



Midterm Exam - Practical

Instructions

Midterm Exam - Practical Question

Instructions

You are permitted (and encouraged) to reference all CST8227 course material hosted on Brightspace as well as your BYOD (for example, your solution to a previous lab).

I simply ask that you complete the practical question on your own --- no help from your lab prof, no help from your friends nor the Internet other than the links provided in course materials.

Note: you are permitted to use the Internet to research on how to debounce a push-button switch. See below.

The Circuit and Serial Monitor

Use your Teensy to build a circuit that plays the Match Game. See below for the rules of the game.

Upon start-up, the initial state of the 7-segment display is continuously looping randomly picked digits (0 – 9) with a 500 mSec delay between each digit. As well, the Serial Monitor displays the current round number as:

Round #1...

An additional requirement is to display the following to the Serial Monitor each time the PBS is pressed: the PBS press #, and the frozen digit being displayed on the 7-segment display.

Follow this template for the Serial Monitor output:

```
PBS press # [1 | 2] ==> [frozen digit on the 7-segment display]
```

After the frozen digit is released, go back to continuously looping randomly picked digits (0 - 9) with a 500 mSec delay between each digit.

Upon the 2nd press of the PBS within a round, display whether or not the digit of the first PBS press equals the digit of the second PBS press.

If the digits match, display to the Serial Monitor:

```
=> **MATCH**
```

As well, blink the match-digit on the 7-segment display for 2-seconds to show a win.

If the digits don't match, display to the Serial Monitor:

```
=> NO Match... period.
```

As well, display the decimal point on the 7-segment display for 2-seconds to indicate a loss.

Build of Materials (BoM)

Build your circuit using the following hardware components:

- 1 x Teensy v3.2 microcontroller
- 1 x USB cable (your own please)
- 1 x breadboard
- 1 x Shift Register (Texas Instruments)
- 1 x 7-Segment Display
- 1 x Push-Button Switch (PBS)
- 9 x 220-Ohm Resistors
 - 8 x 220-Ohm resistors for the 7-segment display; one resistor per 8 segments
 - 1 x 220-Ohm resistor for the PBS; configure in pull-down
- Connection Wires

Software (SW) Requirements

Write, debug and test a Teensyduino sketch that plays the Match Game.
The rules of the game:

- The game is played in rounds, and a round is defined as 2 presses of the push-button switch (PBS).
 - Display the round number to the Serial Monitor: Round #*N*
- At the start of a round, the 7-segment display is continuously looping randomly picked digits, 0 thru 9 (inclusive). Set the delay between each digit to 500 mSec.
- When you press the PBS, freeze the digit on the 7-segment display for 2-seconds (2000 mSec).
 - Display *PBS press #1* to the Serial Monitor
- When the digit un-freezes, go back to continuously looping randomly picked digits (0 - 9). Set the delay to 500 mSec.
- When you press the PBS for the 2nd time in the round, freeze the digit on the 7-segment display for 2-seconds (2000 mSec).
 - Display *PBS press #2* to the Serial Monitor
- Are the two digits --- from PBS press #1 and #2 --- the same?
That's a match!
 - Display => *****MATCH***** to the Serial Monitor
 - Blink the matched-digit for 2-seconds on the 7-segment display
- Else, the two digits are not equal. That's not a match.
 - Display => *NO Match... period.* to the Serial Monitor
 - Display the (right) decimal point for 2-seconds
- Increment the round number, and display Round #2... to the Serial Monitor.
- Repeat until game ends: when Teensy is powered off or player closes IDE & walks away.

Documentation Requirements

Begin your sketch file with this header comment, and tell me the following information:

```
/**  
** Name: enter your full name
```

```
** Email: enter your Algonquin College email
** Expected Grade: enter the grade you expect to see
once I release the marks
** Known Issues: document any known issues that you
encountered, if any
** /
```

Datasheets

I made use of the following datasheets when making my solution:

- 7-segment Display: LDS-A504RI_7_Segment_Display.PDF
- Shift Register (Texas Instruments): sn7hc595.PDF

Download the Data Sheets from Brightspace: Content => Datasheets

References

1. Arduino Language Reference for random function:
<https://www.arduino.cc/reference/en/language/functions/random-numbers/random/>

Deliverables

Create a zip-file (must be zip please) that contains the following items:

1. Your sketch (.INO) file, with header comment.
2. A video(s) recording showing the running behaviour of your circuit.
 1. Show me your circuit: do an 'aerial' shot, starting at the top of the circuit and move to the bottom. I expect to see the 7-segment display turned off, including the decimal point.
2. Show me one (1) round: Match
 1. Show me the 7-segment Display: I need to see all digits (0 - 9)
 2. Show me the Serial.Monitor
3. Show me one (1) round: No Match
 1. I don't need to see the digits (0 - 9)
 2. Show me the Serial.Monitor

3. A "selfie" with your prized circuit. Or, show me your Algonquin College Student ID card in the video recording of the previous step.

4. Screenshot of: Teensyduino IDE => Tools => Get Board Info dialog

How to: Get Board Info Dialog

With you Teensy connected to your BYOD, launch the Teensyduino IDE.

Set the COM port: Tools => Port => select the COM port listed under the Serial ports list

Open the board info dialog: Tools => Port => Get Board Info

Take your screenshot showing the info dialog.

Submission Instructions

Upload and submit your zip-file before the Due Date and Time.

Choose where you would like to upload your zip-file (pick one of):

- upload and submit your zip-file to Brightspace
- store your zip-file on your Algonquin College OneDrive cloud drive, and paste the URL link to Brightspace (instead of the zip-file)

Reference Screenshot: Serial Monitor Dialog

Assuming you debounced the PBS (see Rubric), your output in the Serial Monitor dialog window is to match mine exactly for full marks:



The screenshot shows the TeensyMonitor application window. The title bar reads "TeensyMonitor: /dev/cu.usbmodem58850001 Online". The main text area displays the following serial output:

```
Round #1...
PBS press #1 => 1
PBS press #2 => 1
=> **MATCH**

Round #2...
PBS press #1 => 8
PBS press #2 => 3
=> NO Match... period.

Round #3...
```

At the bottom of the window, there are several controls: a checked "Autoscroll" checkbox, an unchecked "Show timestamp" checkbox, a "Newline" dropdown menu, a "9600 baud" dropdown menu, and a "Clear output" button.

Rubic

- A+ (10) :: plays Match Game and PBS is debounced using hardware xor the Bounce2 library (i.e. software)
 - for the library, I expect to see `#include <Bounce2.h>` at the top of your sketch (.INO) file
- A (9) :: Use the PBS to be in pull-up configuration with no discrete resistor. Make the appropriate change(s) to your sketch.
- A- / B+ (8) :: Serial.Monitor output does not match reference screenshot (minor)
- B (7) :: plays Match Game and PBS is debounced using `millis()` function
- C (6) :: plays Match Game but PBS is not debounced
- D (5) :: does not play Match Game
- F (0) :: no deliverable

Penalties

- Need a hint? -2 to -4 points
- Missing header comment: -2; missing / bogus information: -1
- Missing deliverable: -2.5 points; missing Board Info dialog: -10
- Late Submission: -2 to -5 points
- Video recording does not capture requested behaviour: -2 to -3

//TODO: Independent Research

You'll be required to conduct independent research on the issue of push-button switches and debouncing.

Here's a suggestion.... you could apply SWEng principles:

- Analysis: what is debouncing? what issue are you trying to solve?
- Design: formulate a plan of action, and design a solution in hardware or software
- Implement: build your solution to the problem you are trying to solve; debouncing a PBS in this case
- Test: test, test, test; test with the idea to break your solution

Repeat until the requirement has been fulfilled.

-BP

Submissions

No submissions yet. Drag and drop to upload your assignment below.

Drop file here, or click below!

 Upload

Record ▼

Choose Existing

You can upload files up to a maximum of 2 GB. This is the maximum size allowed by D2L and can't be increased. If you need to share larger files with your learners, we recommend using OneDrive and sharing the link to the file instead.



Activity Details

Task: Submit to complete this assignment



Due November 4 at 5:00 PM



Starts Oct 31, 2022 10:00 AM Ends Nov 4, 2022 5:00 PM

Assessment



Midterm Exam -
Practical