WE5.1

Bookmark this page

Calculator

1.7 + 1.7 + 1.6 [ ] 5.0

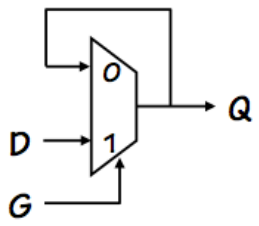
Calculator Instructions

Latch Implementation

3/3 points (ungraded)

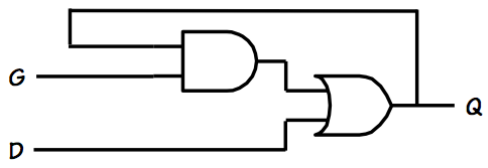
Untel, Inc is a startup exploring a new gate technology that has hired you as a consultant. They have learned how to make reliable, lenient AND gates, OR gates, and inverters, but don't yet have a cell library offering devices like multiplexors. Their current crisis, for which they need your help, is the design of a reliable latch.

The Untel engineers vaguely remember a 6.004 lecture showing how to make a latch using a lenient multiplexor (as shown below), and reason that they can make a latch at least as good starting from AND/OR/inverter logic.

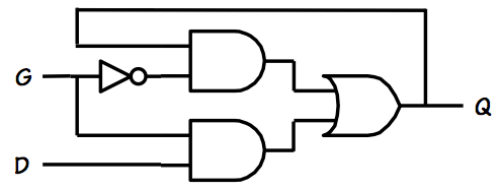


There are three different proposals being considered:

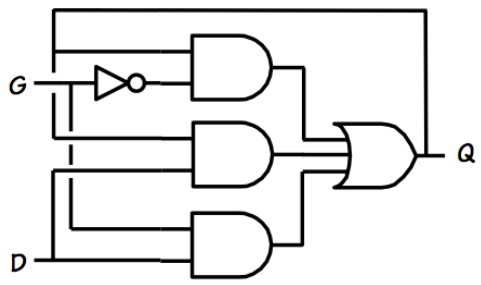
Proposal A



Proposal B



Proposal C



The Untel CTO shows you the diagrams, and asks you characterize each as

- BAD, meaning it doesn't work reliably;
- GOOD, meaning that it works reliably (given appropriate dynamic discipline rules); or
- OBESE, meaning that it works but uses more gates than necessary

Characterize each of the above proposals.

Proposal A

- (•) BAD
- ( ) GOOD
- ( ) OBESE

✔

Proposal B

- (•) BAD
- ( ) GOOD

Calculator

Calculator Instructions

OBESE

✔

Proposal C

BAD

GOOD

OBESE

✔

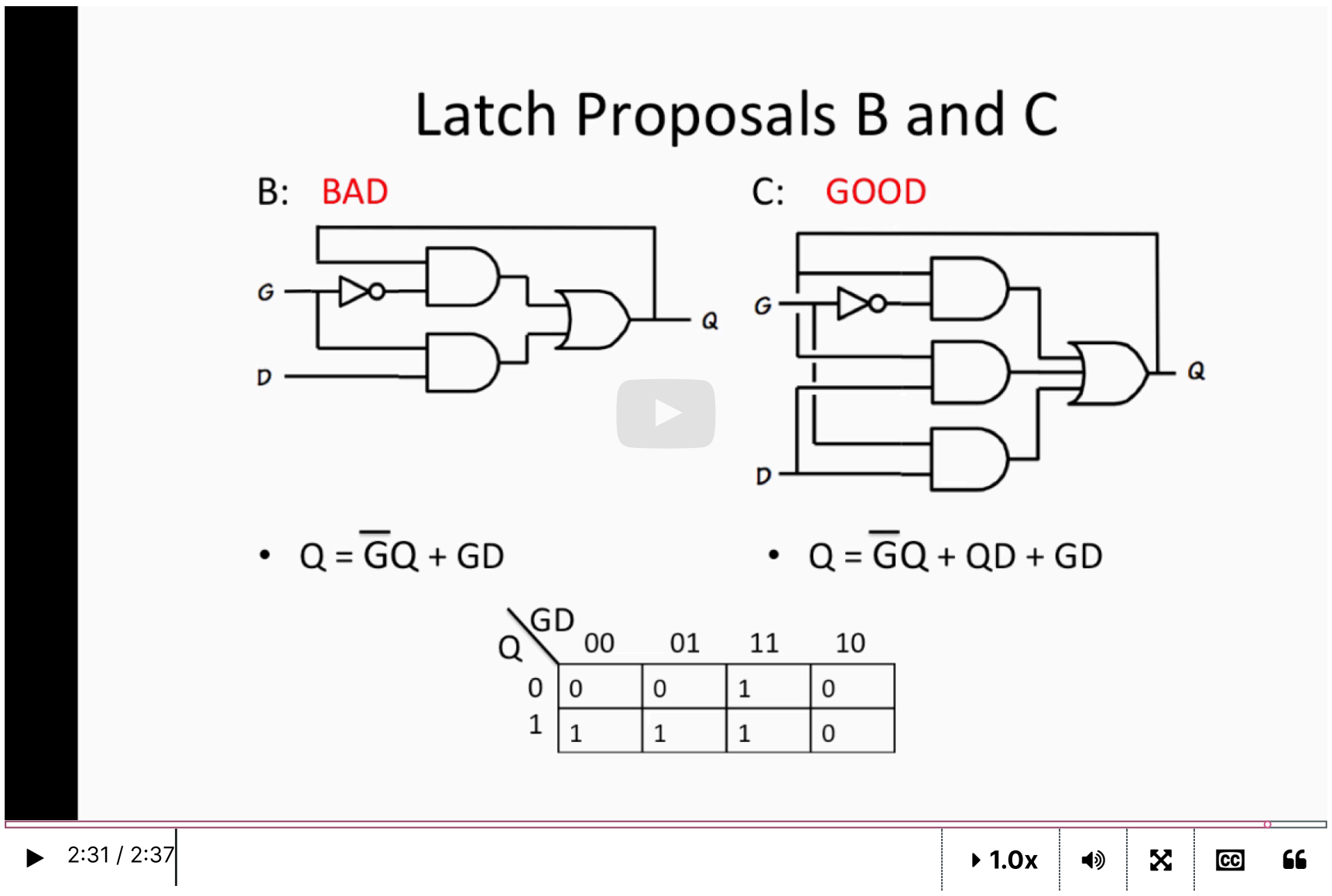
Explanation
A design is good if it is a lenient multiplexer (mux).

- The boolean logic of proposal A is not a equivalent to that of a mux, so proposal A is BAD.
- Proposal B is a mux but isn't lenient, so it is BAD.
- Proposal C is a lenient mux, so it is GOOD.

Submit

Answers are displayed within the problem

Latch Implementation



▶ 2:31 / 2:37 1.0x

Video
Download video file

Transcripts
Download SubRip (.srt) file
Download Text (.txt) file

Calculator

Discussion

Calculator Instructions

Topic: 5. Sequential Logic / WE5.1

Add a Post

All Posts

method to come up with lenient circuit?

question posted 8 years ago by **sajedula**

Unfortunately, the video does not explain why proposal B is not lenient, but simply states so. My understanding was that the minimal logic from K-map always gives a lenient circuit, but apparently that's not the case. In any case is there a methodical way to to come up with Proposal C, which is lenient?

This post is visible to everyone.

silvinahw (Staff)
8 years ago - marked as answer 8 years ago by **silvinahw** (Staff)

Take a look at this lecture video at time 10:45 for an explanation of how to ensure that your circuit is lenient.

Show Comment (1)

Preview

Submit

Previous Next

Your Privacy Choices

Connect

Idea Hub
Contact Us
Help Center
Security
Media Kit

GET IT ON Google play
Download on the App Store

Calculator

Calculator Instructions