-MUX



Claim

The (n:1)-MUX design is correct.

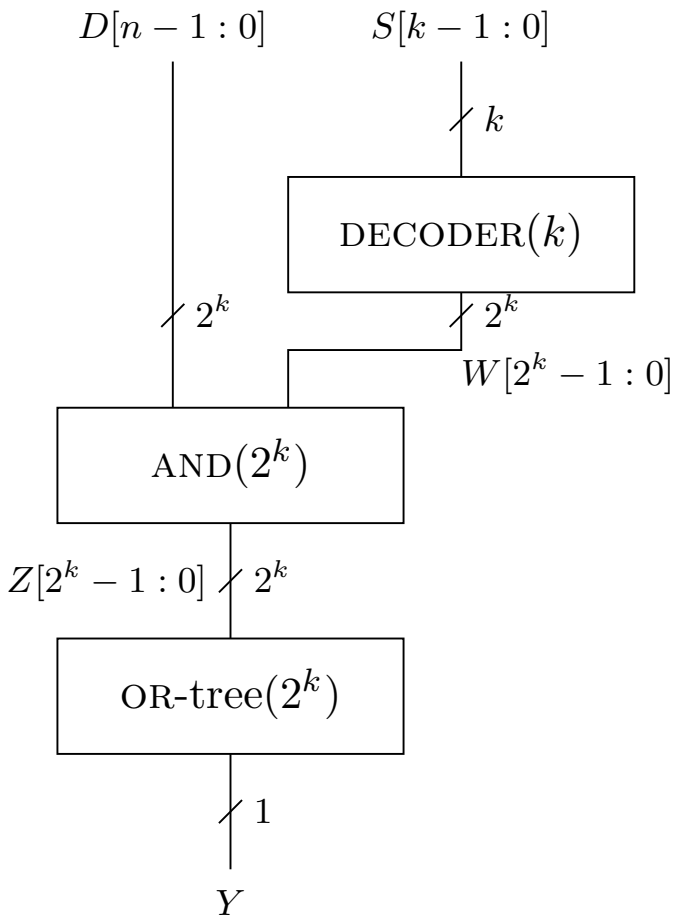
decoder based (n:1)-MUX - cost



Claim

The cost of the (n:1)-MUX design is $\Theta(n)$.

decoder based (n:1)-MUX - delay



Claim

The delay of the (n:1)-MUX design is $\Theta(\log n)$.

(n:1)-MUX - lower bounds

Claim

The cone of the Boolean function implemented by a $(n:1)$*-*MUX *circuit contains at least n elements.*

Consider combinational circuits with with at most two input terminals per gate (or any constant).

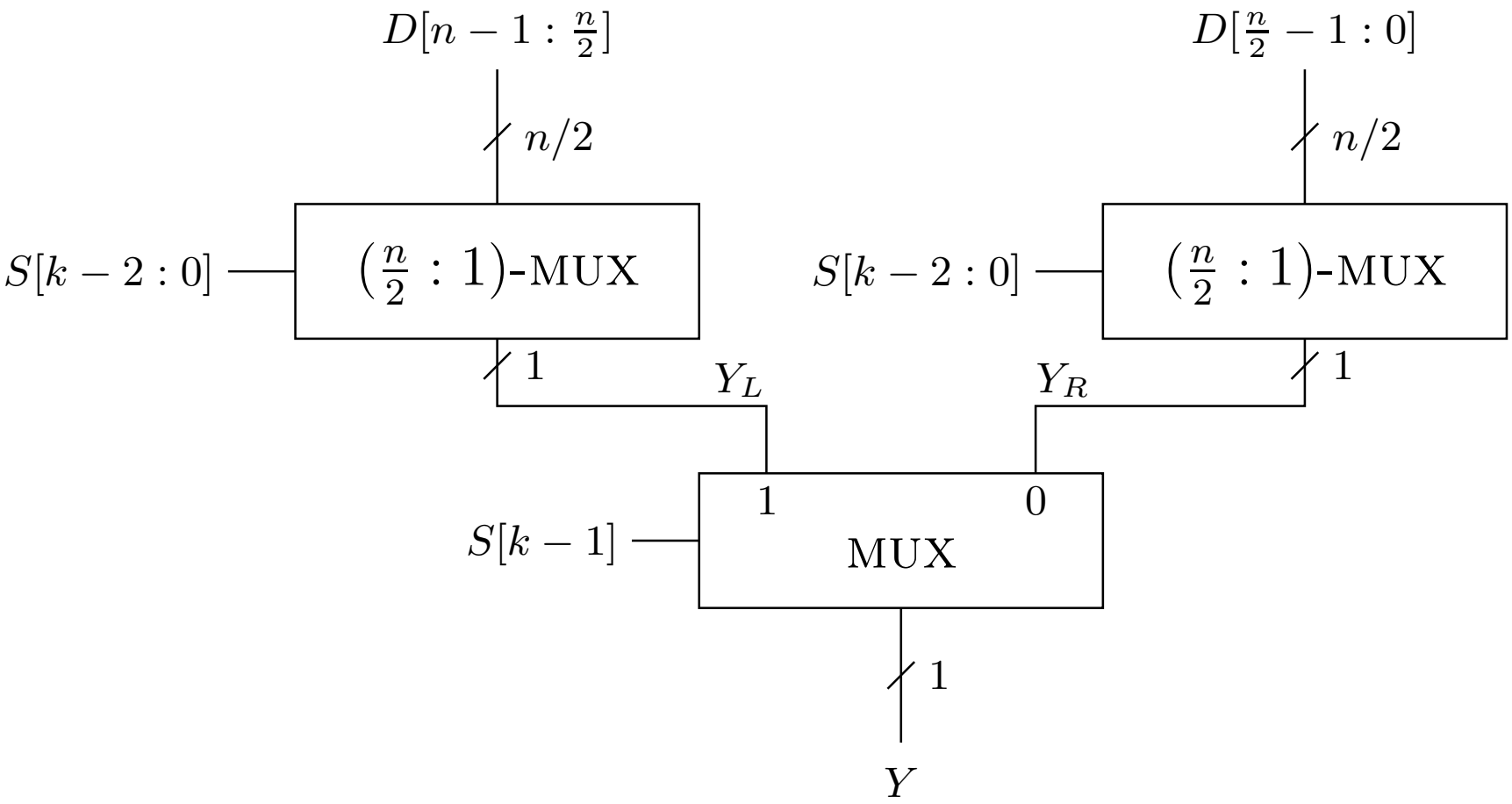
Corollary

*The cost of the (n:1)-*MUX *design is asymptotically optimal.*

Corollary

*The delay of the (n:1)-*MUX *design is asymptotically optimal.*

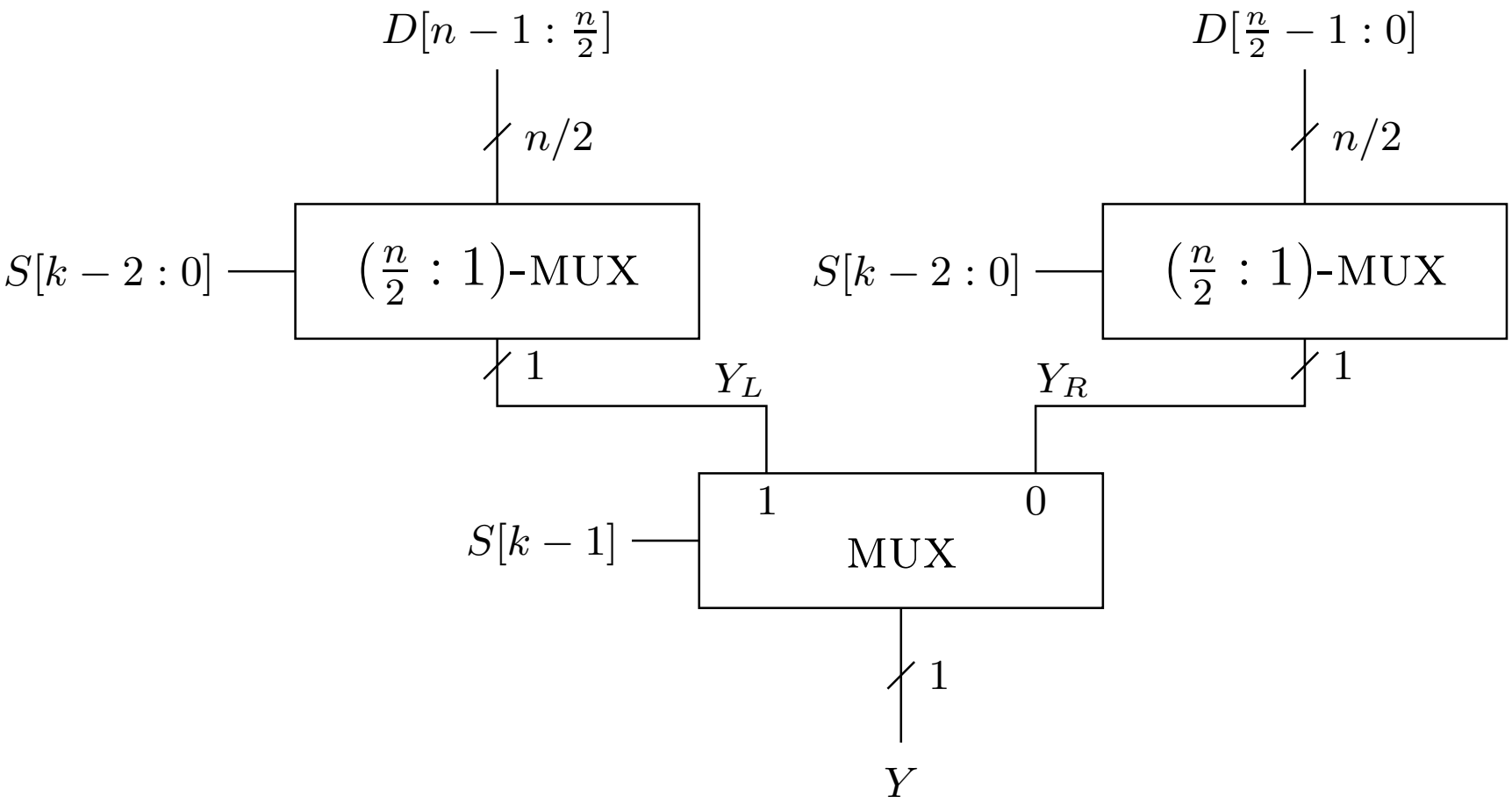
tree based (n:1)-MUX



Claim

The (n:1)-MUX design is correct.

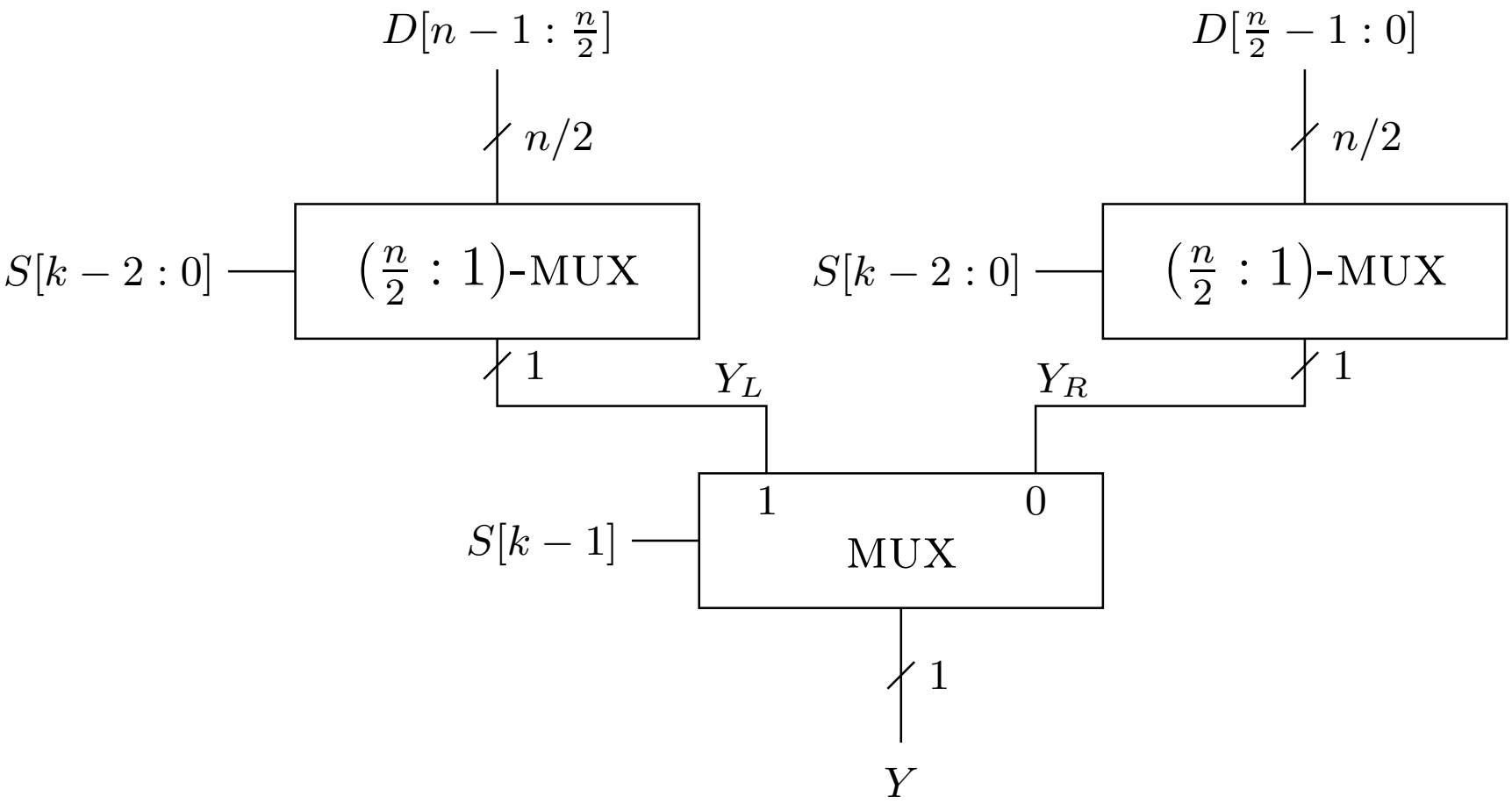
tree based (n:1)-MUX - cost



Claim

The cost of the (n:1)-MUX design is $\Theta(n)$.

tree based (n:1)-MUX - delay



Claim

The delay of the (n:1)-MUX design is $\Theta(\log n)$.

Comparison

- Both implementations are asymptotically optimal with respect to cost and delay.
- The cost/delay table suggests that the tree-like implementation is cheaper and faster.
- Fast and cheap implementations of MUX-gates in CMOS technology do not restore the signals well. This means that long paths consisting only of MUX-gates are not allowed.
- What about physical layout? Which design has a smaller "drawing"?
- Conclusion: our simplified model cannot be used to deduce conclusively which multiplexer design is better.