Lab01

SMC

➤ If (V_GS_n – 1 > V_DS_n)   // Triode mode

   ➤ $I_{D_n} = K_n W[2(V_{GS} - V_{th})V_{DS} - V_{DS}^2] = \frac{1}{3}\{W[2(\text{V\_GS\_n} - 1)\text{V\_DS\_n} - \text{V\_DS\_n}^2]\}$

   ➤ $g_{m_n} = 2K_n W V_{DS} = \frac{2}{3}[W\text{V\_DS\_n}]$

➤ If (V_GS_n – 1 <= V_DS_n)   // Saturation mode

   ➤ $I_{D_n} = K_n W(V_{GS} - V_{th})^2 = \frac{1}{3}[W(\text{V\_GS\_n} - 1)^2]$

   ➤ $g_{m_n} = 2K_n W(V_{GS} - V_{th}) = \frac{2}{3}[W(\text{V\_GS\_n} - 1)]$

```
/* V_GS minus one with Kmap */
sub1_V_GS[i][2] = V_GS[i][2] & (V_GS[i][1] | V_GS[i][0]);
sub1_V_GS[i][1] = ~(V_GS[i][1] ^ V_GS[i][0]);
sub1_V_GS[i][0] = ~V_GS[i][0];
```

$V_{gs}[2]$	$V_{gs}[1]$	$V_{gs}[0]$	$V_{gs}[2]$	$V_{gs}[1]$	$V_{gs}[0]$
0	0	0	X	X	X
0	0	1	0	0	0
0	1	0	0	0	1
0	1	1	0	1	0
1	0	0	0	1	1
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0

Formula

> *If (V_GS_n – 1 > V_DS_n)   // Triode mode*

> > $I_{D_n} = K_n W [2(V_{GS} - V_{th})V_{DS} - V_{DS}^2] = \frac{1}{3}\{W[2(V\_GS\_n-1)V\_DS\_n - V\_DS\_n^2]\}$

> > $g_{m_n} = 2K_n W V_{DS} = \frac{2}{3}[W V\_DS\_n]$

> *If (V_GS_n – 1 <= V_DS_n)   // Saturation mode*

> > $I_{D_n} = K_n W (V_{GS} - V_{th})^2 = \frac{1}{3}[W(V\_GS\_n-1)^2]$

> > $g_{m_n} = 2K_n W (V_{GS} - V_{th}) = \frac{2}{3}[W(V\_GS\_n-1)]$

Let
$$A = V\_GS\_n - 1$$
$$B = V\_DS\_n$$

If (V_GS_n – 1 > V_DS_n)

$$I_{Dn} = \frac{1}{3} W [A * (2B - A)]$$

$$g_{mn} = \frac{1}{3} W (2 * B)$$

If (V_GS_n – 1 <= V_DS_n)

$$I_{Dn} = \frac{1}{3} W (A * A)$$

$$g_{mn} = \frac{1}{3} W (2 * A)$$

```
/* (V_GS_n - 1 > V_DS_n) or not */
Comp_GS_DS[i] = sub1_V_GS[i] > V_DS[i] ? 1'b1 : 1'b0;
/* (V_GS_n - 1) * 2 */
sub1_mul2_V_GS[i] = {sub1_V_GS[i], 1'b0};
/* Choose multiplicand with Comp_GS_DS */
Gmn_mul_choose[i] = (Comp_GS_DS[i]) ? V_DS[i] : sub1_V_GS[i];
Idn_mul_choose[i] = (Comp_GS_DS[i]) ? sub1_mul2_V_GS[i] - V_DS[i] : sub1_V_GS[i];
```

Formula

- *If (V_GS_n − 1 > V_DS_n)   // Triode mode*

  - $I_{D_n} = K_n W [2(V_{GS} - V_{th})V_{DS} - V_{DS}^2] = \frac{1}{3}\{W[2(V\_GS\_n - 1)V\_DS\_n - V\_DS\_n^2]\}$

  - $g_{m_n} = 2K_n W V_{DS} = \frac{2}{3}[W V\_DS\_n]$

- *If (V_GS_n − 1 <= V_DS_n)   // Saturation mode*

  - $I_{D_n} = K_n W (V_{GS} - V_{th})^2 = \frac{1}{3}[W(V\_GS\_n - 1)^2]$

  - $g_{m_n} = 2K_n W (V_{GS} - V_{th}) = \frac{2}{3}[W(V\_GS\_n - 1)]$

Let
$$A = V\_GS\_n - 1$$
$$B = V\_DS\_n$$

- **mode[0] = 1**
  - Larger: $I_{total} = 3$
  - Smaller: $I_{total} = 3$
- **mode[0] = 0**
  - Larger: $gm_{total} =$
  - Smaller: $gm_{total} =$

If (V_GS_n − 1 > V_DS_n)

$I_{Dn} = \frac{1}{3} W [\ A * (2B - A)\ ]$

$g_{mn} = \frac{1}{3} W (\ 2 * B\ )$

If (V_GS_n − 1 <= V_DS_n)

$I_{Dn} = \frac{1}{3} W (\ A * A\ )$

$g_{mn} = \frac{1}{3} W (\ 2 * A\ )$

```
/* Choose multiplier with mode[0] */
choose[i] = (mode[0]) ? Idn_mul_choose[i] : 4'd2;
```

Multiplication, Division

If (V_GS_n − 1 > V_DS_n)

$I_{Dn} = \dfrac{1}{3} W [A * (2B - A)]$

$g_{mn} = \dfrac{1}{3} W (2 * B)$

*If (V_GS_n − 1 <= V_DS_n)*

$I_{Dn} = \dfrac{1}{3} W (A * A)$

$g_{mn} = \dfrac{1}{3} W (2 * A)$

```
/* Gmn_Idn_part = choose * Gmn_mul_choose */
Gmn_Idn_part0_tmp[i] = Gmn_mul_choose[i][0] ? choose[i] : 4'd0;
Gmn_Idn_part1_tmp[i] = Gmn_mul_choose[i][1] ? {choose[i], 1'b0} : 5'd0;
Gmn_Idn_part2_tmp[i] = Gmn_mul_choose[i][2] ? {choose[i], 2'b0} : 6'd0;
Gmn_Idn_part[i] = Gmn_Idn_part0_tmp[i] + Gmn_Idn_part1_tmp[i] + Gmn_Idn_part2_tmp[i];

/* Gmn_Idn_part2 = W * Gmn_Idn_part */
Gmn_Idn0_tmp[i] = (W[i][0]) ? Gmn_Idn_part[i] : 6'd0;
Gmn_Idn1_tmp[i] = (W[i][1]) ? {Gmn_Idn_part[i], 1'b0} : 7'd0;
Gmn_Idn2_tmp[i] = (W[i][2]) ? {Gmn_Idn_part[i], 2'b0} : 8'd0;
Gmn_Idn_part2[i] = Gmn_Idn0_tmp[i] + Gmn_Idn1_tmp[i] + Gmn_Idn2_tmp[i];

/* Divide 3 */
Gmn_Idn[i] = Gmn_Idn_part2[i] / 3;
```

Sorting
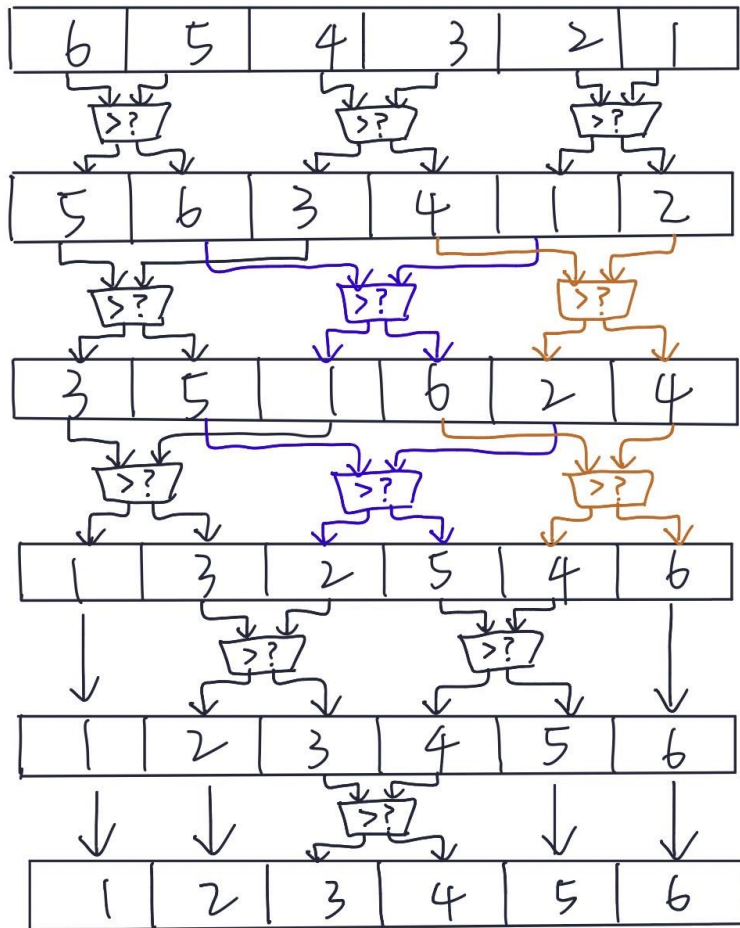

```verilog
always@(*) begin
    a0 = (Gmn_Idn[0] > Gmn_Idn[1]) ? Gmn_Idn[0] : Gmn_Idn[1];
    a1 = (Gmn_Idn[0] > Gmn_Idn[1]) ? Gmn_Idn[1] : Gmn_Idn[0];
    a2 = (Gmn_Idn[2] > Gmn_Idn[3]) ? Gmn_Idn[2] : Gmn_Idn[3];
    a3 = (Gmn_Idn[2] > Gmn_Idn[3]) ? Gmn_Idn[3] : Gmn_Idn[2];
    a4 = (Gmn_Idn[4] > Gmn_Idn[5]) ? Gmn_Idn[4] : Gmn_Idn[5];
    a5 = (Gmn_Idn[4] > Gmn_Idn[5]) ? Gmn_Idn[5] : Gmn_Idn[4];
    b0 = (a0 > a2) ? a0 : a2;
    b1 = (a0 > a2) ? a2 : a0;
    b2 = (a1 > a4) ? a1 : a4;
    b3 = (a1 > a4) ? a4 : a1;
    b4 = (a3 > a5) ? a3 : a5;
    b5 = (a3 > a5) ? a5 : a3;
    c0 = (b0 > b2) ? b0 : b2;
    c1 = (b0 > b2) ? b2 : b0;
    c2 = (b1 > b4) ? b1 : b4;
    c3 = (b1 > b4) ? b4 : b1;
    c4 = (b3 > b5) ? b3 : b5;
    c5 = (b3 > b5) ? b5 : b3;
    d0 = (c1 > c2) ? c1 : c2;
    d1 = (c1 > c2) ? c2 : c1;
    d2 = (c3 > c4) ? c3 : c4;
    d3 = (c3 > c4) ? c4 : c3;
    e0 = (d1 > d2) ? d1 : d2;
    e1 = (d1 > d2) ? d2 : d1;
end
```

Output

- mode[0] = 1
  - Larger: $I_{total} = 3 * n_0 + 4 * n_1 + 5 * n_2$ (mode[1] = 1)
  - Smaller: $I_{total} = 3 * n_3 + 4 * n_4 + 5 * n_5$ (mode[1] = 0)
- mode[0] = 0
  - Larger: $gm_{total} = n_0 + n_1 + n_2$ (mode[1] = 1)
  - Smaller: $gm_{total} = n_3 + n_4 + n_5$ (mode[1] = 0)

```
/* OUTPUT */
wire [6:0] out1 = mode[1] ? c0 : e1;
wire [6:0] out2 = mode[1] ? d0 : d3;
wire [6:0] out3 = mode[1] ? e0 : c5;
wire [7:0] outp1 = (mode[0]) ? {out1, 1'b0} : 8'd0;
wire [8:0] outp2 = (mode[0]) ? {out2, 2'b0} : out2;
wire [8:0] outp3 = (mode[0]) ? {out3, 2'b0} : 9'd0;
assign out_n = outp1 + outp2 + outp3 + out1 + out3;
```