NCTU-EE IC LAB – Fall 2023

Lab01 Exercise

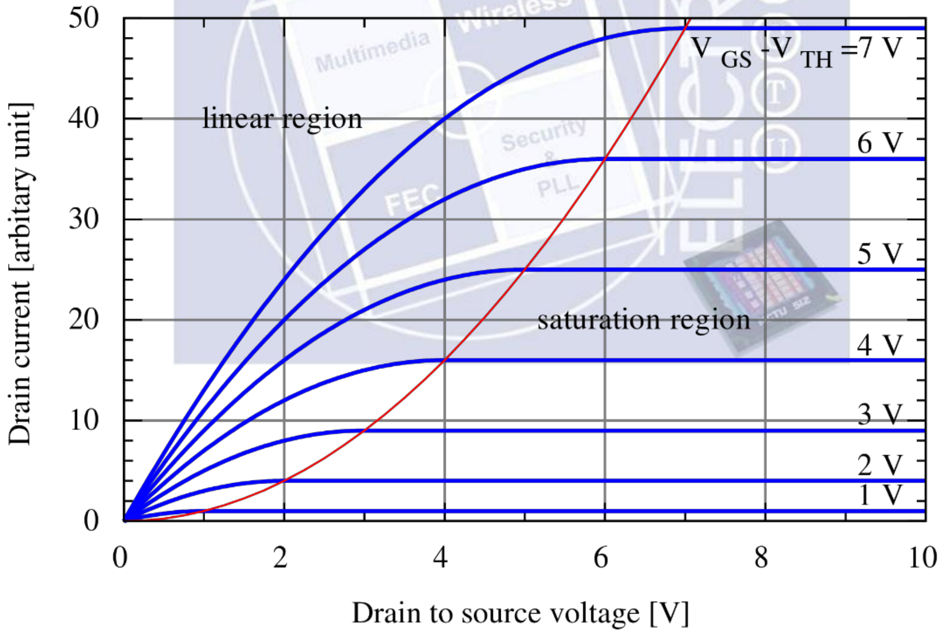
Design: Supper MOSFET Calculator(SMC)

Data Preparation

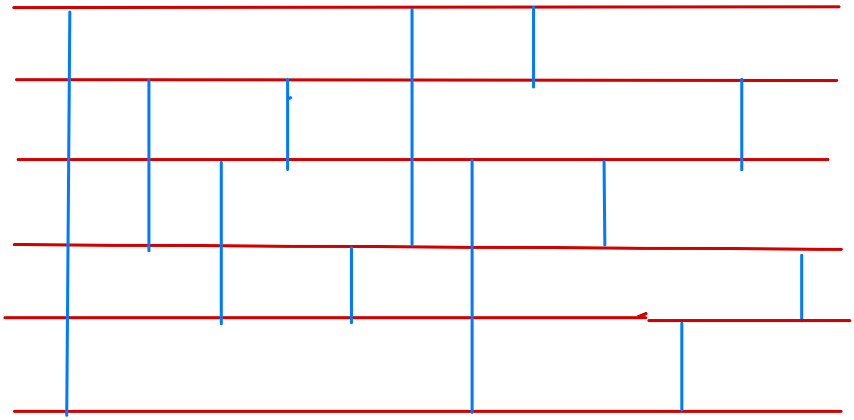
1. Extract files from TA’s directory:
   % tar -xvf ~iclabTA01/Lab01.tar

Design Description and Examples

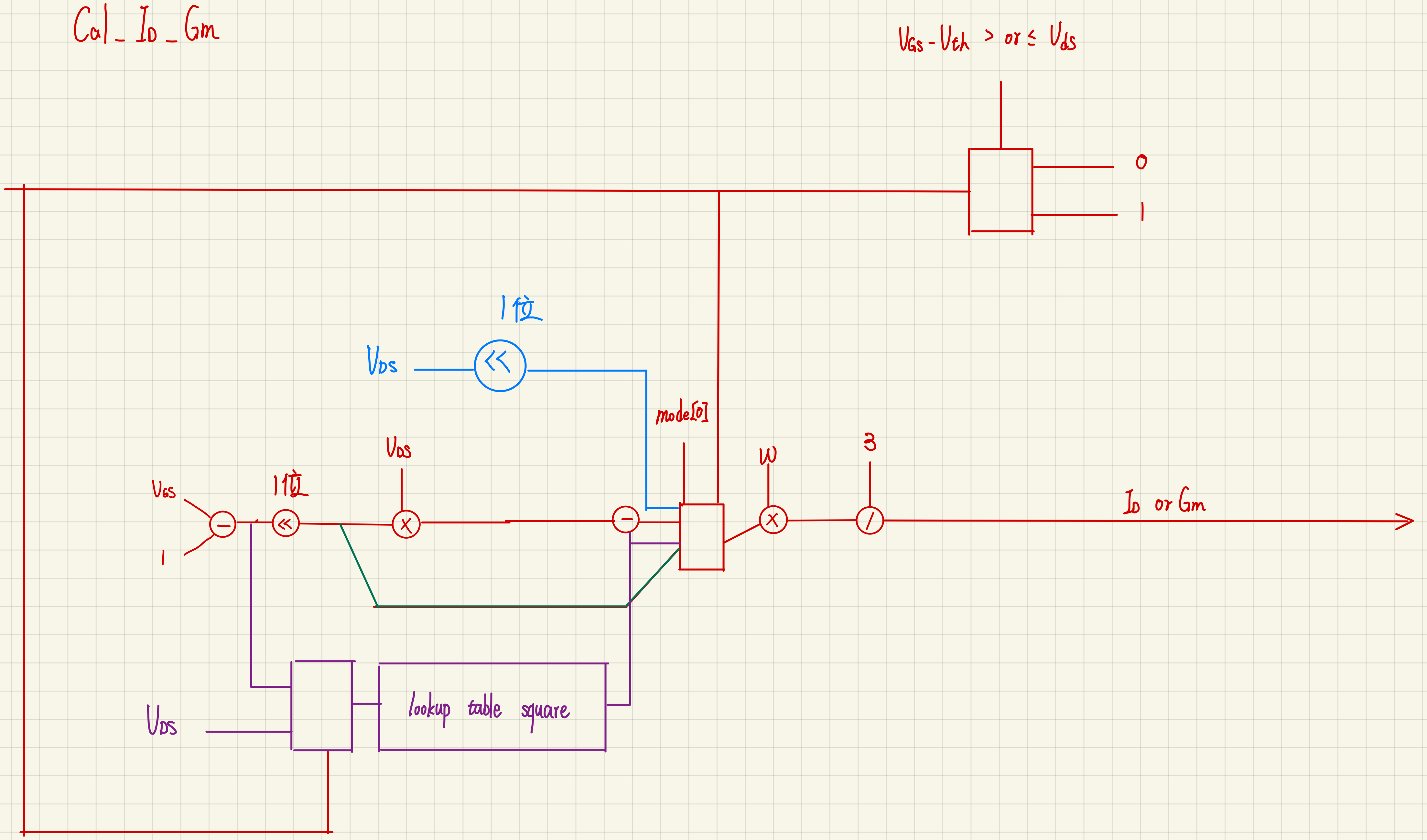
One day after microelectronics class, you are excited about what the professor talked in class, the fantastic characteristics of Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET). Therefore, you want to design a Supper MOSFET Calculator to calculate the drain current $\mathrm{I_D}$ and transconductance $\mathrm{g_m}$ in a short time. Furthermore, you also want to find what if given numerous combinations of width, V_{GS} and V_{DS}, which one could get the maximum value? To satisfy your curiosity, now you are going to conquer this problem.



Before you start your work, here is a quick review of MOS for you.



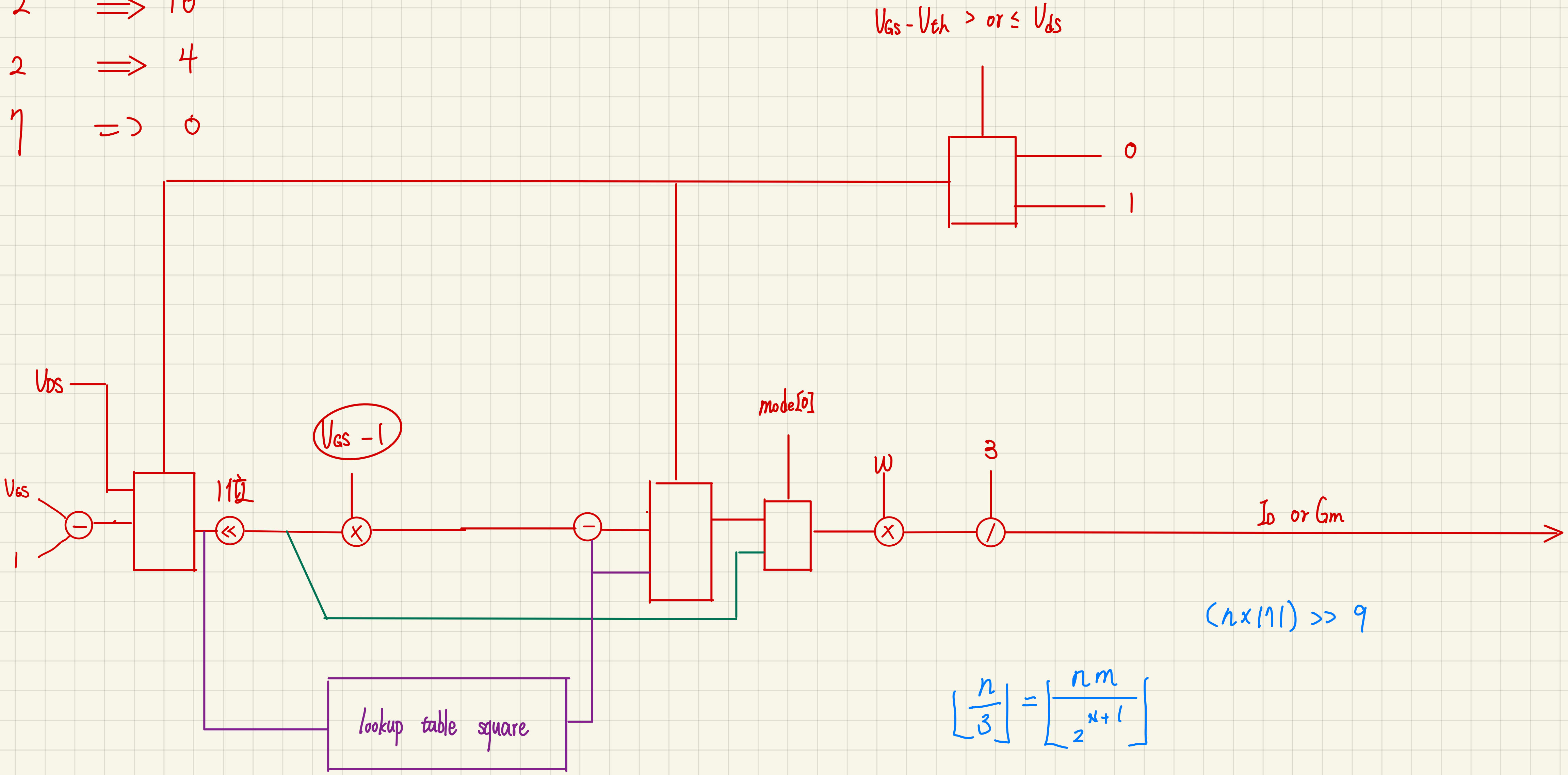
Cal_ID_Gm


VGS - Vth > or ≤ Vds
0
1
1位
VDS
mode[0]
VDS
W
3
VGS
1位
1
ID or Gm
VDS
lookup table square

3

$2 \times 5 \times 2 - 4$

6	2	5	$\Longrightarrow$	1
7	6	6	$\Longrightarrow$	25
5	2	1	$\Longrightarrow$	1
3	6	2	$\Longrightarrow$	16
1	3	2	$\Longrightarrow$	4
0	1	7	$\Longrightarrow$	0



$(n \times 171) >> 9$

$$\left\lfloor \frac{n}{3} \right\rfloor = \left\lfloor \frac{nm}{2^{N+l}} \right\rfloor$$

$$q = \left\lfloor \frac{n}{3} \right\rfloor$$

$$r = n - 3q$$

$$2^{N+l} \leq 3m \leq 2^{N+l} + 2^{l}$$

$$2^{8+l} \leq 3m \leq 2^{8+l} + 2^{l}$$

$$l = 1 \Rightarrow 2^{9} \leq 3m \leq 2^{9} + 2^{l}$$

$m = 171$, $l = 1$

Cal_out

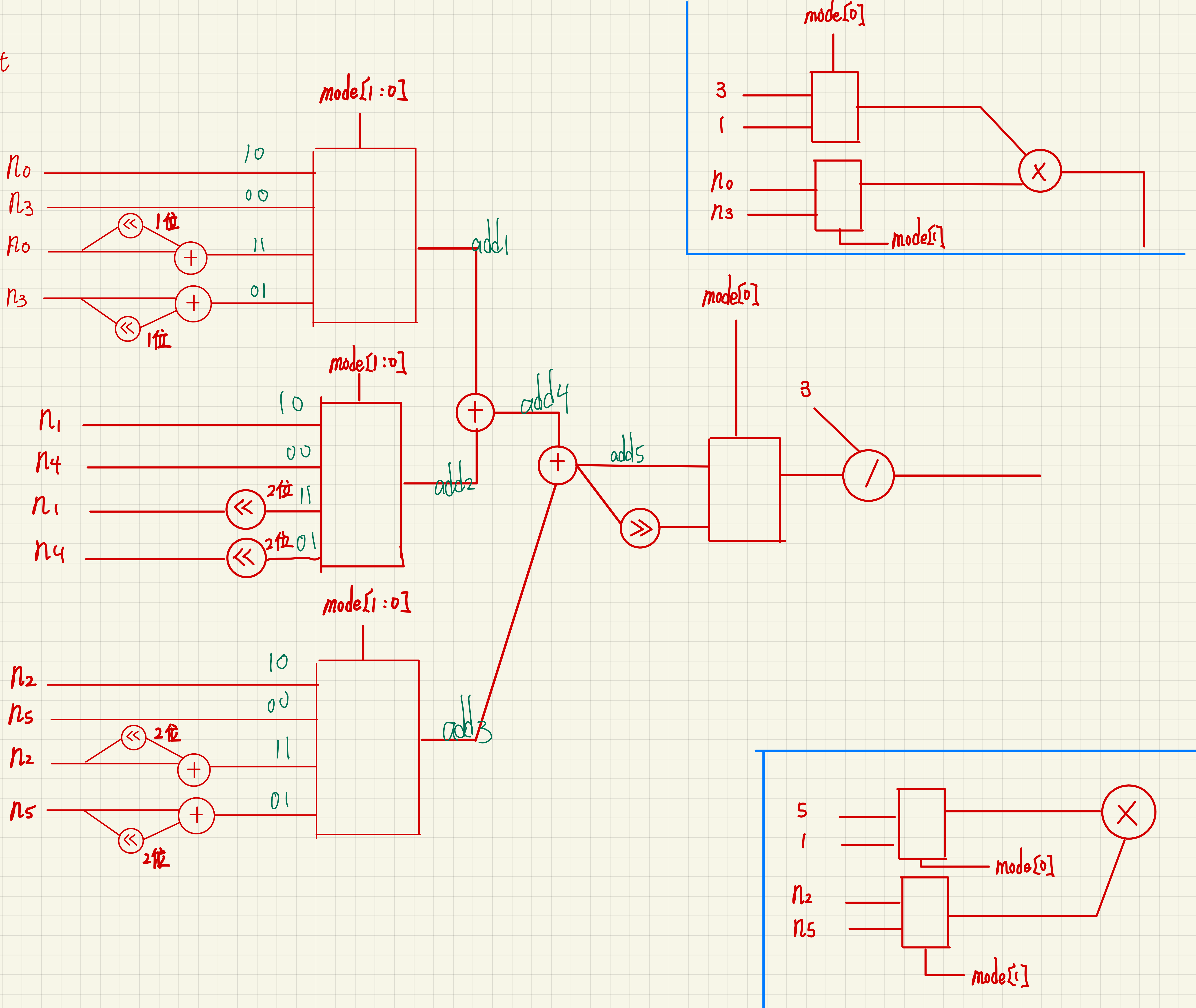
$$\begin{array}{r} 84 \\ \times\ 5 \\ \hline 420 \end{array}$$

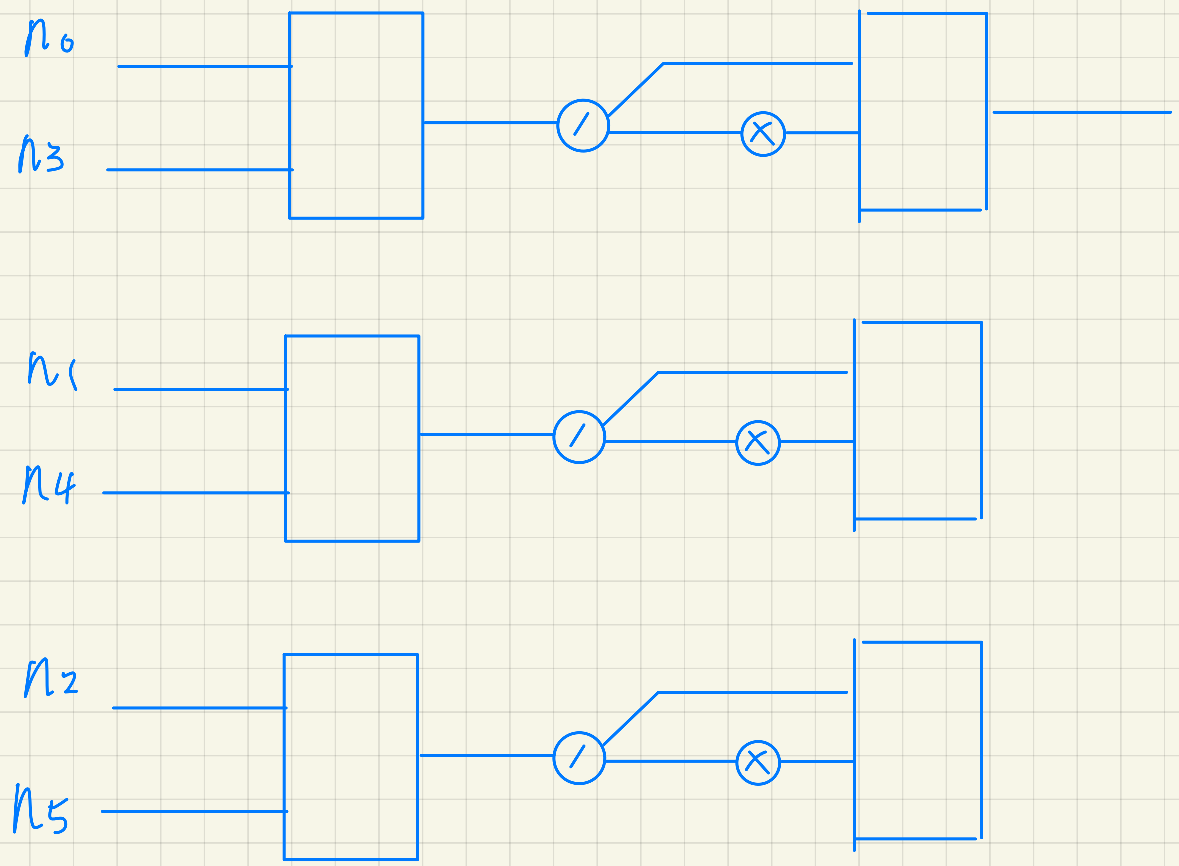
1

$$\begin{array}{r} 84 \\ \times\ 3 \\ \hline 252 \end{array}$$

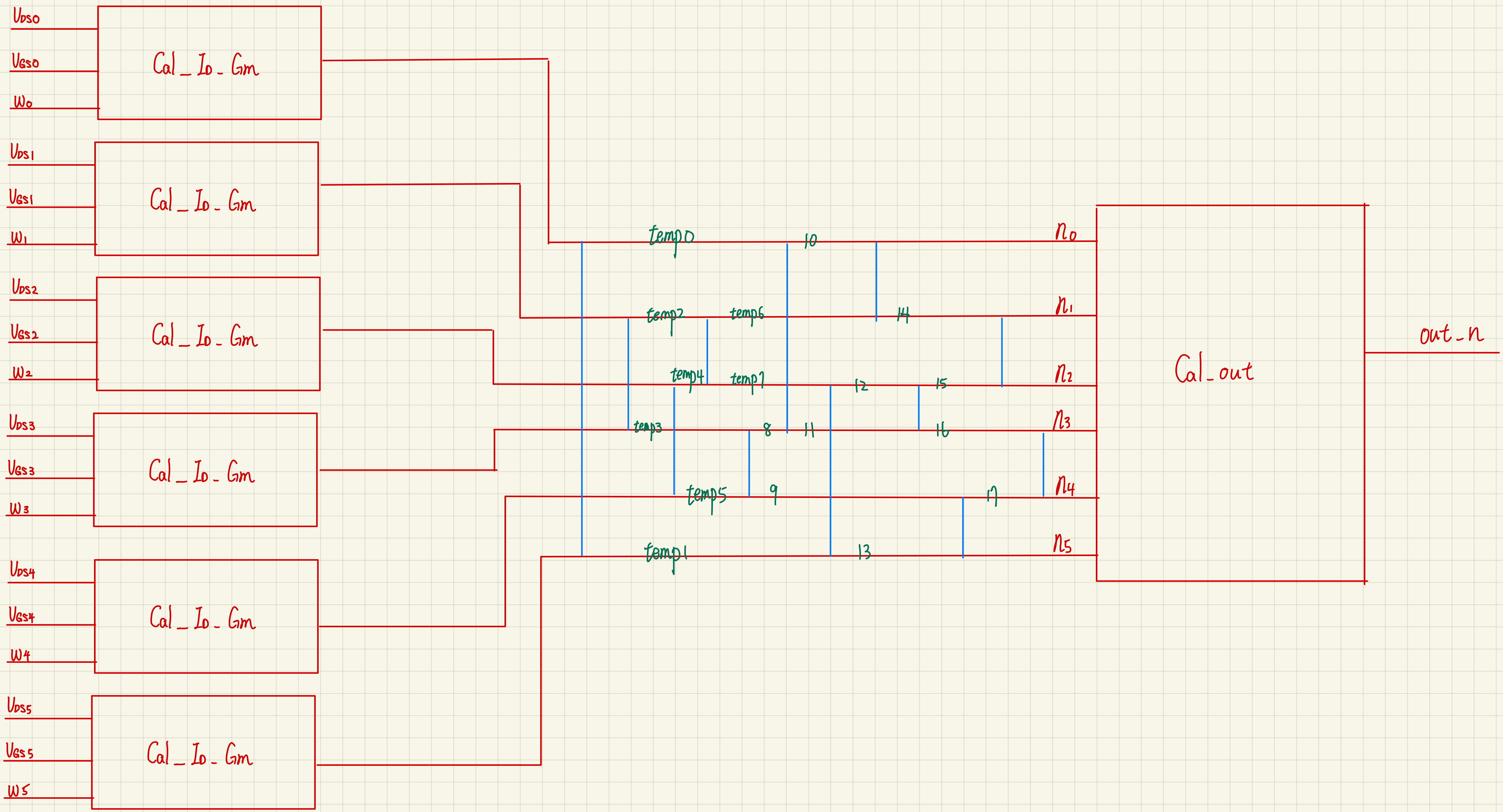
$$\begin{array}{r} 84 \\ \times\ 4 \\ \hline 336 \end{array}$$

| | | | | | | | | | |





n0
n3
n1
n4
n2
n5



VDS0
VGS0
W0
Cal_ID_Gm
VDS1
VGS1
W1
Cal_ID_Gm
VDS2
VGS2
W2
Cal_ID_Gm
VDS3
VGS3
W3
Cal_ID_Gm
VDS4
VGS4
W4
Cal_ID_Gm
VDS5
VGS5
W5
Cal_ID_Gm
temp0
10
n0
temp2
temp6
14
n1
temp4
temp7
12
15
n2
temp3
8
11
16
n3
temp5
9
17
n4
temp1
13
n5
Cal_out
out_n

a	b
0 0 0 0 0 0	0 0 0 0 0 1
0 0 0 0 0 1	0 0 0 0 1 0
0 0 0 0 1 0	0 0 0 0 1 1
0 0 0 0 1 1	0 0 0 1 0 0
0 0 0 1 0 0	0 0 0 1 0 1
0 0 0 1 0 1	0 0 0 1 1 0
0 0 0 1 1 0	0 0 0 1 1 1
0 0 0 1 1 1	0 0 1 0 0 0
0 0 1 0 0 0	0 0 1 0 0 1
0 0 1 0 0 1	0 0 1 0 1 0
0 0 1 0 1 0	0 0 1 0 1 1
0 0 1 0 1 1	0 0 1 1 0 0
0 0 1 1 0 0	0 0 1 1 0 1
0 0 1 1 0 1	0 0 1 1 1 0
0 0 1 1 1 0	0 0 1 1 1 1
0 0 1 1 1 1	1 0 0 0 0 0

$b[0] = \sim a[0]$

$b[1] = a[1] \otimes a[0]$

$b[2] = a[2] \otimes (a[1] \,\&\, a[0])$

$b[3] = a[3] \otimes (a[2] \,\&\, a[1] \,\&\, a[0])$

$0 \otimes 1$

mode=3 7 7 6

✓ MOS's modes of operation

The operation of a MOSFET can be separated into four different modes, depending on the voltages at the terminals. Here we consider an NMOS model.

2X

1. **Triode region (Linear region)**

最大 252

- ✓ Condition: $\boxed{V_{GS} - V_{th} \geq 0 \,\&\&(V_{GS} - V_{th}) > V_{DS}}$
- ✓ Current: $\mathrm{I_D} = \frac{1}{2}\mu_n C_{OX}\left(\frac{W}{L}\right)[2(V_{GS} - V_{th})V_{DS} - V_{DS}^2] = K_n W[2(V_{GS} - V_{th})V_{DS} - V_{DS}^2]$
- ✓ Transconductance: $\mathrm{g_m} = \mu_n C_{OX}\left(\frac{W}{L}\right)V_{DS} = 2K_n W V_{DS}$

$\frac{1}{3} \times 7[2 \times 6 \times 6 - 36]$

2. **Saturation region**

- ✓ Condition: $\boxed{V_{GS} - V_{th} \geq 0 \,\&\&(V_{GS} - V_{th}) \leq V_{DS}}$
- ✓ Current: $\mathrm{I_D} = \frac{1}{2}\mu_n C_{OX}\left(\frac{W}{L}\right)(V_{GS} - V_{th})^2 = K_n W(V_{GS} - V_{th})^2$
- ✓ Transconductance : $\mathrm{g_m} = \mu_n C_{OX}\left(\frac{W}{L}\right)(V_{GS} - V_{th}) = 2K_n W(V_{GS} - V_{th})$

$\begin{array}{r} 36 \\ 7 \\ \hline 252 \end{array}$

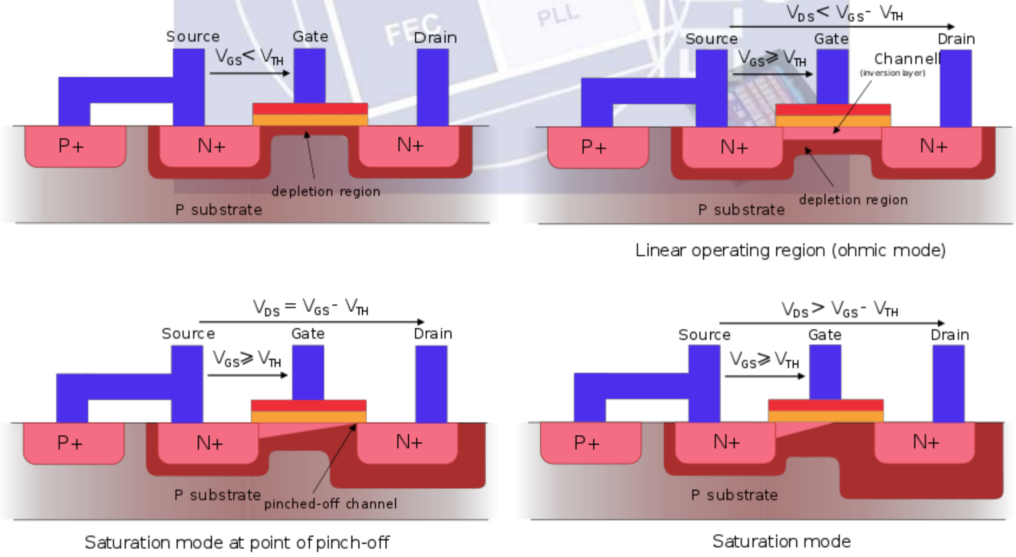
3. **Subthreshold region (cut-off)**

- ✓ Condition: $V_{GS} - V_{th} < 0$
- ✓ Current: $\mathrm{I_D} \propto e^{V_{GS}}$ (behavior like BJT, very small current)
- ✓ Will not happen in this lab since V_{GS} is range from 1~7 which won't smaller than $V_{th} = 1$

4. **Deep triode region**

$3 \times 81 + 4 \times 63 + 5 \times 48$

- ✓ Condition: $V_{GS} - V_{th} \geq 0 \,\&\&\, 2(V_{GS} - V_{th}) \gg V_{DS}$
- ✓ Current: $\mu_n C_{OX}\left(\frac{W}{L}\right)[(V_{GS} - V_{th})V_{DS}]$
- ✓ Will not happen in this lab (since V_{DS} is range from 1~7 which is not negligible level.)



Linear operating region (ohmic mode)

Saturation mode at point of pinch-off

Saturation mode

➢ To simplify calculation, we assume $\boxed{K_n = \frac{1}{2}\mu_n C_{OX}\left(\frac{1}{L}\right) = \frac{1}{3}, V_{th} = 1}$ in this lab

➢ Also, we **don't** consider **body effect** and **channel length modulation** in this lab.

➢ No need to consider fixed-point division, round-down to **integer only**.

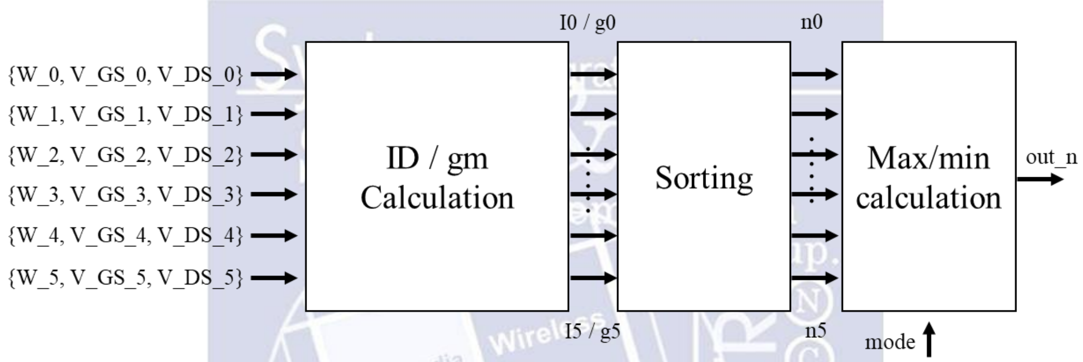
✓ Description of this lab

➢ Input:

You will receive a sequence with 6 combinations of **width,** $\mathbf{V_{GS}}$ $\boldsymbol{and}$ $\boldsymbol{V_{DS}}$, each input value would range from 1~7 (3 bit). Each name is as following.

- W_0, V_GS_0, V_DS_0
- W_1, V_GS_1, V_DS_1
- …
- W_5, V_GS_5, V_DS_5

Also, you will receive 2-bits mode code. In this lab, one bit **mode[0]** indicates that you would calculate whether the drain current or the transconductance. And, another one bit **mode[1]** shows that your output should be largest or smallest value.



➢ Calculate

You need to determine which mode would the MOS operate based on the given V_{GS} and V_{DS}. Next, you should calculate the drain current or the transconductance with corresponding equation. Below is the equation you would use in this lab.

- *If (V_GS_n – 1 > V_DS_n)   // Triode mode*
  - $I_{D_n} = K_n W[2(V_{GS} - V_{th})V_{DS} - V_{DS}^2] = \frac{1}{3}\{W[2(\text{V\_GS\_n} - 1)\text{V\_DS\_n} - \text{V\_DS\_n}^2]\}$
  - $g_{m_n} = 2K_n W V_{DS} = \frac{2}{3}[W\text{V\_DS\_n}]$ max : 32
- *If (V_GS_n – 1 <= V_DS_n)   // Saturation mode*
  - $I_{D_n} = K_n W(V_{GS} - V_{th})^2 = \frac{1}{3}[W(\text{V\_GS\_n} - 1)^2]$ max : 84
  - $g_{m_n} = 2K_n W(V_{GS} - V_{th}) = \frac{2}{3}[W(\text{V\_GS\_n} - 1)]$ max : 28

For example, $V_{GS} = 3$ and $V_{DS} = 1 => (3-1) > 1$, which means that you need to use I_D or g_m equation in triode mode. Assume W = 7, and thus, you would get

- $I_{D_n} = \frac{1}{3}\{W[2(\text{V\_GS\_n} - 1)\text{V\_DS\_n} - \text{V\_DS\_n}^2]\} = \frac{1}{3} * 7 * [2(3-1) * 1 - 1^2] = 7$
- $g_{m_n} = \frac{2}{3}[W\text{V\_DS\_n}] = \frac{2}{3}[7 * 1] = 4$
- Round-down the answer if it is not integer

11100 01110

00111

Another example, $V_{GS} = 3$ *and* $V_{DS} = 5 \Rightarrow (3-1) \leq 5$, which means that you need to use I_D or g_m equation in saturation mode. Assume W = 7, and thus, you would get

42727.608258

- $I_{D_n} = \frac{1}{3}[W(V\_GS\_n - 1)^2] = \frac{1}{3}[7 * (3-1)^2] = 9$
- $g_{m_n} = \frac{2}{3}[W(V\_GS\_n - 1)] = \frac{2}{3}[7 * (3-1)] = 9$
- Round-down the answer if it is not integer

- **Sort** 11 13 18 19 20 5

After calculation, you would have $I_{D_0} \sim I_{D_5}$ or $g_{m_0} \sim g_{m_5}$. To find the maximum and minimum total current or transconductance, now you need to preprocess your result according to **mode[0]** code. If **mode[0] == 1'b1,** you would sort the sequence $I_{D_0} \sim I_{D_5}$ as new sequence $n_0 \sim n_5$. As for **mode[0] == 1'b0,** you would sort the sequence $g_{m_0} \sim g_{m_5}$ as new sequence $n_0 \sim n_5$. Notice that n_0 is the largest one, and n_5 is the smallest one.

0 1 2 3 4 5

11111111 00000000

For instance, **mode[0] == 1'b1,** now your own $\{I_{D_0}, \ldots, I_{D_5}\} = \{25,33,27,5,6,0\}$, then after sorting you would get $\{n_0, \ldots, n_5\} = \{33,27,25,6,5,0\}$

1000 0101 0000000

Another example, **mode[0] == 1'b0,** now your own $\{g_{m_0}, \ldots, g_{m_5}\} = \{11,5,27,10,3,1\}$, then after sorting you would get $\{n_0, \ldots, n_5\} = \{27,11,10,5,3,1\}$

0111 0000

- **Calculate & Output** 1000

252 336

011 101

  - **mode[0] = 1 (Weighted Average)**
    - Larger: $\boldsymbol{I_{avg} = (3 * n_0 + 4 * n_1 + 5 * n_2)/12)}$ **(mode[1] = 1)**
    - Smaller: $\boldsymbol{I_{avg} = (3 * n_3 + 4 * n_4 + 5 * n_5)/12}$ **(mode[1] = 0)**
  - **mode[0] = 0 (Average)**
    - Larger: $\boldsymbol{gm_{avg} = (n_0 + n_1 + n_2)/3}$ **(mode[1] = 1)**
    - Smaller: $\boldsymbol{gm_{avg} = (n_3 + n_4 + n_5)/3}$ **(mode[1] = 0)**
  - Round-down the answer if it is not integer

000 001 010 0101 011 100 101 1010 11

62

30 + 32 + 15

6 - 5 0110 1010 X0000

Example 1:

$$mode = 2'b11, \{n_0, \ldots, n_5\} = \{33,27,25,6,5,0\}$$
$$\text{out\_n} = \boldsymbol{I_{avg}} = \boldsymbol{(3 * 33 + 4 * 27 + 5 * 25)/12 = 8'd27}$$

Example 2:

$$mode = 2'b00, \ \{n_0, \ldots, n_5\} = \{27,11,10,5,3,1\}$$
$$\text{out\_n} = \boldsymbol{gm_{avg}} = \boldsymbol{(5 + 3 + 1)/3 = 8'd3}$$

The summary of the description and specifications are as followings:

<table>
<tr><th>Input Signal</th><th>Bit Width</th><th>Description</th></tr>
<tr><td>W_0,
V_GS_0,
V_DS_0</td><td>3 / per signal</td><td>ranged from 1~7
unsigned integer</td></tr>
<tr><td>W_1,
V_GS_1,
V_DS_1</td><td>3 / per signal</td><td>ranged from 1~7
unsigned integer</td></tr>
<tr><td>W_2,
V_GS_2,
V_DS_2</td><td>3 / per signal</td><td>ranged from 1~7
unsigned integer</td></tr>
<tr><td>W_3,
V_GS_3,
V_DS_3</td><td>3 / per signal</td><td>ranged from 1~7
unsigned integer</td></tr>
<tr><td>W_4,
V_GS_4,
V_DS_4</td><td>3 / per signal</td><td>ranged from 1~7
unsigned integer</td></tr>
<tr><td>W_5,
V_GS_5,
V_DS_5</td><td>3 / per signal</td><td>ranged from 1~7
unsigned integer</td></tr>
<tr><td>mode</td><td>2</td><td>Mode indicates different operations. The operation will be encode as following:
mode [0]: 1: current, 0: transconductance
mode [1]: 1: larger eq., 0: smaller eq.
<table>
<tr><td>mode</td><td colspan="2">calculation</td></tr>
<tr><td>2’b00</td><td>smaller</td><td>transconductance</td></tr>
<tr><td>2’b01</td><td>smaller</td><td>current</td></tr>
<tr><td>2’b10</td><td>larger</td><td>transconductance</td></tr>
<tr><td>2’b11</td><td>larger</td><td>current</td></tr>
</table></td></tr>
</table>

Output Signal	Bit Width	Description
out_n	8	The answer. Ranged from 0~256

Inputs

1. The input signals W_n, V_GS_n, V_DS_n, for n = 0~5, are 3-bit inputs
2. The input signal mode is a 2-bit input indicates whether to do the operations and which equation to use to get the final result.

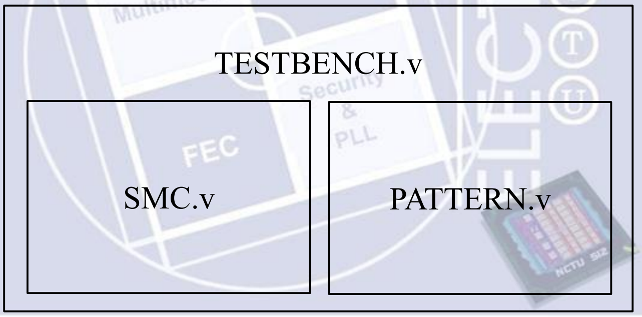
Outputs

The output signal **out_n** is a unsigned number ranged from **0~256**. This represents the correct result.

Specifications

1. Top module name : SMC(File name: SMC.v)
2. After synthesis, check the "SMC.area" and "SMC.timing" in the folder "Report". **The area report is valid only when the slack in the end of "SMC.timing" is "MET".**
3. The synthesis result **cannot** contain any **latch**.
   Note: You can check if there is a latch by searching the keyword "**Latch**" in 02_SYN/**syn.log**

Block Diagram



Grading Policy

The performance is determined by the area of your design. The less area your design has, the higher grade you get. Try to reach better performance by thinking your architecture before coding.

Function Validity: 70%

Performance: area 30%

If you fail Lab01 at first demo, and pass at second demo, you will get 30% off of your original score. Get no score if you fail both first and second demo. Note that you will get 0 score if you are found plagiarism at your code.

Note

1. Tar all your design by run the command **Lab01/09_SUBMIT/00_tar**
2. Submit your design through **Lab01/09_SUBMIT/01_submit**
    a. 1st_demo deadline: 2023/09/25(Mon.) 12:00:00
    b. 2nd_demo deadline: 2023/09/27(Wed.) 12:00:00
3. If your file violates the naming rule, you will lose 5 points.
4. Don't use any wire/reg/submodule/parameter name called *error*, *Congratulations*, *latch* or *FAIL* otherwise you will fail the lab. Note: * means any char in front of or behind the word. e.g: error_note is forbidden.

Be careful about all details!

Template folders and reference commands:

In demo, the reference commands is:

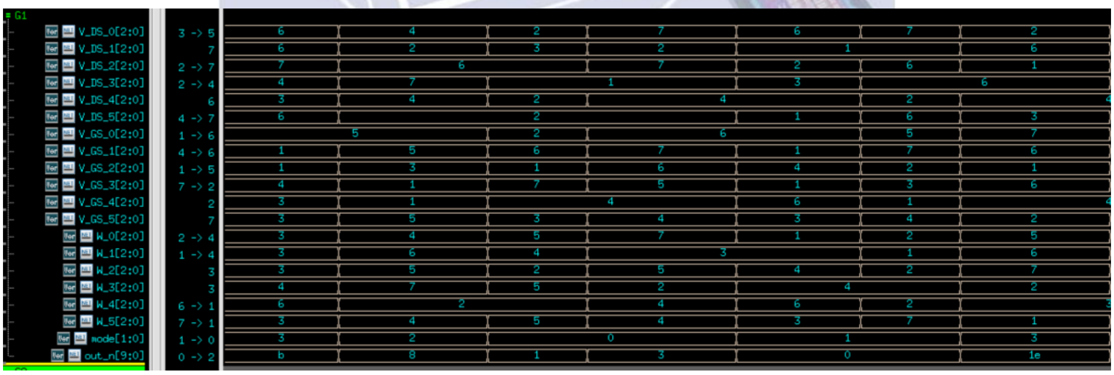
1. 01_RTL (RTL simulation):
    ./01_run_vcs_rtl
2. 02_SYN/ (Synthesis):
    ./01_run_dc_shell
    (Check **latch** by searching the keyword "**Latch**" in 02_SYN/**syn.log**)
    (Check the design's timing in /Report/ SMC.timing)
    (Check the design's area in /Report/ SMC.area)
3. 03_GATE/ (Gate-level simulation):
    ./01_run_vcs_gate
4. 09_SUBMIT/ (submit your files):
    ./00_tar
    ./01_submit
    ./02_check

You can key in **./09_clean_up** to clear all log files and dump files in each folder

```
|-- 00_TESTBED
|   |-- PATTERN.v
|   |-- TESTBED.v
|   |-- Test_data_gen_ref.py
|   |-- filelist.f
|   |-- input.txt
|   |-- makefile
|   `-- output.txt
|-- 01_RTL
|   |-- 01_run_vcs_rtl
|   |-- 02_irun_rtl
|   |-- 03_xrun_rtl
|   |-- 04_verdi
|   |-- 05_nWave
|   |-- 08_check
|   |-- 09_clean_up
|   |-- PATTERN.v -> ../00_TESTBED/PATTERN.v
|   |-- SMC.v
|   |-- TESTBED.v -> ../00_TESTBED/TESTBED.v
|   |-- filelist.f -> ../00_TESTBED/filelist.f
|   `-- makefile -> ../00_TESTBED/makefile
|-- 02_SYN
|   |-- 01_run_dc_shell
|   |-- 02_run_design_vision
|   |-- 03_read_dv_rtl
|   |-- 04_read_ddc
|   |-- 08_check
|   |-- 09_clean_up
|   |-- Netlist
|   |-- Report
|   |-- makefile -> ../00_TESTBED/makefile
|   `-- syn.tcl
|-- 03_GATE
|   |-- 01_run_vcs_gate
|   |-- 02_irun_gate
|   |-- 03_xrun_gate
|   |-- 04_verdi
|   |-- 05_nWave
|   |-- 08_check
|   |-- 09_clean_up
|   |-- PATTERN.v -> ../00_TESTBED/PATTERN.v
|   |-- SMC_SYN.sdf -> ../02_SYN/Netlist/SMC_SYN.sdf
|   |-- SMC_SYN.v -> ../02_SYN/Netlist/SMC_SYN.v
|   |-- TESTBED.v -> ../00_TESTBED/TESTBED.v
|   |-- filelist.f -> ../00_TESTBED/filelist.f
|   `-- makefile -> ../00_TESTBED/makefile
`-- 09_SUBMIT
    |-- 00_tar -> /RAID2/COURSE/BackUp/2023_Spring/iclab/iclabta01/AutoSubmitScript/Lab01/09_SUBMIT/00_tar
    |-- 01_submit -> /RAID2/COURSE/BackUp/2023_Spring/iclab/iclabta01/AutoSubmitScript/Lab01/09_SUBMIT/01_submit
    `-- 02_check -> /RAID2/COURSE/BackUp/2023_Spring/iclab/iclabta01/AutoSubmitScript/Lab01/09_SUBMIT/02_check
```

Example Waveform

Input and output signal:



Hint

Hint1: Try to use **behavior modeling description** instead of gate level description.

Hint2: Try to use **submodule** rather than copy and paste to simply your design. (not necessary in this lab)

```
// -------------------------------------------------
// Example for using submodule
// BBQ bbq0(.meat(meat_0), .vagetable(vagetable_0), .water(water_0),.cost(cost[0]));
// -------------------------------------------------
```

Hint3: Try to think if there is any possible **hardware** that can be **shared** with different mode operation. You can use command dc_shell-gui to examine your design.(not necessary in this lab)

Hint4: Pattern provided by TA will cover only some simple cases, you can try to write your own input / output file by yourself. Here is the format how TA will read in PATTERN:

```
/* input.txt format
1. [PATTERN_NUM]

repeat(PATTERN_NUM)
    1. [mode]
    2. [W_0 V_GS_0 V_DS_0]
    3. [W_1 V_GS_1 V_DS_1]
    4. [W_2 V_GS_2 V_DS_2]
    5. [W_3 V_GS_3 V_DS_3]
    6. [W_4 V_GS_4 V_DS_4]
    7. [W_5 V_GS_5 V_DS_5]
*/

/* output.txt format
1. [out_n]
```

You can check input.txt and PATTERN.v in 00_TESTBED as a reference, and choose to write either c++/python or Verilog code for generating corner cases.

1. 除跟平方用lookup table記

2.