Engs 31 / CoSc 56

DIGITAL ELECTRONICS

Day 24

Today: How to manage and document your project

25.1

The project schedule

Project demonstrations are Monday-Wednesday, August 23-25.

- In-person, unless sick
- On Zoom, or make a video and discuss on Zoom

Turn final written report in to Canvas by

Tuesday, August 31, 11:59 PM (EDT)

Last chance for demonstrating a working project is also

Tuesday, August 31, 5:00 PM (EDT)

25.2

Project management (1) — Planning and design

Design from the top down

- You have the most leverage at the beginning, while the design is still on paper. Once you start building, you're committed and it's harder to make changes.
- The real creative design is at the RTL / state diagram level. VHDL should simply be a translation of the design.

Implement from the bottom up

- Divide the build into modules: block diagram becomes a checklist.
- Implement and test, module-by-module.
- Simulation **and** physical test.
- Update documents, then (and only then) check it off the list.

25.6

Project management (2)— Documentation

Don't leave documentation to the final report, **keep a current collection as you go.**

- RTL and state diagrams
- VHDL sources
- Elaborated schematics
- Simulation waveforms
- Utilization summary

Use your shared folder to store your document set.

25.11

General advice

1. Keep the Digital Lab clean.
2. Store your project in a locker to avoid accidents and to make space for others. Don't hoard the bench space — there are 18 groups!
3. If you don't need the test equipment, consider working in M210 or C013.
4. Ask for help when needed
   - Your coach can help with general questions, point you to resources.
   - Ben Dobbins, Prof Hansen, TAs can help with debugging when you're stuck.
5. Need parts? — see Ben
6. Strive for clean VHDL, and maintain backups of your VHDL sources.
7. Report computer system failures *immediately* to me, to Ben, to a TA, or (after hours) to computing@thayer.dartmouth.edu.
8. Do the online course evaluation (you will get an email about this).

25.13

Project management (3): version control

Keep a debugging log to track problems and solutions.

Use the Revisions: field in your VHDL file header to track major design changes.

Keep old copies of your code so that you can always go back to something that worked.
- Xilinx software doesn't have built in version control
- However, Google Drive keeps your old versions around, so an easy way is to back up your VHDL sources to a Google Drive folder.

If you have access to a version control system (Git, Subversion, etc), the things to track are:
- The project (.xpr) file
- Anything you created with a text editor (.vhd, .xdc, .coe)
- Simulator waveform files (.wfcg)
- Core generator files (.xci)

Everything else can be recreated by synthesis.

25.21

Project management (4)— Using internet code

Code you find on the web should be avoided, because:

- It may be buggy and/or badly written (e.g., some code at the Digilent website).
- Some of the VHDL seen on the Web was written for simulation but not synthesis.
- It may be using the non-standard libraries std_logic_arith, etc — poor practice.

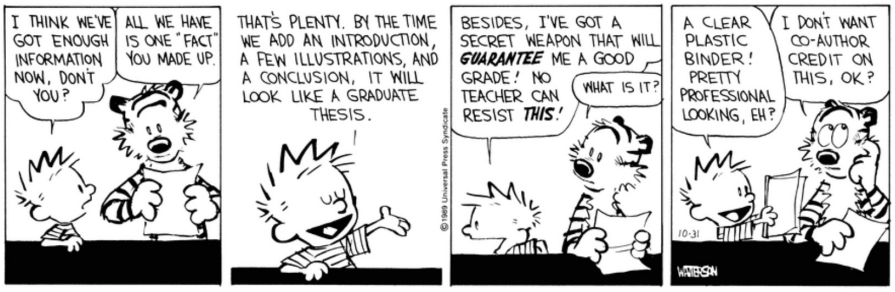
If you find code that you'd like to copy or adapt for a specific purpose (like binary-BCD conversion), show it to me. Then, if you use it, cite the source in your report.

25.23

And then, success

25.27

And after that, the final report



25.35

Final report — main body (see handout for details)

Abstract

Table of Contents

1. Introduction, problem statement — why did we do this?
2. Design solution
   - 2.1 Specifications — what it does; include clock speed, audio sampling rate, video resolution, etc, as applicable.
   - 2.2 Operating instructions — how you use it, illustrated with annotated photo of controls and display
   - 2.3 Theory of operation — how it works, illustrated with:
     - Top-level block diagram
     - Datapath (block diagram)
     - Controller (state diagram)
3. Evaluation of your design — how well did it work?
4. Conclusions and recommendations — what would we do differently?
5. Acknowledgments
6. References
7. Appendices

25.36

Final report — appendices (see handout for details)

Basic idea: What is necessary for someone to reproduce your design?

1. System level diagrams
   - Block diagram — from top-level to more detailed
   - Package map—(photo of FPGA board with callouts to buttons, switches, lights, etc)
2. Programmed logic information
   - State diagrams for state machines
   - VHDL code, organized hierarchically (top level file first, then components, testbenches)
   - Resource utilization (excerpted from implementation report)
   - Analysis of residual warnings (from synthesis) — there should be very few!
3. Memory map, for ROM or RAM, if used — how data are formatted and stored
4. Simulation or oscilloscope waveforms of important features (you decide, remember to annotate the important features)

25.38

Final report — document as you go

Avoid the last-minute crunch!

1. Text — begin with your proposal and design review descriptions.
2. Keep a folder or binder with current drawings and listings, and keep it updated.
3. When you complete a simulation, format the waveform display well, print it out and annotate it.
4. VHDL code

   Keep backups.

   Comments!

   Eventually, get rid of orphaned code fragments.

   Use the "Revisions" field of the header to say what's been changed from version to version.

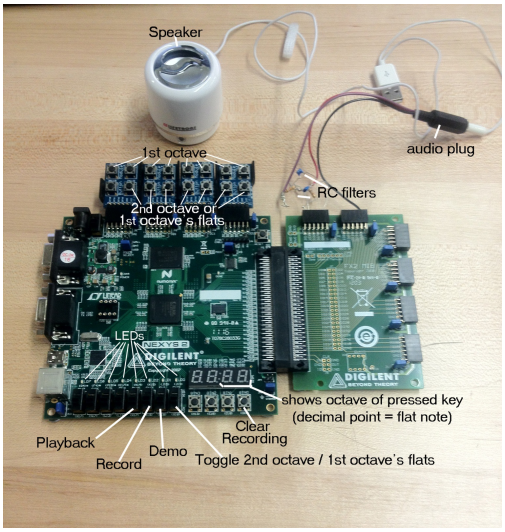
   Check your synthesis warnings.

25.39

Final report: Front panel

Front panel is an "aerial view" of your FPGA board, showing where everything is.

OK to annotate a digital photo.

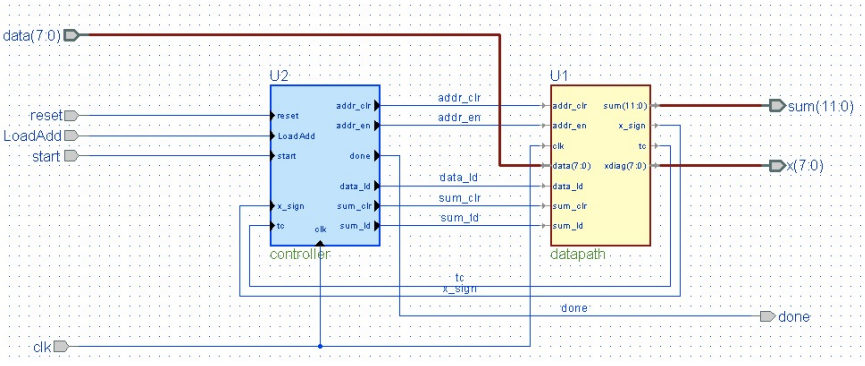


25.40

Final report: Block diagram

Top level. Inputs at left, outputs at right. All "nets" (connecting wires) are labeled with signal names.

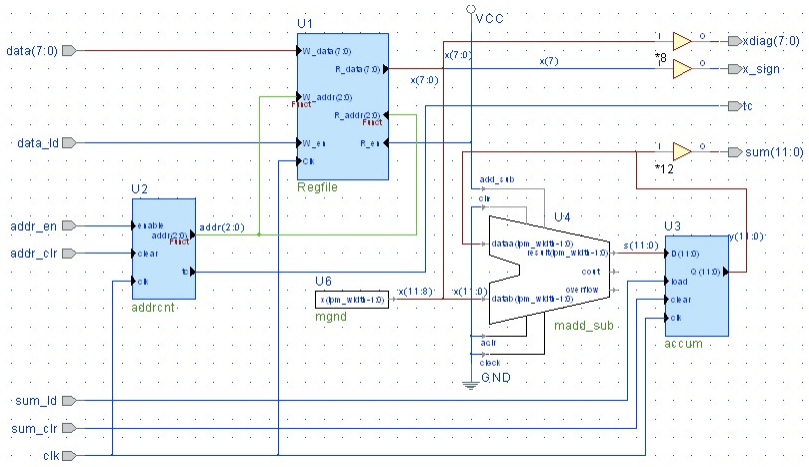
"RTL view" schematics from Vivado might be OK if they are readable when printed on one page --- frequently, they aren't. If in doubt, ask Prof Hansen.



25.41

Final report: Block diagram

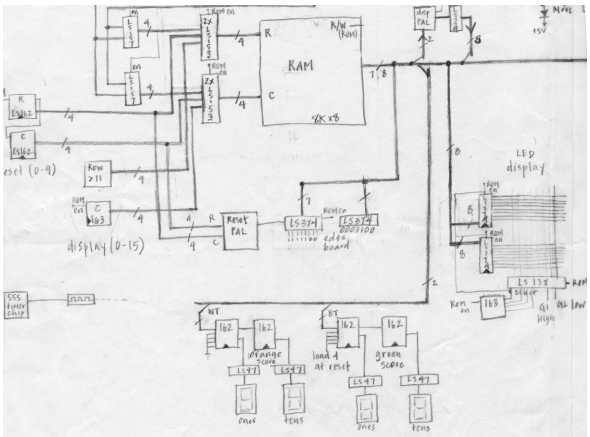
Expand hierarchical blocks. Inputs at left, outputs at right. Nets are labelled.



25.42

Hand-drawn block diagram

This block diagram is laid out OK, but drawn in pencil and hard to read. Better to use ink and draw on graph paper with a ruler (or use a drawing program if you're already skilled with it).



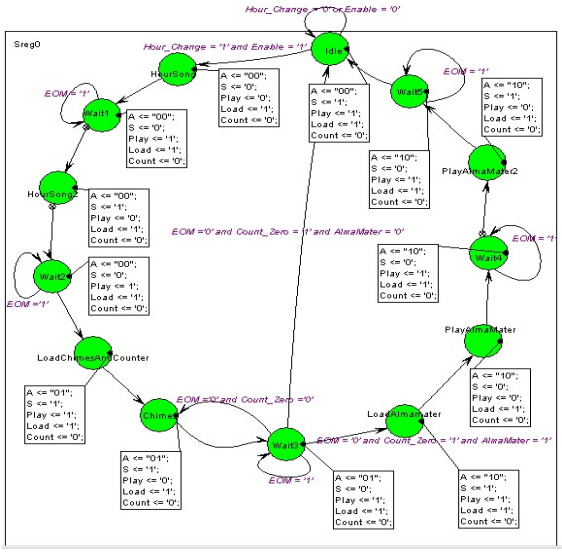
Online source of graph paper PDFs: http://incompetech.com/graphpaper/square.html

25.44

Final report: State diagrams

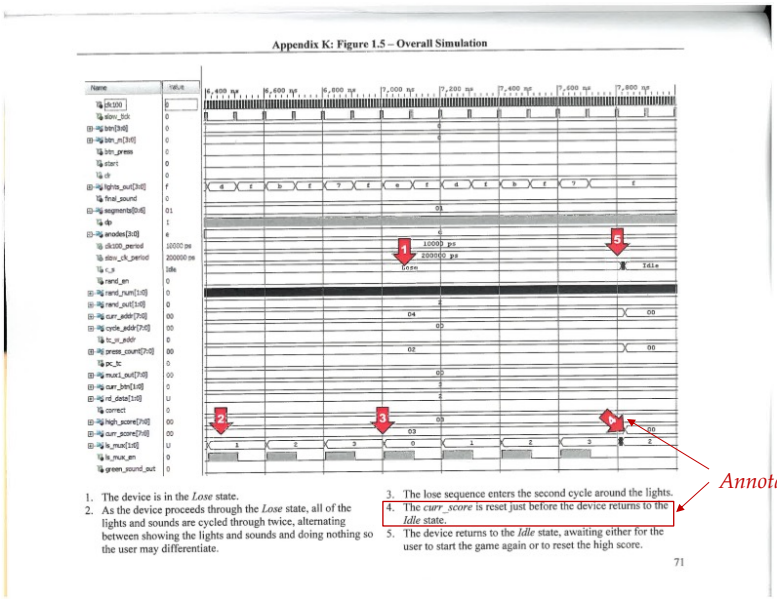
Draw neatly by hand, or use a drawing program.

(What did this project do?)



25.49

Final report: simulation waveforms



25.51

Final report: FPGA utilization

Overall resource utilization — gleaned from log files.

Design Summary:

Number of errors:	0	
Number of warnings:	2	
Logic Utilization:		
Number of Slice Flip Flops:	55 out of 3,840	1%
Number of 4 input LUTs:	52 out of 3,840	1%
Logic Distribution:		
Number of occupied Slices:	51 out of 1,920	2%
Number of Slices containing only related logic:	51 out of 51	100%
Number of Slices containing unrelated logic:	0 out of 51	0%
Number of 4 input LUTs:	88 out of 3,840	2%
Number used as logic:	52	
Number used as a route-thru:	36	
Number of bonded IOBs:	49 out of 173	28%
Number of GCLKs:	1 out of 8	12%
Total equivalent gate count for design:	1,007	
Additional JTAG gate count for IOBs:	2,352	

Device Utilization Summary:

Number of BUFGMUXs	1 out of 8	12%
Number of External IOBs	49 out of 173	28%
Number of LOCed IOBs	49 out of 49	100%
Number of Slices	51 out of 1920	2%
Number of SLICEMs	0 out of 960	0%

25.56

Final report: Residual warnings

Ideally, your synthesis should be warning-free, but it is unlikely. Include the list of warning messages from synthesis, and explain why they are not a problem.

- Latches are always a problem. Your final design must have no latches!
- Signals "used but not on sensitivity list" are always a problem and must be fixed.
- Signals "assigned but not used" may or may not be a problem. You should know.
- Any other warnings you get along the way — see me.

You may ignore warnings from the map and place-and-route steps.

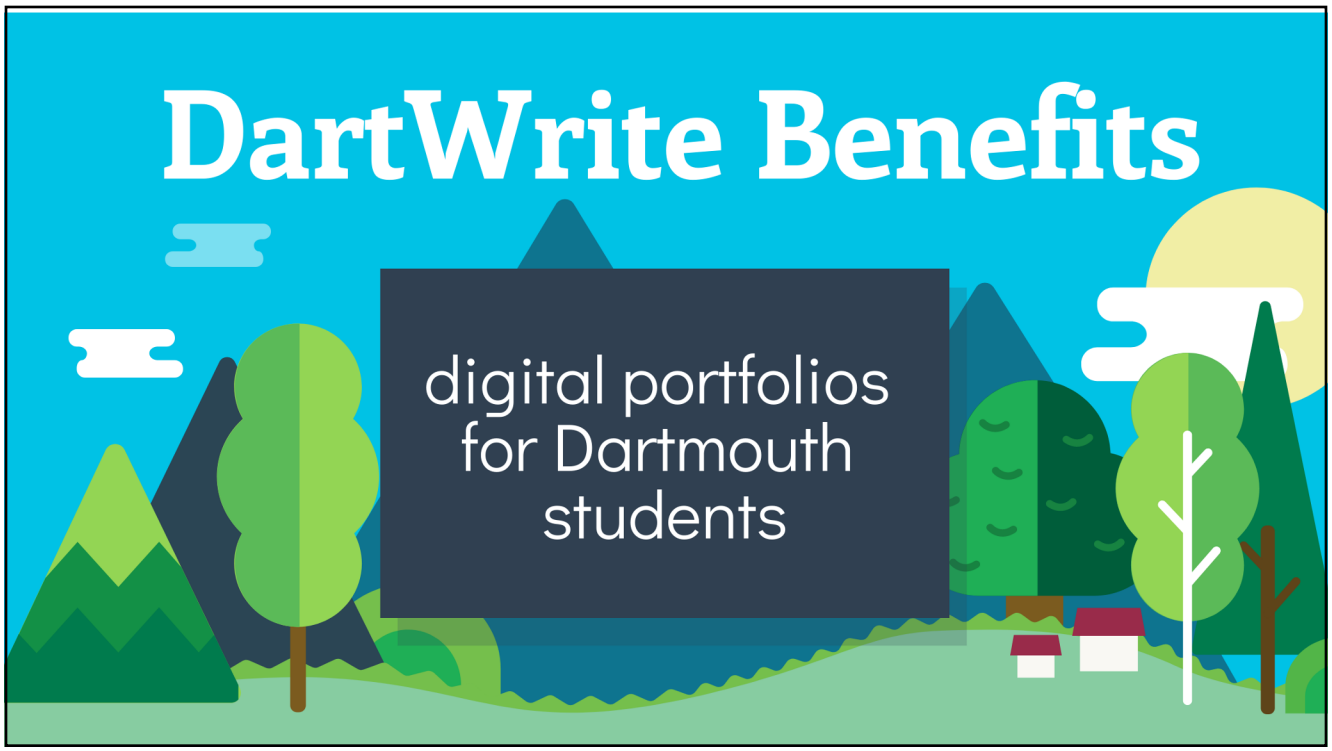
25.58

DartWrite

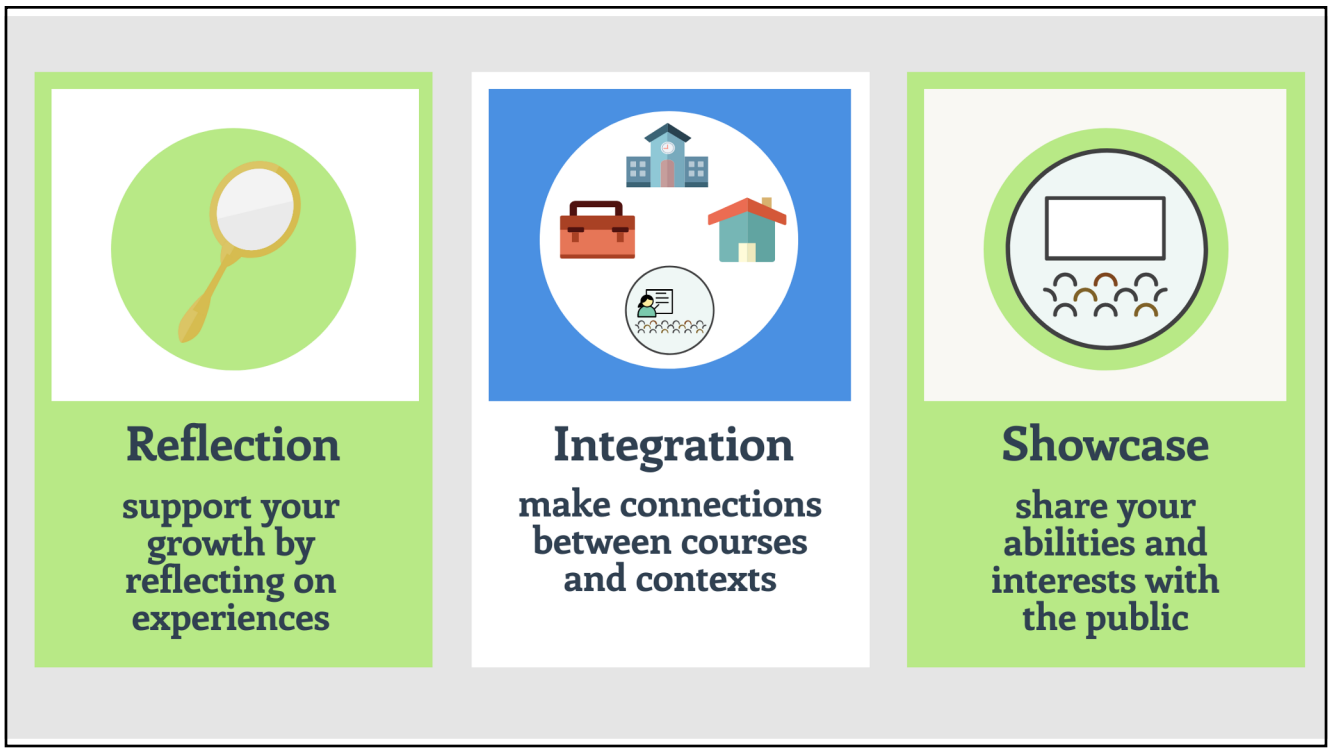
Digital WordPress Portfolio
Self-Guided Tutorial

reflect | showcase | succeed

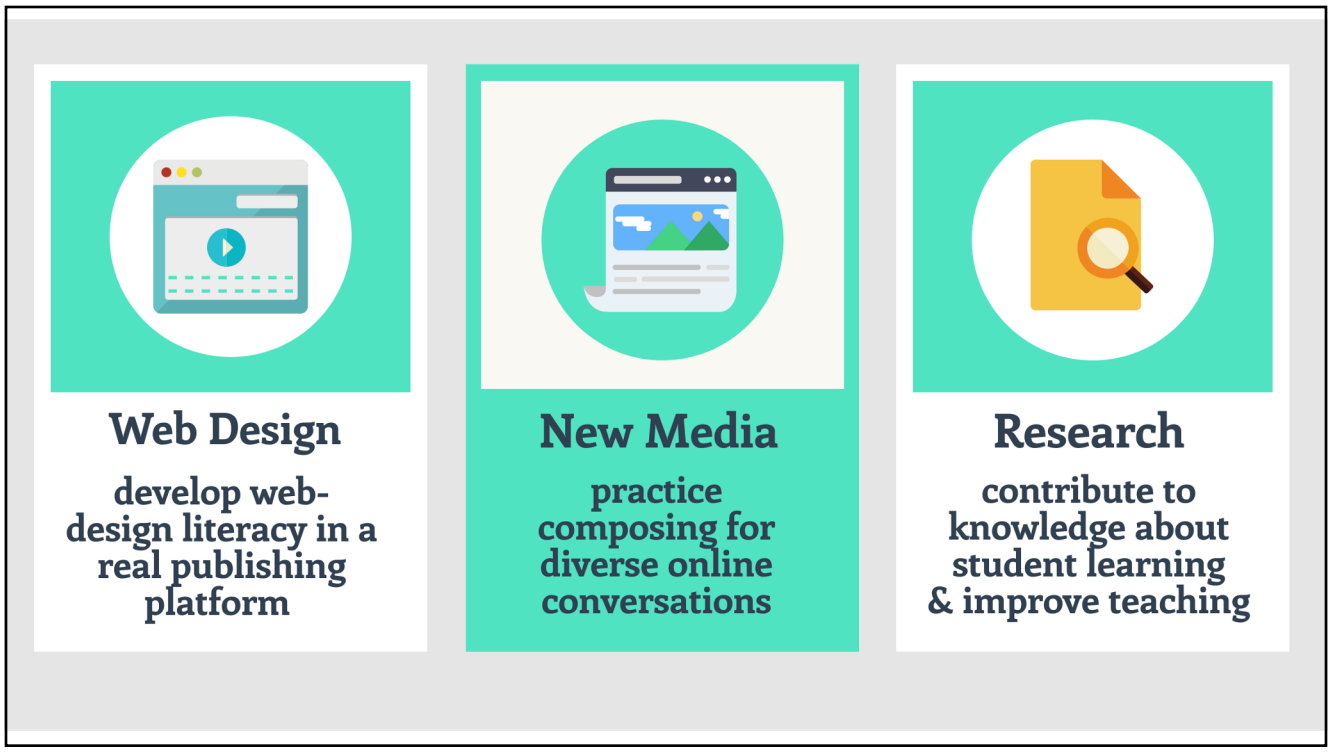
1



2


Reflection
support your growth by reflecting on experiences
Integration
make connections between courses and contexts
Showcase
share your abilities and interests with the public

3


Web Design
develop web-design literacy in a real publishing platform
New Media
practice composing for diverse online conversations
Research
contribute to knowledge about student learning & improve teaching

4

Hands-on

- First:

- Create a “Courses” page, then an ENGS 31 page.
- Upload a Ppt, an image, a drawing
- Place the new pages in the right spots and order, in the Menu

- Then if you want to do more:

- Create an About Me, add an image to About Me
- Play with themes and see what might be good for you
- Archive some existing pages (example, “Pre-College” if you didn’t use it).

5

Site Visibility

6

Match site visibility to audience

What audiences do you want to reach?

- As many people as possible!
- Some folks outside of Dartmouth
- Lots of Dartmouth folks
- Just a few Dartmouth folks
- Just you and your professor
- Just you

Site-wide visibility options (dashboard > settings > reading)

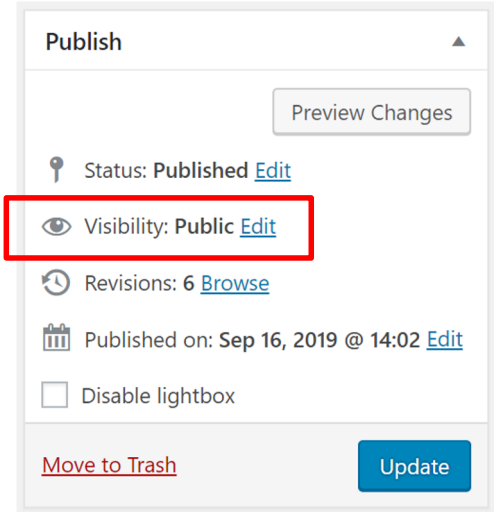
- Encourage indexing (google-able)
- Discourage indexing (anyone with link)
- Visitors must log in (anyone with Dart ID)
- Only registered users (Dart IDs you pick)
- Only admins (IDs you name admins)
- Password required (anyone with pw)

7

What if it's more complicated?

Visibility settings for each page/post

1. Enter "Edit" mode on a page or post
2. Look or hover your mouse in the upper right
3. Find the visibility options:
   a. Public - whatever the site settings are
   b. Private - only admins can see
   c. Password Protected - enter a **page/post-specific** pw



8

Additional Help

9

Portfolio Resources

- In-Depth CampusPress User Guides: https://help.edublogs.org/user-guide/
- How-To videos: in your site's Dashboard menu, at the top left, click on Video Tutorials
- Contact DartWrite Portfolio Project Fellow: Felicia Ragucci '22 (felicia.a.ragucci.22@dartmouth.edu)
- Schedule a one-on-one with an RWIT Tutor: https://students.dartmouth.edu/rwit/
- Write to DartWrite@Dartmouth.Edu with *any* questions and to WordPress@Dartmouth.Edu for technical support.

10