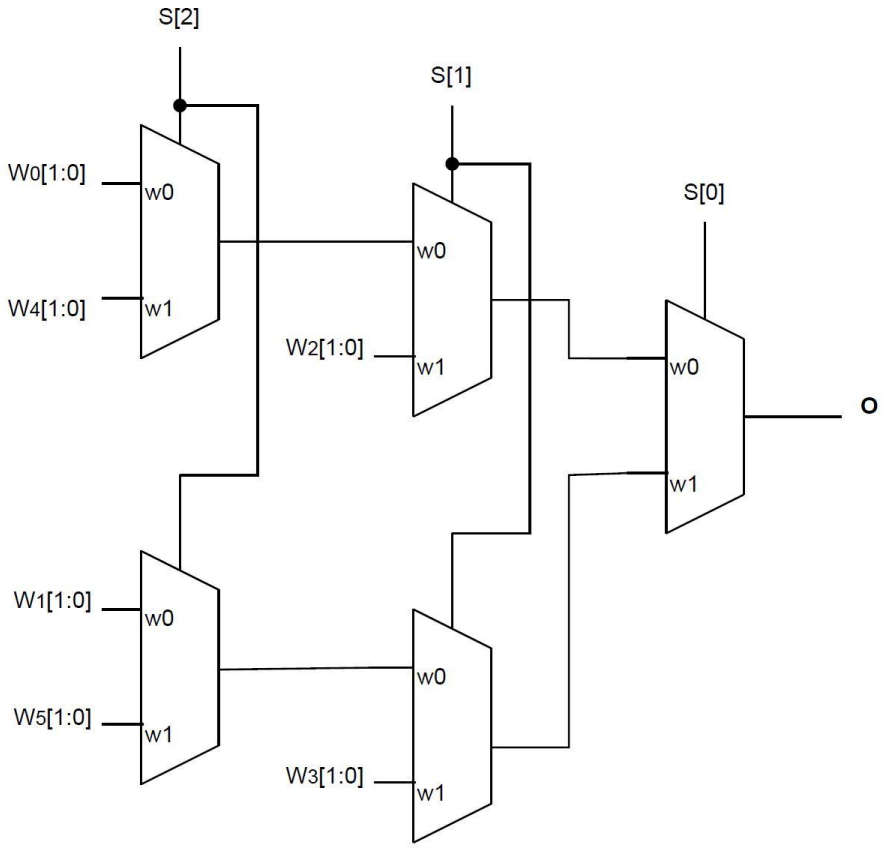
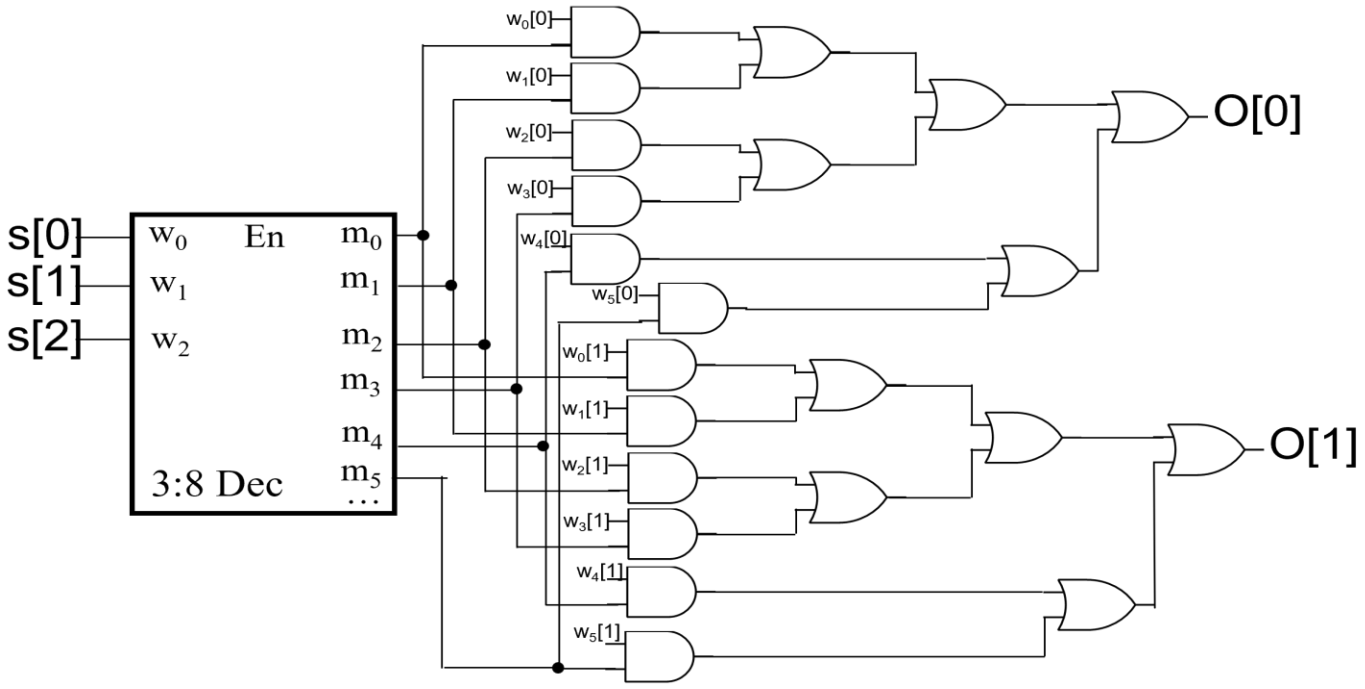
CprE 381 Homework 0

[Note: This homework first covers some of the background material that you are expected to know from CprE281 and then covers material from the first few class periods.]

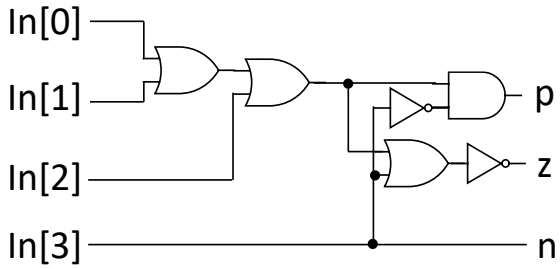
1. Review
    a. I want to select between 6 different 2-bit inputs and output it. How many bits do I need to specify which input? Draw an implementation of this using the minimum number of 2:1 Bus MUXes and no other logic.
    Since we need to uniquely name 6 values, there are ceil($\log_2(6)$) = 3 bits required.
    Solutions may vary slightly:



    b. Use one appropriately sized decoder and a small number of AND, OR, and NOT gates to implement the functionality from 1.a.
    Since select bit number is 3 (see 1.a), a 3:8 Decoder is needed. The decoder either should not have an enable or the enable should be always on (i.e. set to 1). The one-hot outputs of the decoder should be ANDed with the corresponding data inputs as shown below.



c. Design a circuit using AND, OR, and NOT gates that takes a 4-bit two's complement number as input and outputs whether the number is positive (p), negative (n), or zero (z).
Solutions may vary:



d. How many bits are required to encode the number of ps in a min?
1E12 ps/s * 60s/min = 6E13 ps/min
ceil($\log_2$(6E13)) = 46 bits

e. Convert 0x38157EE9 into binary and decimal.
0x38157EE9 = 0011 1000 0001 0101 0111 1110 1110 1001_2
= $3*16^7 + 8*16^6 + 1*16^5 + 5*16^4 + 7*16^3 + 14*16^2 + 14*16^1 + 9*16^0$
= 805306368 +134217728+1048576+327680+28672+3584+224+9
= 940932841

2. Introduction to computers
   a. Using your favorite computer as an example (cannot be the Amazon Dash button or the Blue Waters supercomputer), map at least 8 different subcomponents (e.g., chips and antennas) into one of the five main components of a computer presented in the textbook or pre-recorded lecture videos. Use online

descriptions, teardowns, and component data sheets to help you with this task, but cite your sources.
5 components: Input, Output, Memory, Control (in CPU), and Datapath (in CPU).
Solutions:
Students may also choose other systems (e.g., SIM cards, gaming desktop, and cell phones)

Example of Amazon Dash button:
Input:
Atmel ATWINC1500B wireless chip (also used for output)
Cypress CYBL10563-68FNXI Bluetooth Low Energy chip (also used for output)
InvenSense INMP441 microphone
Push-button
Output:
BCM43362 Wi-Fi module, RGB LED
Memory:
Micron M25P32 32Mbit/M25P16 16Mbit flash memory
Atmel microcontroller memory 128 Kbytes SRAM
STM32F205/215 - Arm Microcontrollers (MCUs) for Medical, Industrial and Consumer applications - STMicroelectronics

Processor:
Datapath:
- Cortex M4 processor inside Atmel microcontroller

Control:
- Atmel microcontroller MU internal system controller

Petroff, Matthew. New Amazon Dash Button Teardown (JK29LP). Matthew Petroff, 12 July 2016, https://mpetroff.net/2016/07/new-amazon-dash-button-teardown-jk29lp/

b. Look up one stored program computer from each of the last 9 decades (1940's-2020's) and report its primary/main memory capacity (in B and then the most appropriate term such as Gibibyte). Do NOT list the same computers as your friends, although you may work together to find multiple computers per decade. Also do NOT use a single source for the entire list (this is cheating and not helpful for your learning)—you must CITE the source for each computer. For extra enrichment (i.e., not graded): See if you can find the physical size and cost of the systems in today's (i.e., 2023) dollar.
Solutions widely variable but should generally show near exponential increase in memory capacity if a similar "class" of system is chosen (e.g., mainframe-class).

c. Implement a program in C to calculate the average of an array of N integers (you may assume int-sized values whose initialization values are up to your testing

approach). (1) Report the time it takes you to (correctly) write and test this program. (2) Report the program size in number of characters. Compile the program to an executable object file. (3) Report the executable size and the architecture for which you compiled it. Finally, run the program and (4) report the time it takes to execute for 3 different test arrays that differ in size by orders of magnitude. Include the program files (C language and executable object code) in a tarball (HW00_2c.tgz).

Solutions widely variable.

2c.

```c
#include <stdio.h>

int main() {
  int arr[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
     int sum = 0;
     int i = 0;
     float average = 0;

    for(i = 0; i < 10; i++) {
       sum = sum + arr[i];
    }

    average = (float) sum / i;
    printf("Average: %.2f", average);

    return 0;
  }
```

```
16008 Jan 26 22:57 a.out
  318 Jan 26 22:57 avg.c
 1792 Jan 27 00:18 avg.o
 1508 Jan 26 22:58 avg.s
```

2c2. Program size: 318 characters

2c3. Executable size: 16008 Bytes

```
a.out: ELF 64-bit LSB pie executable, x86-64,
```

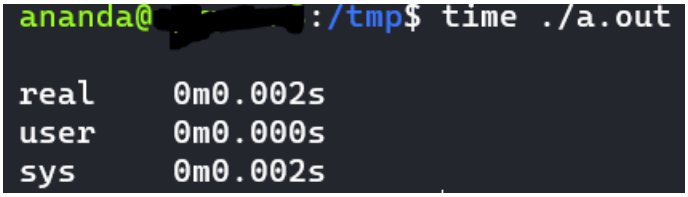
Architecture: 64-bit x86.

2c4. N=100, time= 1ms

N=1000, time= 1ms

N=10000, time= 2ms

Example for N = 10000



3. MS Teams
    a. Login to MS Teams and post an interesting, non-technical fact about yourself or your interests in the random channel. Use this channel for any vaguely relevant discussions (e.g., new chips or computing paradigms) or just course comradery posts.
    See MS Teams discussion for correct solutions.