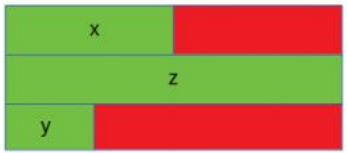
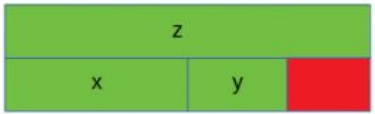
The size of a C structure

```c
#include <stdio.h>
struct myStruct {
    char a;
    int b;
    char c;
};

int main() {
    printf("%lu\n", sizeof(struct myStruct));
    return 0;
}
```





The size of a structure in C depends on several factors:

1. **Data Types and Sizes of Members**: Each member in the structure contributes to its overall size, and their sizes depend on the data types (e.g., `int`, `char`, `float`, etc.).
2. **Padding**: To align data in memory for faster access, many compilers add padding bytes between members or at the end of the structure. This padding aligns each member according to its alignment requirements (e.g., `int` may be 4-byte aligned).
3. **Order of Members**: The order of the members can affect padding. Struct members are laid out in memory in the order they are declared, and different orders may lead to different amounts of padding.
4. **Compiler and Architecture**: Different compilers and architectures may have different alignment requirements, which affect the structure's size.

slt instruction

In MIPS assembly, which of the following code snippets correctly checks if the value in register $t1 is not greater than that in register $t2 and branches to the label Equal_Or_Less if true?

A.
```
slt   $t0, $t2, $t1
beq   $t0, $zero, Equal_Or_Less
```

B.
```
slt   $t0, $t1, $t2
bne   $t0, $zero, Equal_Or_Less
```

C.
```
slt   $t0, $t1, $t2
beq   $t0, $zero, Equal_Or_Less
```

D.
```
slt   $t0, $t1, $t2
beq   $t0, $zero, Equal_Or_Less
j     end
```

2s-complement notation

The following is an 8-bit binary representing an integer in 2s-complement notation.

$1111\ 1010_2$

Please convert it into a decimal number.

IEEE floating point number

The following 32-bit binary representation is a floating-point number in IEEE std 754.

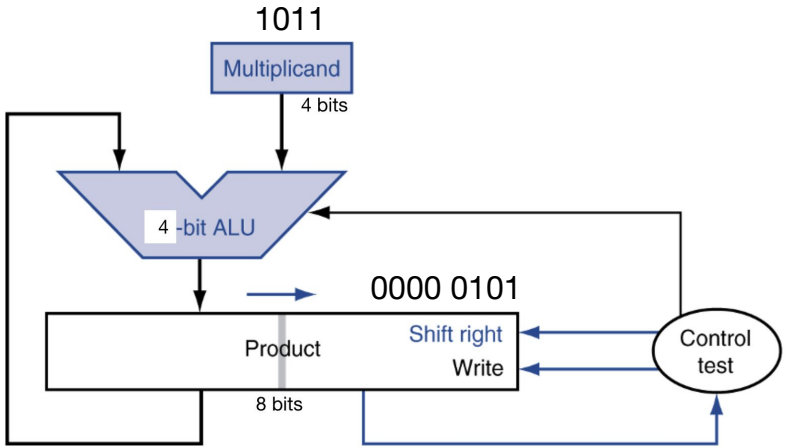
$$1000\ 0000\ 0100\ 0000\ 0000\ 0000\ 0000\ 0000_2$$

Please convert it into a decimal number. (The result can be an expression with power items. You don’t have to convert it into a real number.)

Translate MIPS assembly to C

```
main:
  lw $t0, n
  la $t1, arr           #load base address of arr to $t1
  li $t2, 0             #load 0 to $t2
  li $t3, 0             #load 0 to $t3
  li $t4, 0             #load 0 to $t4
Loop:
  lw    $t5, 0($t1)     #load the value store in the address in $t1 to $t5
  mul   $t3, $t5, $t5   #multiply values in $t5 and $t5, save the result in $t3
  add   $t2, $t2, $t3   #add values in $t2 and $t3, save the result in $t2
  addi  $t1, $t1, 4     #increment the value in $t1 by 4
  addi  $t4, $t4, 1     #increment the value in $t4 by 1
  slt   $t6, $t4, $t0   #if the value in $t4 is less than $zero, set $t6 to 1; otherwise set $t6 to 0
  bne   $t6, $zero, Loop #if the value in $t6 is zero, go to Loop
  #system call to print the value in $t2
  li    $v0, 1
  add   $a0, $t2, $zero
  syscall
  #exit the program
  li $v0, 10
  syscall
```

Multiplication



Iteration	Step	Multiplicand	Product
0	Initial values	1011	0000 0101
1	1: ProdH += Macand	1011	1011 0101
	Shift product right	1011	0101 1010
2	0: No operation	1011	0101 1010
	Shift product right	1011	0010 1101
3	1: ProdH += Macand	1011	1101 1101
	Shift product right	1011	0110 1110
4	0: No operation	1011	0110 1110
	Shift product right	1011	0011 0111

Computer Architecture

Introduction to Circuitlab

Exercise 3 in Week 8

Complete the design of the seven-segment decoder by designing boolean equations for the segments S_c and S_d:

- assuming that inputs greater than 9 must product blank (0) outputs
- assuming that inputs greater than 9 are don't cares.

Then, sketch a reasonably simple gate-level implementation in the case b and simulate the resulting circuits on CircuitLab (https://www.circuitlab.com/).

K-map for Sc

$D_{3:0}$	Sa	Sb	Sc	Sd	Se	Sf	Sg
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	1	0	1	1
others	X	X	X	X	X	X	X

Here, the value **X** means that the value is undetermined : it can be 0 or 1.

Sc

$D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	1	x	1
01	1	1	x	1
11	1	1	x	x
10	0	1	x	x

K-map minimization

Sc

$D_{3:2}$

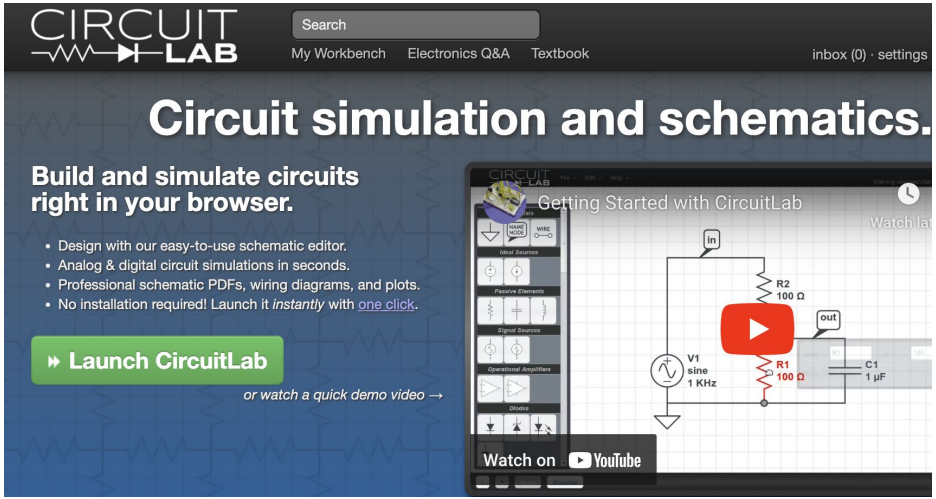
$D_{1:0}$

	00	01	11	10
00	1	1	x	1
01	1	1	x	1
11	1	1	x	x
10	0	1	x	x

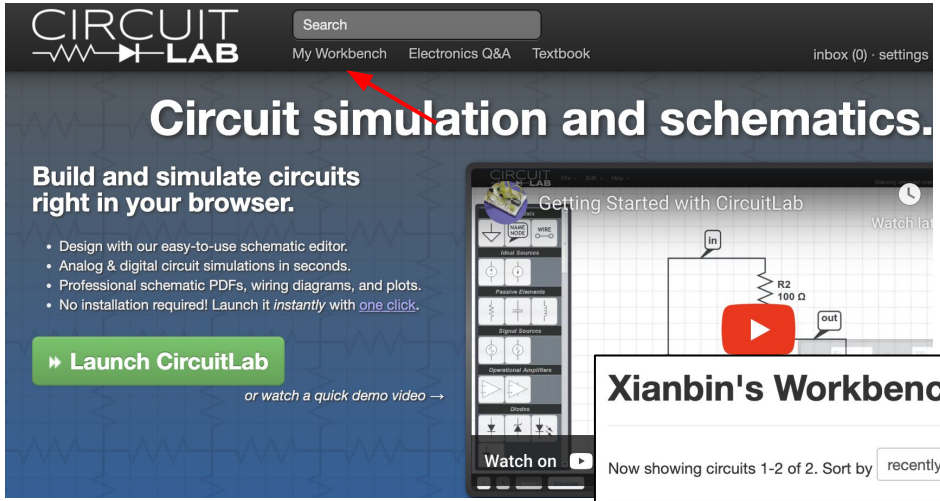
$$Sc = \overline{D}_1 + D_3 + D_1D_0 + \overline{D}_3D_2$$

Circuitlab

- CircuitLab(https://www.circuitlab.com/): an online platform for circuit simulation and schematic.
- You can log in using your NYU account.

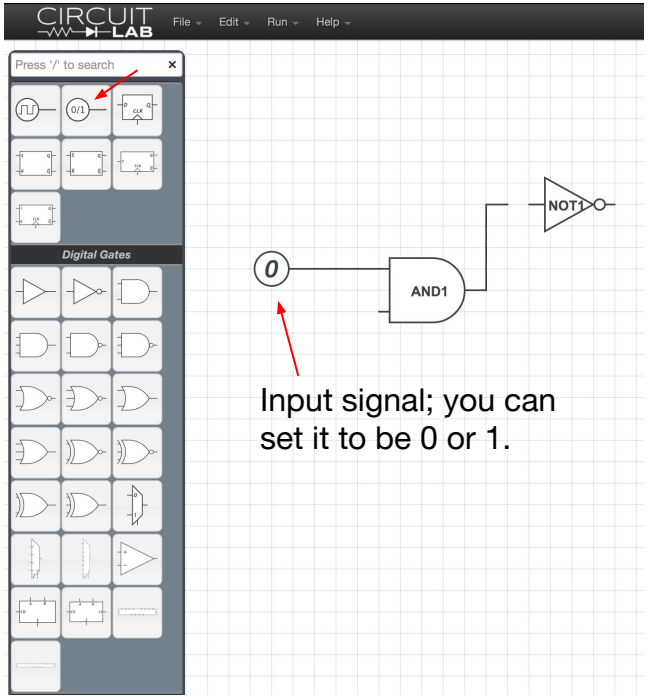


Start a schematic



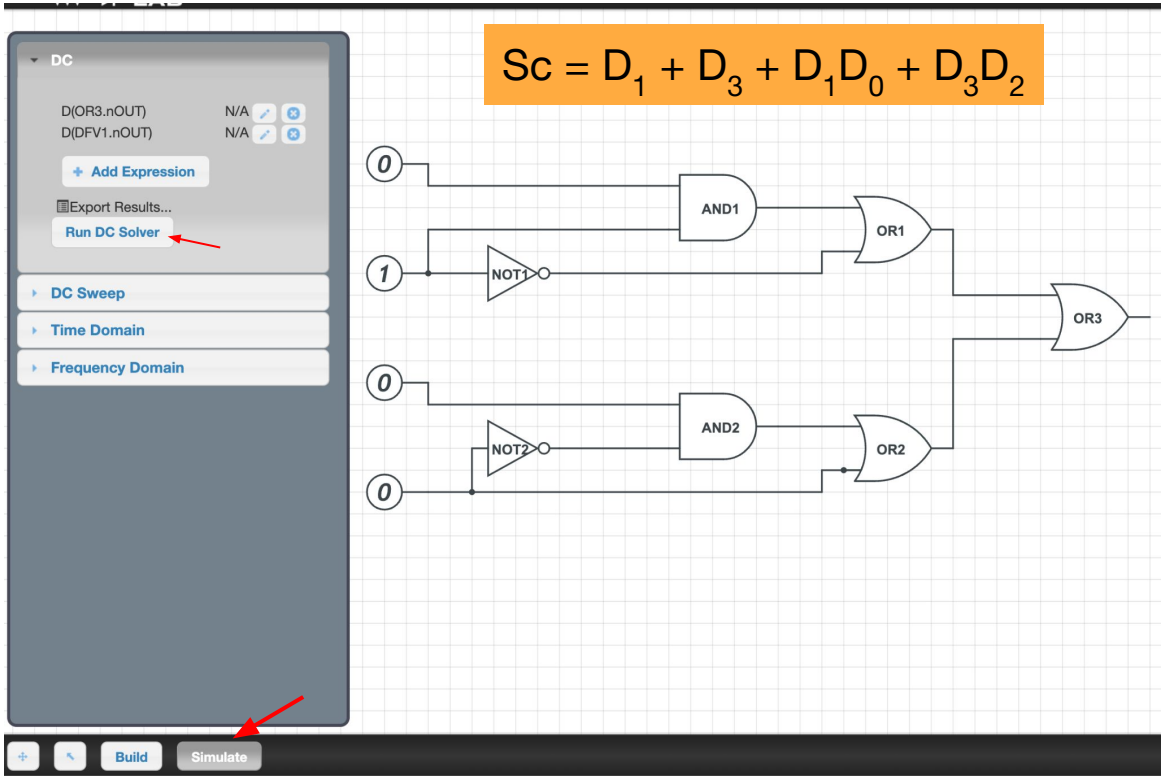
Click the My Workbench and then, the New circuit to start the schematic of your circuit.

Plotting



- You can drag symbols from the left panel, put them onto the canvas, and connect them by lines.

Schematic of Sc



Click the Simulate button, and then click Run DC Solver, you can see the outputs

Now, it is your turn.

Please draw a schematic for Sd.